Modeling recurrent circuits of the primary visual cortex using neural network models

Thesis proposal

Yimeng Zhang

Advisor: Tai Sing Lee

Center for the Neural Basis of Cognition and Computer Science Department
Carnegie Mellon University

This document is submitted for the degree of
Doctor of Philosophy

November 2018

Abstract

There has been great interest in the primary visual cortex (V1) since pioneering studies decades ago. However, existing models cannot explain V1 neural responses to complex stimuli satisfactorily. One possible reason for this failure is the models' lack of recurrent connections, which form the bulk of synaptic connections in V1 and greatly contribute to the complexity of the visual system.

The goal of my thesis is to develop, test, and understand neural network models of recurrent circuits in V1 and the visual system in general. I have completed two studies and propose to finish an additional study.

My first study has demonstrated that the Boltzmann machine, a type of recurrent neural network, is useful for conceptualizing certain V1 recurrent computations.

My second study has demonstrated that the CNN's key components are crucial to its superior performance in explaining V1 data and are consistent with previous V1 studies. While not directly related to recurrent circuits, this project has demonstrated that neural response prediction is a useful metric for selecting models with high correspondence with biological reality. The metric and analysis methods here will be used in the proposed work.

In the final project, I propose to advance our understanding of recurrent circuits of V1 in a two-part investigation. First, I will find candidate models for V1 recurrent circuits, by designing models with recurrent computation components for predicting neural responses as well as predicting certain phenomena observed in V1 studies. The models to be explored will feature two new complementary designs: model architecture and training methodology. Preliminary results show that models with these designs can perform as well as state-of-the-art approaches using fewer parameters and less data.

Second, I will explain the role of recurrent connections in the candidate models for modeling V1 using various tools in machine learning as well as existing knowledge about V1.

Overall, this investigation will provide new neural network models with recurrent connections for explaining more V1 phenomena with higher accuracy, establish correspondence between model components and biological reality, and provide new insights about the roles of recurrent circuits in V1 and visual signal processing in general.

Table of contents

Chapter 1

Introduction

1.1 Background and motivation

There has been great interest in the primary visual cortex (V1) since pioneering studies decades ago [60, 58, 59]. V1 neurons are traditionally classified as simple and complex cells, which are modeled by linear-nonlinear (LN) models [52] and energy models [1], respectively. By construction, these standard V1 models (LN models and energy models) only respond to stimulus change with in neurons' classical receptive fields or CRFs [50], which are classically defined as regions in the visual space where the presence or absence of small, impulse like stimuli (bright or dark dots) can cause change in neural response.

The above standard V1 models have been successful in explaining V1 response to relatively simple stimuli, such as bars, edges and gratings [117]. However, they cannot explain satisfactorily neural responses to more complex stimuli such as complex shapes and natural images [27, 140, 53, 77, 134, 11], under which V1 neurons exhibit complex nonlinear response properties; these properties not predicted by standard models are collectively called *non-classical receptive field (nCRF) effects* [160] in this document. When natural stimuli are used, V1's nCRF effects result in a significant portion (no less than 50 %) of variance in neural responses left unexplained by most existing models, including those state-of-the-art ones based on neural networks (see below), for predicting V1 data [27, 77, 11].

One possible reason for the failure of existing models in terms of explaining neural responses to natural stimuli is the lack of recurrent connections in these models. In the brain, it is well known that there are local horizontal recurrent connections between neurons in the same area [65] and long-range feedback recurrent connections between neurons in different areas [39]. By some count [34], more than 90% of the excitatory synapses and virtually all of the inhibitory ones in V1 come from recurrent connections. These connections greatly contribute to the complexity of the visual system, and may be essential for the success of

the visual systems in reality; for example, there are evidences that recurrent connections are crucial for object recognition under noise, clutter, and occlusion [102, 129, 112].

One possible way to study the potential role of recurrent connections in nCRF effects and other phenomena in V1 is to model them using neural networks. In recent years, neural network models have been used to study various visual areas with great success (see Section 3.2 for details). As a consequence of their current success and biological realism relative to other computational models, neural network models provide a viable tool for exploring computational (and, to a lesser extent, mechanistic) mechanisms of V1 and early visual areas in general. Furthermore, as neural network models in general allow end-to-end training more easily than other computational models, V1 models based on neural networks can be trained on data sets containing large numbers of neurons and complex, natural stimuli to test the validity of these models in a more realistic setting. In contrast, studies on V1 nCRF effects based on other computational models (see Section 3.1 for details) involve a significant amount of "hand crafting": filter weights that are fixed and may not generalize to complex natural stimuli, architectures that specifically engineered for specific experiments conducted in the studies, etc.; therefore, the utility of (most of) these models for explaining neural responses to large numbers of complex, natural stimuli is limited. Recent studies have begun to explore the benefits of recurrent connections in many machine learning settings [97, 25, 91, 88]; however, the role recurrent connections play in the primary visual cortex and early visual areas in general remains unexplained.

1.2 Summary of the proposal

The overall goal of my thesis is to develop, test, and understand neural network models of recurrent circuits in the primary visual cortex.

My first project [158] attempted to learn horizontal recurrent connections between V1 disparity-tuned neurons by learning from 3D natural scene data. The learned connectivity patterns were consistent with connectivity constraints in stereopsis models [92, 120], and simulated neurophysiological experiments on the model were consistent with neurophysiological data in terms of functional connectivities among disparity-tuned neurons in V1 [119]. This study has demonstrated that the Boltzmann machine, which is inspired by neuroscience but often dismissed to be too abstract and different from the real brain, is in fact a useful and viable model for conceptualizing certain recurrent computations in the primary visual cortex.

My second project [157] systematically evaluated the relative merits of different (feedforward) CNN components in the context of modeling V1 neurons. We demonstrated that key components of the CNN (convolution, thresholding nonlinearity, and pooling) contributed to

its superior performance in explaining V1 responses to complex stimuli [134] and these key components are consistent with previous V1 modeling and neurophysiology studies. While this project is not directly related to recurrent circuits, it has demonstrated that predicting neural responses to natural and complex stimuli accurately is a useful objective metric for identifying neural network models with high correspondence with biological reality; in addition, it has shown the usefulness of various ablation, dissection, and visualization methods for comparing and understanding neural network models of different architectures. The metric and methods in this study have laid the foundations for developing and analyzing models with recurrent computation components in the proposed work.

In the proposed final project in this document, I plan to advance our understanding of recurrent circuits of the primary visual cortex and the visual system in general, in a two-part investigation.

First, I will try to find candidate models for recurrent circuits of V1, by designing and evaluating different neural network models with recurrent computation components for predicting V1 neural responses to natural images as well as predicting other phenomena observed in V1 neurophysiology studies. The rationale here is that better-performing models for V1 data may have more similarities to the biological reality [157, 151]. Compared to feedforward neural network models used in typical data-driven and transfer learning methods (Section 3.2), the candidate models to be explored will feature new designs from two complementary aspects: model architecture (recurrent vs. feedforward computation components) and training methodology (loss functions, number of training phases, etc.). Preliminary results show that models with these two new designs can perform as well as state-of-the-art approaches with fewer parameters and less data.

Second, I will try to explain the role of recurrent computation components in these high-performing models for explaining V1 data, by first simplifying models with various model compression techniques without sacrificing much performance and then analyzing the simplified models using various tools for neural network analysis as well as existing knowledge about V1. In particular, I propose to use recurrent computation components in my candidate models to 1) provide alternative and more-detailed explanation of contextual modulation [23], a well-known phenomena in V1, and 2) explain familiarity effect in early visual areas [56], a newly found phenomena in V2 and early visual areas in general. The end product of achieving any of the two above goals would be a valuable contribution to NIPS or other high-profile neuroscience conferences and journals.

Overall, this investigation will provide new neural network models with recurrent computational components for explaining more V1 phenomena with higher accuracy, establish correspondence between model components and biological reality, and hopefully provide

new insights about the roles of recurrent circuits in V1 and visual signal processing in general in the context of contextual modulation and familiarity effect.

I will summarize my two completed studies on V1 in Chapter 2 and present the proposed work plus preliminary results in Chapter 4; in addition, background knowledge on the proposed work is provided in Chapter 3.

1.3 Timeline

- November 2018–May 2019, finish proposed work in Chapter 4.
- May 2019–July 2019, thesis writing and defense.

1.4 List of finished works

1. Relating functional connectivity in V1 neural circuits and 3D natural scenes using Boltzmann machines [158], see Section 2.1.
2. Convolutional neural network models of V1 responses to complex patterns [157], see Section 2.2.

Chapter 2

Two completed studies on the computational mechanisms of V1

In this chapter, I will first present the main results of my two completed studies on V1 with Zhang et al. [158] in Section 2.1 and Zhang et al. [157] in Section 2.2; then I will discuss their relationships to the proposed work (Section 2.3).

2.1 Relating functional connectivity in V1 neural circuits and 3D natural scenes using Boltzmann machines

Bayesian theory has provided a compelling conceptualization for perceptual inference in the brain. Central to Bayesian inference is the notion of statistical priors. To understand the neural mechanisms of Bayesian inference, we need to understand the neural representation of statistical regularities in the natural environment. In this paper, we investigated empirically how statistical regularities in natural 3D scenes are represented in the functional connectivity of disparity-tuned neurons in the primary visual cortex of primates. We applied a Boltzmann machine model to learn from 3D natural scenes, and found that the units in the model exhibited cooperative and competitive interactions, forming a "disparity association field", analogous to the contour association field. The cooperative and competitive interactions in the disparity association field are consistent with constraints of computational models for stereo matching. In addition, we simulated neurophysiological experiments on the model, and found the results to be consistent with neurophysiological data in terms of the functional connectivity measurements between disparity-tuned neurons in the macaque primary visual cortex. These findings demonstrate that there is a relationship between the functional connectivity observed in the visual cortex and the statistics of natural scenes.

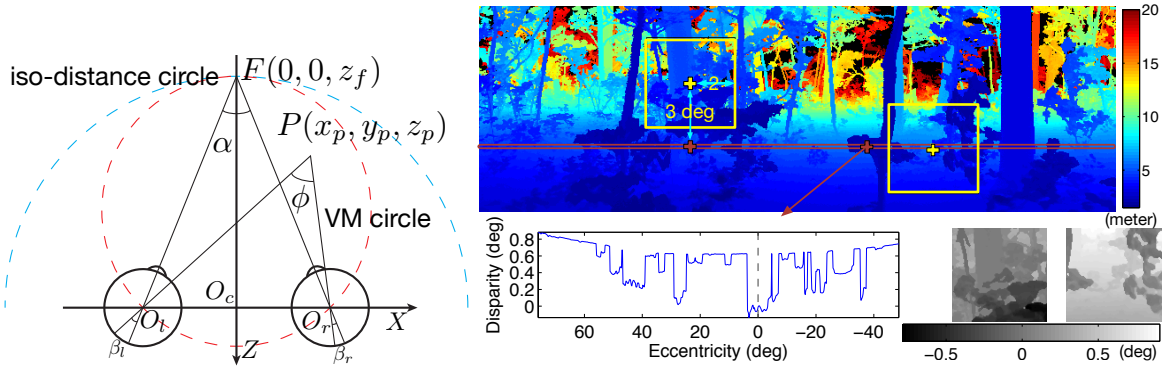
They also suggest that the Boltzmann machine can be a viable model for conceptualizing computations in the visual cortex and, as such, can be used to predict neural circuits in the visual cortex from natural scene statistics.

2.1.1 Introduction

Natural scenes contain significant ambiguity. To resolve ambiguities and obtain a stable 3D percept of the world, the visual system (as well as the whole brain) must perform inference, integrating current sensory data with prior knowledge of the world formulated from past experience. Therefore, (Bayesian) inference has long been proposed as a fundamental computational principle of the brain [144, 74]. In this work, we attempt to address one of the key questions for understanding Bayesian inference in the brain, in the context of the primary visual cortex (V1): how might an internal model of natural scenes—the Bayesian prior—be encoded in the brain?

To support visual inference, an internal representation of the visual scenes requires encoding both the statistical regularities of the boundaries and of the surfaces themselves. There have been studies suggesting that the neural circuits in the primary visual cortex (V1) encode contour priors in the form of the contour association field [40, 64, 44, 38, 10, 85, 96, 119, 120]. Recent neurophysiological evidence suggests that disparity-tuned neurons in the primary visual cortex might form a recurrent network for stereo processing [119, 120]. This network encodes the statistical correlation of disparity signals in natural scenes, complementing the contour association field, and is referred to as the disparity association field. However, the neural mechanisms by which statistical priors of boundaries and surfaces from the environment can be learned are not well understood.

We hypothesize that the empirically observed neural connectivity between disparity-tuned neurons in V1 can be predicted from 3D natural scenes using a Boltzmann machine. To test this hypothesis, we fitted a Boltzmann machine neural network model [54] to disparity signals derived from 3D natural scene data, and found that 1) learned parameters in the model were consistent with connectivity constraints in stereopsis models [92, 120]; 2) the model was consistent with neurophysiological data in terms of functional connectivities among disparity-tuned neurons in V1 [119]. The results provide further evidence in support of the notion of the disparity association field, and demonstrate that the Boltzmann machine is a viable model for describing cortical computation in the sense that they can be used to predict functional neural circuitry in the visual cortex.



(a) (b)

Fig. 2.1 **(a)** Diagram for calculating disparity. Adapted from Liu et al. [89]. See Eqs. (1)-(3) of Zhang et al. [158] for detail. **(b)** One sample range image from the Brown data set (upper) with disparity values along one line in it (lower left), and two extracted disparity patches (lower right). In the upper image: red crosses, fixations points for two patches; yellow crosses, center of patches; red long rectangle, the row shown disparities. Patches were 3° away from fixation and had a half-width of 2°.

2.1.2 Methods

2.1.2.1 3D scene data

We trained a Boltzmann machine to model the disparity signals over a small visual field. These signals were derived from the Brown Range Image Database [57]. A total of 200K disparity image patches with a 2° half-width were extracted from 172 images (54 forest, 49 interior, 69 residential). The images in the Brown data set were captured by a scanner with range at 2 m to 200 m, and image resolutions were approximately 5 pixels per degree of visual angle.

Disparity image patches were generated from each range image as follows (Figure 2.1b). A random point in the range image was chosen as the fixation point. Given the fixation point, the disparities at its surrounding pixels were computed using the method in Liu et al. [89]. Finally, a disparity image patch with a 2° half-width was extracted 3° away from the fixation point. This eccentricity was chosen to roughly match the typical receptive field locations of recorded V1 neurons in our earlier neurophysiological experiments.

2.1.2.2 Boltzmann machines

Interaction among neurons modeled by Boltzmann machines The extracted disparity image patches reflect the prior of disparity signals in the natural scene, and we modeled this prior by fitting a Boltzmann machine to the patches. Boltzmann machines [54] are a

class of stochastic recurrent neural network models that can learn internal representations to explain or generate the distribution of the input data, using pairwise connectivity between units to encode the structures of the input data. Boltzmann machines are also a type of Markov random fields, which are widely used in computer vision for solving a variety of early vision problems such as surface interpolation and stereo inference [45, 75, 7, 135]. We hypothesize that Boltzmann machines are a viable computational model for understanding the circuitry of the visual cortex, and we will examine if they can explain interactions among neurons in other computational and neurophysiological studies [92, 119, 120]. Specifically, the interaction terms $\vec{\boldsymbol{\beta}}$ (Eq. (2.1)) in our Boltzmann machine model were compared with existing computational models in Section 2.1.3.2, and neurophysiological experiments were simulated on the model (Section 2.2.2 of Zhang et al. [158]) to compare it with neural data in Section 3.3 of Zhang et al. [158].

The units in our Boltzmann machine model (Figure 2.2a) are organized into a hidden layer and a visible layer, arranged in a spatial 5 by 5 grid of "hypercolumns" (in total $C = 25$ columns). Each hypercolumn has a bank of $M = 16$ visible units that encode the disparity input, and a bank of 16 corresponding hidden units $\vec{\boldsymbol{h}}$, all sharing the same spatial receptive field location. The $N = MC = 400$ hidden units are fully connected, each of them driven by its corresponding input visible unit. The collective spiking activity at each bank of visible units encodes the disparity signal at the corresponding hypercolumn.

This model is formally expressed as a probability distribution over hidden and visible units:

$$P(\vec{\boldsymbol{h}},\vec{\boldsymbol{v}};\vec{\boldsymbol{\alpha}},\vec{\boldsymbol{\beta}},\vec{\boldsymbol{\gamma}},\vec{\boldsymbol{\lambda}}) = \frac{1}{Z}\exp\left(\sum_{i=1}^{N}\alpha_i h_i + \sum_{i<j}\beta_{i,j}h_i h_j + \sum_{i=1}^{N}\lambda_i h_i v_i + \sum_{i=1}^{N}\gamma_i v_i\right). \tag{2.1}$$

In Eq. (2.1), $\vec{\boldsymbol{h}}$ and $\vec{\boldsymbol{v}}$ are binary vectors whose distributions are to be captured by the model, representing spiking activities of hidden and visible units. The other model parameters capture the distributions of $\vec{\boldsymbol{h}}$ and $\vec{\boldsymbol{v}}$, as well as their interactions. Specifically, $\vec{\boldsymbol{\alpha}}$ and $\vec{\boldsymbol{\gamma}}$ capture the baseline firing rates of hidden and visible units, $\vec{\boldsymbol{\beta}}$ models the pairwise lateral interactions among hidden units, and $\vec{\boldsymbol{\lambda}}$ models the interactions between hidden and visible units. Z is a normalization constant.

This Boltzmann machine was fitted by finding parameters $\vec{\boldsymbol{\alpha}}$, $\vec{\boldsymbol{\beta}}$, $\vec{\boldsymbol{\gamma}}$, and $\vec{\boldsymbol{\lambda}}$ that maximize the probability of the model for generating the spike patterns $\vec{\boldsymbol{v}}$, corresponding to the disparity signals in the extracted patches. Formally, the following log likelihood was maximized:

$$L(\vec{\boldsymbol{\alpha}},\vec{\boldsymbol{\beta}},\vec{\boldsymbol{\gamma}},\vec{\boldsymbol{\lambda}}) = \sum_{i=1}^{T}\log P(\vec{\boldsymbol{v}}^{(i)};\vec{\boldsymbol{\alpha}},\vec{\boldsymbol{\beta}},\vec{\boldsymbol{\gamma}},\vec{\boldsymbol{\lambda}}) = \sum_{i=1}^{T}\log\sum_{j=1}^{2^N}P(\vec{\boldsymbol{h}}^{(j)},\vec{\boldsymbol{v}}^{(i)};\vec{\boldsymbol{\alpha}},\vec{\boldsymbol{\beta}},\vec{\boldsymbol{\gamma}},\vec{\boldsymbol{\lambda}}). \tag{2.2}$$

In Eq. (2.2), $\vec{\boldsymbol{v}}^{(i)}$'s are T binary spike patterns of the visible units converted from the disparity signals based on the tuning curves of the visible units (see Figure 2.3 and Section 2.1.2.2). The likelihood of observing $\vec{\boldsymbol{v}}^{(i)}$ is computed as the sum of $P(\vec{\boldsymbol{h}}^{(j)}, \vec{\boldsymbol{v}}^{(i)})$ over all possible 2^N hidden unit patterns, which is the marginal probability for the model to generate $\vec{\boldsymbol{v}}^{(i)}$, regardless of the hidden units. Finally, the log probability for the model to generate all the input spike patterns due to the disparity signals is computed as the sum of log probabilities for generating each particular spike pattern $\vec{\boldsymbol{v}}^{(i)}$. The model was trained using contrastive divergence mean field learning [148]. See Section 2.2.3 of Zhang et al. [158] and Welling and Hinton [148] for more detail.

Conversion of disparity signals into binary spike patterns From each disparity image patch i, disparity values $s^i_1, s^i_2, \ldots, s^i_{C=25}$ corresponding to the locations of the 25 hypercolumns were extracted, and the model was fitted to explain these disparity values across all patches. Disparity signals are real-valued, and must be converted into binary spike patterns, which can be considered as the spiking activities of the bottom-up input to V1 neurons. Following the approach of Ganguli and Simoncelli [42], we derived a set of $M = 16$ tuning curves for visible units (same for all the hypercolumns, Figure 2.2c) according to the distribution $P(s)$ of extracted disparity values from all patches (Figure 2.2b). Each disparity value was converted to the mean firing rates of $M = 16$ visible units based on their tuning curves.

Given the above derived tuning curves, for each patch, we first converted the $C = 25$ disparity values into the mean firing rates of the $N = 400$ visible units. Then for each of these N units, a spike train of 200 ms was generated based on its mean firing rate using an independent homogeneous Poisson process, and the whole spike train was partitioned into 20 bins of 10 ms[1]. A "1" was assigned to a bin of a unit if there were one or more spikes for that unit within that time bin; otherwise, a "0" was assigned. The whole generation process (for one disparity value) is schematically shown in Figure 2.3.

2.1.3 Results

We mainly compared the model with existing computational models in terms of connectivity constraints (Section 2.1.3.2), and neurophysiological data in terms of functional connectivities (Section 3.3 of Zhang et al. [158]). The model showed qualitative agreement in both aspects. In the following comparisons, the hidden units correspond to the disparity-tuned V1 neurons, likely realized in terms of the complex cells in the superficial layer of V1 where there are

[1] Our implementation was written in terms of bins, with no notion of the physical duration of each bin. We arbitrarily assumed each bin to be of 10 ms, for easier comparison with neurophysiological data and other studies based on Ising models (a type of Boltzmann machines).

extensive horizontal axonal collaterals forming a recurrent network. The visible units provide the bottom-up input to these V1 complex cells, and they encode disparity signals which in the brain are computed by combining monocular left and right eye simple cells based on phase-shift or position-shift mechanisms [41]. The input from visible units, or the corresponding signals in the brain, are assumed to be "decorrelated" across space when stimulus correlation is factored out [37]. The prior of natural scenes is assumed to be captured by the lateral connectivity among hidden units in the model or among disparity-tuned V1 neurons in the brain. These intrinsic horizontal connections can give rise to noise correlation and other correlated activities among neurons [128, 68, 24].

2.1.3.1 First order properties of learned hidden units

Figure 2.4 shows typical tuning curves of the hidden units obtained from the model simulation of neurophysiological experiments (Section 2.2.2 of Zhang et al. [158]), and the distribution of bias terms $\vec{\boldsymbol{\alpha}}, \vec{\boldsymbol{\gamma}}$. Hidden units shared the same preferred disparity and the general shape as their corresponding input visible units. The bias terms are negative, indicating that the hidden units tend to fire sparsely.

2.1.3.2 Comparison with computational models in terms of connectivity constraints

The learned lateral connections $\vec{\boldsymbol{\beta}}$ among hidden units form what we call the **disparity association field**, analogous to the well-known contour association field for orientation-tuned neurons [40]. The lateral connectivity, or the disparity association field, observed in the trained Boltzmann machine model is qualitatively in agreement with the cooperative and competitive circuits predicted by Marr and Poggio [92], and with the recent model of Samonds et al. [120] which has been successful in more accurately accounting for neurophysiological data of disparity-tuned neurons in V1.

We define the disparity association field of a hidden unit as the set of lateral connections between it and other hidden units. Figure 2.5a illustrates the disparity association field of one unit tuned near zero disparity in the center column of the 5×5 grid, showing its lateral connections $\vec{\boldsymbol{\beta}}$ to all other units in the network along a particular direction in the grid. The x-axis indicates different hypercolumns or spatial locations, and the y-axis indicates units with different disparity tunings.

The disparity association field learned by the Boltzmann machine has a number of noteworthy features. First, in terms of **inter-columnar connections**, i.e. connections between a unit with units in other hypercolumns, units with the same or similar disparity tunings tended to form positive connections across hypercolumns (spatial receptive field locations)

and units with very different disparity tunings formed negative connections. Figures 2.5b and 2.5c show in greater detail how each unit in one hypercolumn was connected to units of various disparity tunings in other hypercolumns. The dark bold line highlights that unit 8 in one hypercolumn formed positive (excitatory) connections to similarly tuned units (units 6, 7, 8, 9) in the other hypercolumns, and negative (inhibitory) connections to units tuned to very different disparities. Second, in terms of **intra-columnar connections**, i.e. connections between units in the same hypercolumn, units exhibited excitation for very similarly tuned units in the same hypercolumn, but exerted a suppressive effect on units of dissimilar tuning properties, as shown in Figure 2.5d. These properties of inter- and intra-columnar connections are roughly consistent with the cooperation between neurons of similar disparities across space (the so-called continuity constraint), and the competition among neurons of different disparities at the same spatial location (the so-called uniqueness constraint) in Marr and Poggio [92]'s classical stereopsis model for solving the correspondence problem.

However, the lateral connectivity exhibited by the Boltzmann machine model was richer than that in Marr and Poggio [92]'s model. First, in terms of intra-columnar connections, in addition to the negative (competitive) intra-columnar connections in Marr and Poggio [92]'s model (Figure 2.6a, blue), our model also learned positive intra-columnar connections among units of similar tunings (Figure 2.6b). In this aspect, our model is more consistent with the model in Samonds et al. [120], which assumes that the intra-columnar interaction has a center excitatory (cooperation between similar neurons) surround inhibitory (competition between dissimilar neurons) profile. This profile is more biologically realistic than that of Marr and Poggio [92], taking into account the overlapping nature of tuning curves within a hypercolumn, and the model in Samonds et al. [120] has been shown to explain well the temporal evolution of a number of tuning properties of V1 disparity-tuned neurons.

Second, in terms of inter-columnar connections, Marr and Poggio [92]'s model only specifies positive inter-columnar connections between neurons of the same tuning (Figure 2.6a, red), implicitly making the strong assumption that the surfaces of the world are all fronto-parallel. However, surfaces in natural scenes are more diverse, characterized with a variety of surfaces such as slants and tilts, convexities and concavities. This richness in natural scene surface structures likely induced the greater variety of inter-columnar connectivity observed in our model (Figure 2.6c) that captures the 3D surface priors to a higher degree than connectivity constraints made in the works of Marr and Poggio [92] and Samonds et al. [120]. Our model is likely more consistent with more advanced computational models for stereopsis that take into account slant, tilt, and curvature [83, 7, 110].

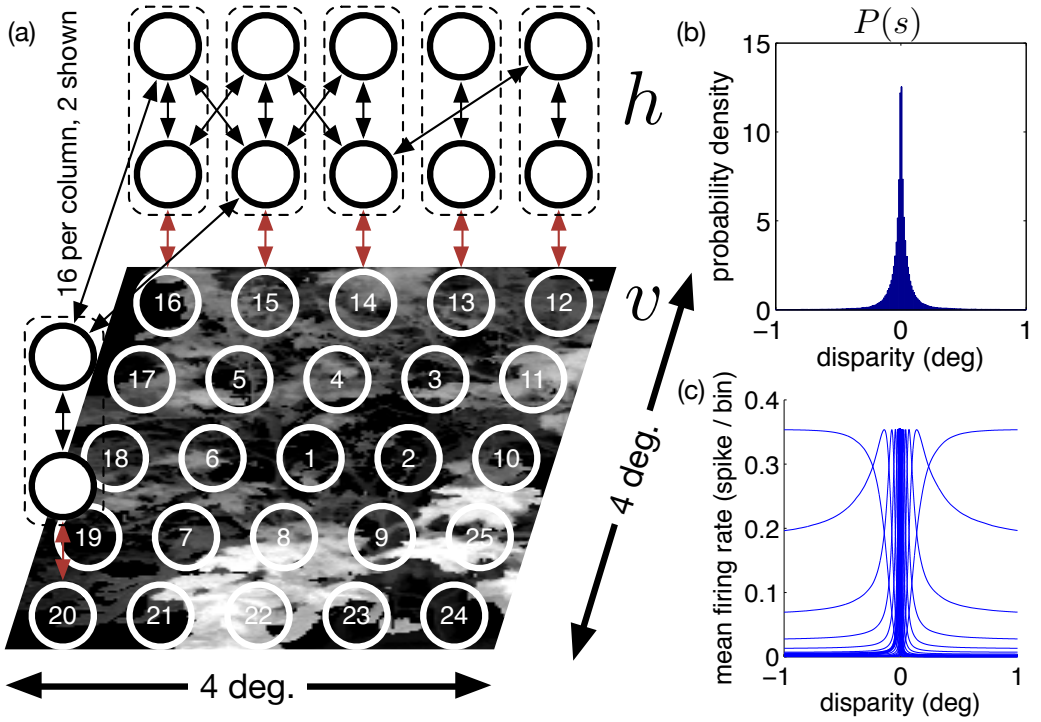


Fig. 2.2 Schematic of our Boltzmann machine model (a), distribution of extracted disparity values $P(s)$ (b), and derived tuning curves of input visible units, with one curve highlighted (c). **(a)** 25 "hypercolumns" laid in a 5 by 5 grid covering a 4° by 4° patch, with hidden units ($\vec{\boldsymbol{h}}$, black outline) in the same column grouped in dashed box. Each hidden unit has connections (black) to all other ones, and one connection (red) to its own visible unit ($\vec{\boldsymbol{v}}$, white outline). At most two hidden units and one visible unit drawn per column, with many connections missing for clarity. Columns are numbered for reference in Section 2.1.3. **(b)** The distribution of extracted disparity values was sharp, and peaked at zero. **(c)** Tuning curves of $\vec{\boldsymbol{v}}$ were derived based on Ganguli and Simoncelli [42] with the following details: "baseline" curve was a t distribution with d.o.f. $\nu = 2$, total expected firing rate (R in Ganguli and Simoncelli [42]) was unity, and "infomax" optimization criterion was used. Only tuning curves between $-1°$ and $1°$ are shown for clarity. Given the sharp distribution of disparity values, the theory in Ganguli and Simoncelli [42] made the tuning curves at large disparities different from those close to zero. Instead of Gaussian-like (see Figure 2.3a for a zoom-in view of tuning curves close to zero), the tuning curves at the two ends of the input distribution were relatively flat at large (positive/negative) disparities, and dropped to zero near zero disparity. Interestingly, these were very similar to the near-tuned and far-tuned neurons in Poggio and Fischer [108] and Poggio et al. [109]. We also tried Gaussian distribution as the "baseline" curve, but that gave much sharper tuning curves and less co-activation between dissimilar units, which resulted in a less biologically realistic training result.

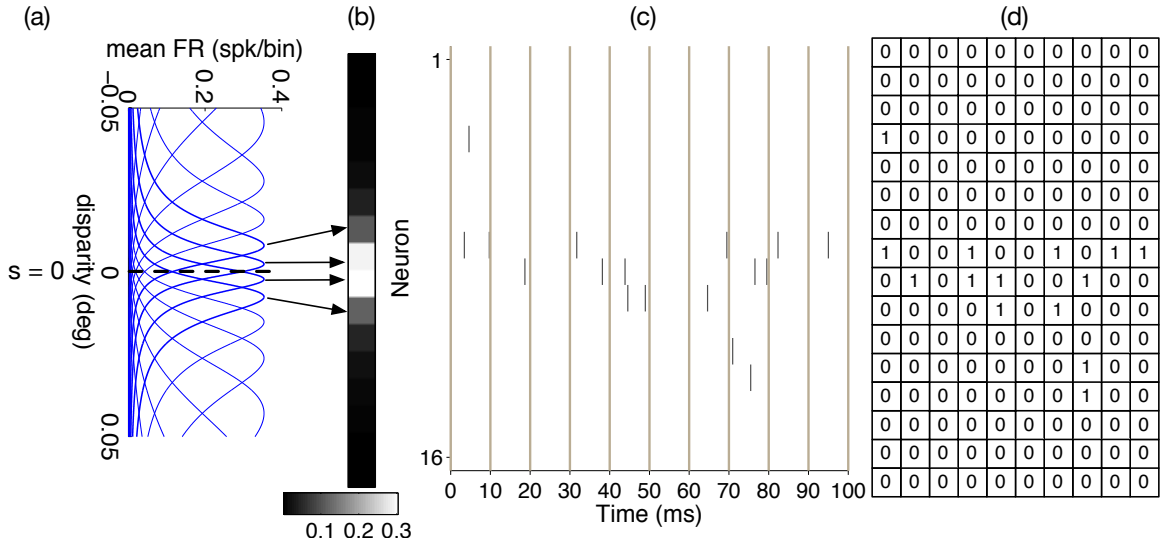


Fig. 2.3 Generation of training data for one disparity value. Given one disparity value **(a)** (in this case $s = 0$), we transformed it into $M = 16$ mean firing rates (b) using tuning curves (between **(a)** and **(b)**), generated spike trains **(c)**, and binned it into a binary matrix **(d)** as training data to the Boltzmann machine.

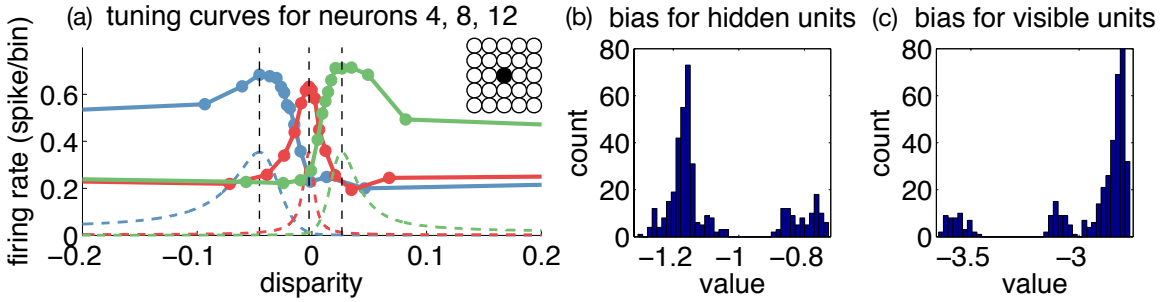


Fig. 2.4 Some first order properties of the trained Boltzmann machine model. **(a)** tuning curves of hidden units 4, 8, 12 in column 1 of the model (inset on top right), shown as solid curves, with corresponding tuning curves of their input units shown as dashed curves. Tuning curves for hidden units were computed using the mean firing rates during simulation at different testing stimuli (dots on curves). **(b)** Histogram of bias terms for hidden units. **(c)** Histogram of bias terms for visible units. The cluster of values around -0.8 for hidden units and that around -3.6 for visible units mostly came from units tuned to two ends of the disparity distribution (large negative / positive disparities), which was due to a border effect of the theory in Ganguli and Simoncelli [42].

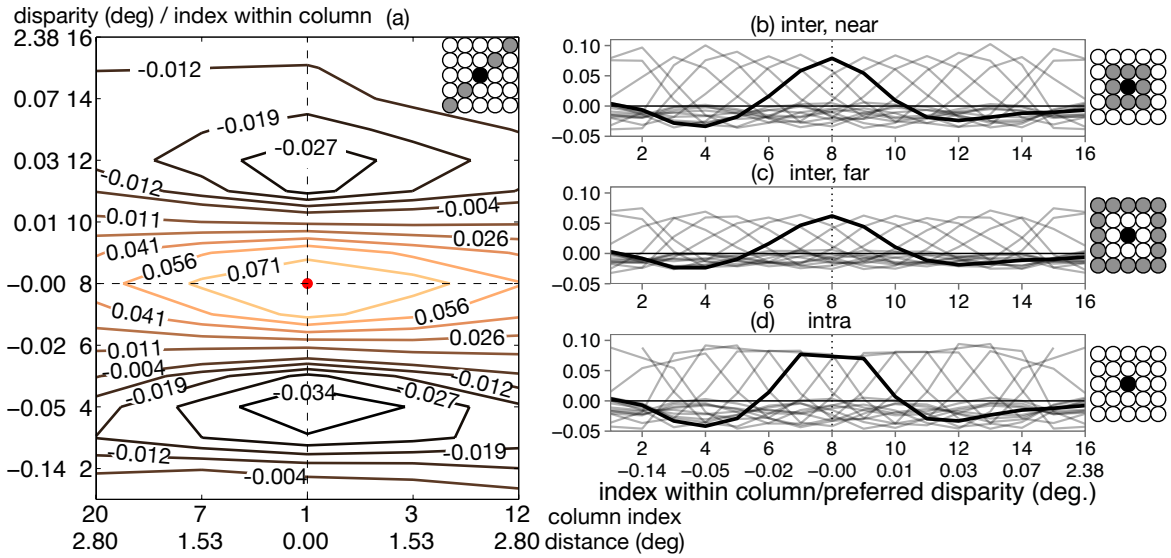


Fig. 2.5 Disparity association field. **(a)** disparity association field from unit 8 in the central hypercolumn (column 1) to all other units in 5 hypercolumns (inset on top right; black for column 1, and gray for all 5 but column 1), shown in a contour map of lateral connections. The shapes of contour lines resemble those in the contour association field [40]. The horizontal axis has two rows, first column index, and second distance to column 1; the vertical axis has two columns, from left to right being the preferred disparity of units, and index of the unit in its hypercolumn (1 to 16). **(b)** Inter-columnar connections from column 1 to 8 nearby columns. Each curve shows the average connections between one unit in column 1 to unit of a certain index in other 8 columns (inset on right; black for column 1, and gray for other 8). The curve for unit 8 in the central column is highlighted for clarity. Its value at, say, index $= 9$ is the average of connections from unit 8 in the central column to every unit 9 in surrounding 8 columns, and so on. **(c)** Inter-columnar connections from column 1 to 16 columns further away (inset on right). The connections were generally weaker than those in the panel above, due to longer distances between columns. **(d)** Intra-columnar connections within column 1 (inset on right), with the curve for unit 8 highlighted for clarity. Units of very similar tunings tended to facilitate and help each other, but units of different tunings would inhibit each other.

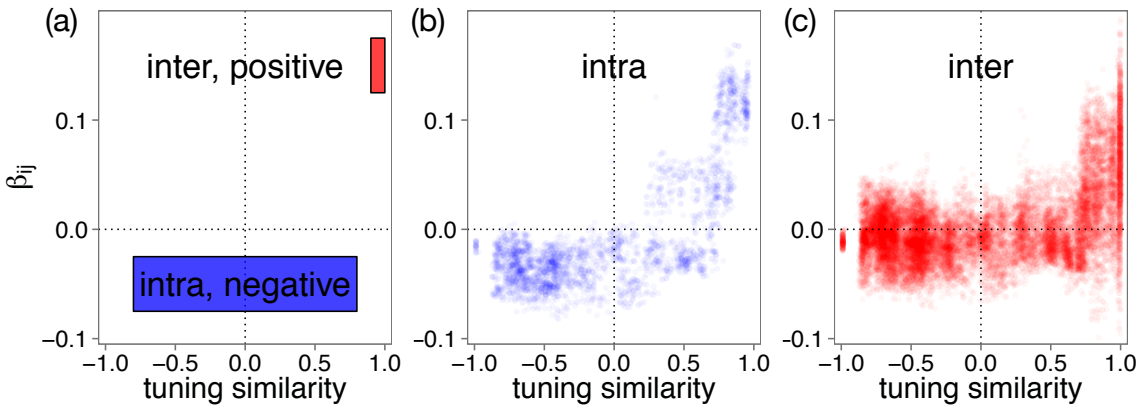


Fig. 2.6 Comparison of connectivity in Marr and Poggio [92]'s model and in our Boltzmann machine model, in terms of scatter plot of connection vs. tuning similarity between neurons. **(a)** Schematic of Marr and Poggio [92]'s model, with negative intra-columnar connections between all neurons of different tunings (blue), and positive inter-columnar connections only between neurons of the same tuning (red). **(b)** Intra-columnar connections of our model. **(c)** Inter-columnar connections of our model. The tuning similarity between pairs of hidden units is defined as the Pearson's correlation coefficient between their tuning curves measured from the model simulation (Section 2.2.2 of Zhang et al. [158]).

2.2 Convolutional neural network models of V1 responses to complex patterns

In this study, we evaluated the convolutional neural network (CNN) method for modeling V1 neurons of awake macaque monkeys in response to a large set of complex pattern stimuli. CNN models outperformed all the other baseline models, such as Gabor-based standard models for V1 cells and various variants of generalized linear models. We then systematically dissected different components of the CNN and found two key factors that made CNNs outperform other models: thresholding nonlinearity and convolution. In addition, we fitted our data using a pre-trained deep CNN via transfer learning. The deep CNN's higher layers, which encode more complex patterns, outperformed lower ones, and this result was consistent with our earlier work on the complexity of V1 neural code. Overall, we believe this study is the first one that systematically evaluates the relative merits of different CNN components in the context of modeling V1 neurons. We demonstrated that key components of the CNN (convolution, thresholding nonlinearity, and pooling) contributed to its superior performance in explaining V1 responses. Our results suggests that that there is a high degree of correspondence between the CNN and biological reality.

2.2.1 Introduction

There has been great interest in the primary visual cortex (V1) since pioneering studies decades ago [60, 58, 59]. V1 neurons are traditionally classified as simple and complex cells, which are modeled by linear-nonlinear (LN) models [52] and energy models [1], respectively. However, a considerable gap between the standard theory of V1 neurons and reality has been demonstrated repeatedly, at least from two aspects. First, although standard models explain neural responses to simple stimuli such as gratings well, they cannot explain satisfactorily neural responses to more complex stimuli, such as natural images and complex shapes [27, 140, 53, 77]. Second, more sophisticated analysis techniques have revealed richer structures in V1 neurons than those dictated by standard models [118, 13]. As an additional yet novel demonstration of this gap, using large-scale calcium imaging techniques, we [84, 134] have recently discovered that a large percentage of neurons in the superficial layers of V1 of awake macaque monkeys respond strongly to highly specific complex features; this finding suggests that some V1 neurons act as complex pattern detectors rather than Gabor-based edge detectors as dictated by classical studies [63, 28].

While our previous work [134] has shown the existence of complex pattern detector neurons in V1, a quantitative understanding of the relationship between input stimuli and

neural responses for those neurons has been lacking. One way to better understand these neurons quantitatively is to build computational models that predict their responses given input stimuli [149]. If we can find a model that accurately predicts neural responses to (testing) stimuli not used during training, a careful analysis of that model should give us insights into the computational mechanisms of the modeled neuron(s). For example, we can directly examine different components of the model [95, 94, 111], find stimuli that maximize the model output [71, 98], and decompose model parameters into simpler, interpretable parts [115, 104].

A large number of methods have been applied to model V1 neural responses, such as ordinary least squares [136, 27], spike-triggered average [136], spike-triggered covariance [137, 118], generalized linear models (GLMs) [67, 107], nested GLMs [94], subunit models [143], and artificial neural networks [111]. Compared to more classical methods, convolutional neural networks (CNNs) have recently been found to be more effective for modeling retinal neurons [71] and V1 neurons in two studies concurrent to ours [95, 11]. In addition, CNNs have been used for explaining inferotemporal cortex and some other areas [150, 79, 151]. Nevertheless, existing studies mostly treat the CNN as a black box without analyzing much the reasons underlying its success relative to other models, and we are trying to fill that knowledge gap explicitly in this study.

To understand the CNN's success better, we first evaluated the performance of CNN models, Gabor-based standard models for simple and complex cells, and various variants of GLMs on modeling V1 neurons of awake macaque monkeys in response to a large set of complex pattern stimuli [134]. We found that CNN models outperformed all the other models, especially for neurons that acted more like complex pattern detectors than Gabor-based edge detectors. We then systematically explored different variants of CNN models in terms of their nonlinear structural components, and found that thresholding nonlinearity and max pooling, especially the former, were important for the CNN's performance. We also found that convolution (spatially shifted filters with shared weights) in the CNN was effective for increasing model performance. Finally, we used a pre-trained deep CNN [126] to model our neurons via transfer learning [11], and found that the deep CNN's higher layers, which encode more complex patterns, outperformed lower ones; the result was consistent with our earlier work [134] on the complexity of V1 neural code. While some of our observations have been stated in alternative forms in the literature, we believe that this is the first study that systematically evaluates the relative merits of different CNN components in the context of V1 neuron modeling.

2.2.2 Stimuli and neural recordings

2.2.2.1 Stimuli

Using two-photon calcium imaging techniques, we collected neural population data in response to a large set of complex artificial "pattern" stimuli. The "pattern" stimulus set contains 9500 binary (black and white) images of about 90 px by 90 px from five major categories: orientation stimuli (OT; bars and gratings), curvature stimuli (CV; curves, solid disks, and concentric rings), corner stimuli (CN; line or solid corners), cross stimuli (CX; lines crossing one another), and composition stimuli (CO; patterns created by combining multiple elements from the first four categories). The last four categories are also collectively called non-orientation stimuli (nonOT). See Figure 2.7 for some example stimuli. In this study, the central 40 px by 40 px parts of the stimuli were used as model input as 40 pixels translated to 1.33 degrees in visual angle for our experiments and all recorded neurons had classical receptive fields of diameters well below one degree in visual angle around the stimulus center [134]. The cropped stimuli were further downsampled to 20 px by 20 px for computational efficiency. Later, we use $\boldsymbol{x}_t$ to represent the t-th stimulus as a 20 by 20 matrix, with 0 for background and 1 for foreground (there can be intermediate values due to downsampling), and $\vec{\boldsymbol{x}}_t$ to denote the vectorized version of $\boldsymbol{x}_t$ as a 400-dimensional vector.

2.2.2.2 Neural recordings

The neural data were collected from V1 superficial layers 2 and 3 of two macaque monkeys A and B. For monkey A, responses of 1142 neurons in response to all 9500 (1600 OT and 7900 nonOT) stimuli were collected. For monkey B, responses of 979 neurons in response to a subset of 4605 (800 OT and 3805 nonOT) stimuli were collected due to time constraints. Each stimulus was presented for 5 repetitions for both monkeys. During each repetition, all recorded neurons' responses in terms of $\Delta F/F$ were collected. Later, we use $r^n_{t,i}$ to denote the neural response of the n-th neuron for the t-th stimulus in the i-th trial ($i = 1,\ldots,5$), r^n_t to denote the average neural response over trials, and $\vec{\boldsymbol{r}}^n$ to denote all the average neural responses for this neuron as a vector. Specifically, we have $n = 1,\ldots,1142$, $t = 1,\ldots,9500$ for monkey A and $n = 1,\ldots,979$, $t = 1,\ldots,4605$ for monkey B.

Cell classification The recorded neurons in the neural data had mixed tuning properties [134]: some acted more like complex pattern detectors, some acted more like simple oriented edge detectors, and some had weak responses to all the presented stimuli. To allow cleaner and more interpretable model comparisons, we evaluated model performance for different types of neurons separately (Section 2.2.5). For example, when comparing a CNN model

and a GLM, we computed their performance metrics over neurons that were like complex pattern detectors and those more like simple edge detectors separately, as it is possible that neurons of different types are modeled best by different model classes. To make such per-neuron-type comparison possible, a classification of neurons is required. Here we use the neuron classification scheme in Tang et al. [134]. First, neurons whose maximum mean responses were not above 0.5 ($\max r_t^n \leq 0.5$) were discarded as their responses were too weak and might be unreliable; then, among all the remaining neurons that passed the reliability test, neurons whose maximum mean responses over nonOT stimuli were more than twice of those over OT stimuli ($\frac{\max r_{t_1}^n}{\max r_{t_2}^n} > 2$, where t_1 and t_2 go over all nonOT and OT stimuli respectively) were classified as HO (higher-order) neurons and the others were classified (conservatively) as OT neurons; finally, all the HO and OT neurons were further classified into subtypes, such as curvature neurons and corner neurons, based on ratio tests similar to the one above—for example, an HO neuron was additionally considered as a curvature neuron if its maximum response over curvature stimuli was more than twice of that over non-curvature stimuli. Overall, ignoring the unreliable ones, at the top level, there were OT neurons and HO neurons; OT neurons were further classified as classical and end-stopping (neurons that responded well to short bars but poorly to long ones) neurons; HO neurons were further classified as curvature, corner, cross, composition, and mixed (neurons that failed ratio tests for all the four types of nonOT stimuli) neurons. Figure 2.12 shows example neurons of different classes.

2.2.3 Methods

Here, we describe three classes of models for modeling V1 neurons in our data set. All the models explored in this study can be considered variants of one-hidden-layer neural networks with different constraints and components. By considering them in the framework of one-hidden-layer neural networks (Section 2.2.3.4), we can easily identify key components that make CNNs perform better. In addition, all the methods here model each neuron separately (no parameter sharing among models fitted to different neurons) and the numbers of parameters of different models are kept to be roughly the same if possible; the parameter separation and equality in model size ensure a fairer comparison among models. For each neuron n from some monkey, all our models take image $\boldsymbol{x}_t$ of size 20 by 20 as input and try to predict the neuron's mean response r_t^n of image t as output. See Section 2.2.2 for an explanation of the notation.

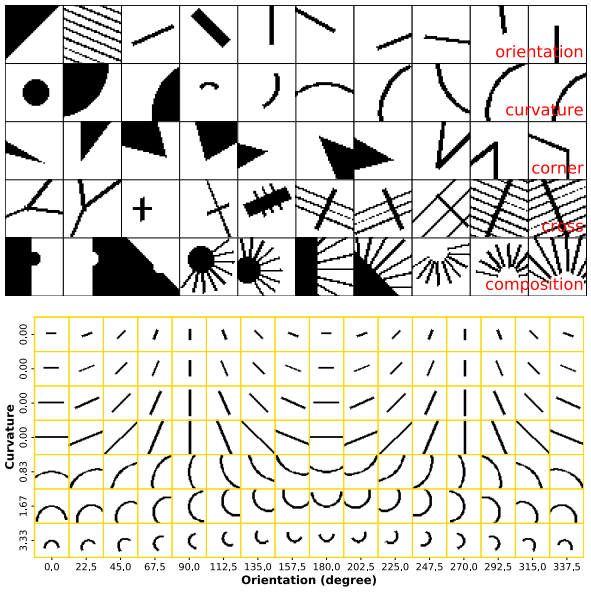


Fig. 2.7 **Top** "Pattern" stimulus set. Stimuli are arranged in rows, each row showing 10 randomly drawn stimuli for each of the five categories (see the bottom right corner of each row). Only the central 40 px by 40 px parts of stimuli are shown. Refer to Tang et al. [134] for details. **Bottom** A subset of curvature and line stimuli in the stimulus set, ordered by stimulus parameters (curvature, length, and orientation). Only the central 40 px by 40 px parts are shown.

2.2.3.1 CNN models

A CNN model passes the input image through a series of linear-nonlinear (LN) operations—each of which consists of convolution, ReLU nonlinearity [80], and (optionally) max pooling. Finally, outputs of the final LN operation are linearly combined as the predicted response of the neuron being modeled. Our baseline CNN model for V1 neurons is shown in Figure 2.8, with one (convolutional) layer and 9 filters. Given a 20 by 20 input, it first convolves and rectifies ("convolve + threshold" in the figure) the input with 9 filters of size 9, yielding 9 feature maps (channels) of size 12 by 12, one for each filter. Then max pooling operation ("max pool" in the figure) is performed for each feature map separately to produce 9 pooled feature maps of size 4 by 4. Finally, all the individual output units across all the pooled feature maps are linearly combined ("linear combination" in the figure), plus some bias, to generate the predicted neural response.

As shown in Table 2.2 of Section 2.2.4.1, apart from the baseline model with 9 channels (`B.9` in the table), we also explored other CNN models with the same overall architecture but different numbers of channels.

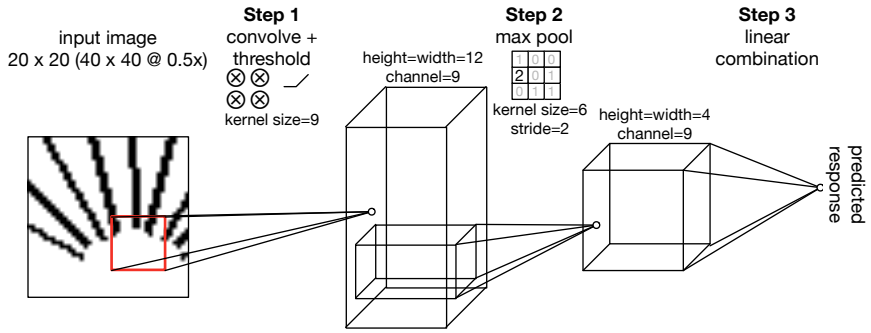


Fig. 2.8 The architecture of our baseline CNN model (or `B.9` in Table 2.2). Given a 20 by 20 input image (40 by 40 downsampled to half; see Section 2.2.2), the model computes the predicted neural response in three steps. In **Step 1** ("convolve + threshold"), the model convolves and rectifies the input to generate an intermediate output of size 12 by 12 by 9 (height, width, channel; 3D block in the middle); concretely, for each of the model's 9 filters of size 9 by 9 (kernel size), the model computes the dot product (with some bias) between the filter and every 9 by 9 subregion in the input (red square being one example), rectifies ($x \mapsto \max(0,x)$) all the dot products, and arranges the rectified results as a 12 by 12 feature map; the process is repeated for each of the 9 filters (channels) and all the 9 feature maps are stacked to generate the 12 by 12 by 9 intermediate output. In **Step 2** ("max pool"), max pooling operation is performed for each feature map separately to produce 9 pooled feature maps of size 4 by 4; concretely, for each of the 12 feature maps obtained in Step 1, maximum values over 6 by 6 subregions are computed every 2 data points (stride) and arranged as a 4 by 4 pooled feature map; the process is repeated for each of the 9 feature maps to generate the 4 by 4 by 9 pooled output. In **Step 3** ("linear combination"), all the individual output units across all the pooled feature maps are linearly combined plus some bias to generate the predicted neural response. See Section 2.2.3.1 as well.

2.2.3.2 "Standard" Gabor-based models

Gabor filters are widely used in theoretical models of V1 neurons [28, 63, 26]. Therefore, we tried to fit (relatively speaking) standard Gabor-based V1 models to our data as control. We tried Gabor simple cell models, Gabor complex cell models, as well as their linear combinations (Figure 2.9). Interestingly, to the best of our knowledge, such models were not examined in the existing V1 data fitting literature in terms of their performance compared to more popular ones such as GLMs. Check Section 3.2 of Zhang et al. [157] for details.

2.2.3.3 Generalized linear models

We consider the following set of Poisson generalized linear models [93, 103] with possibly nonlinear input transformations: vanilla GLMs, Fourier power models [27], and generalized quadratic models [105, 104] (Figure 2.10). We also tried Gaussian GLMs and they performed

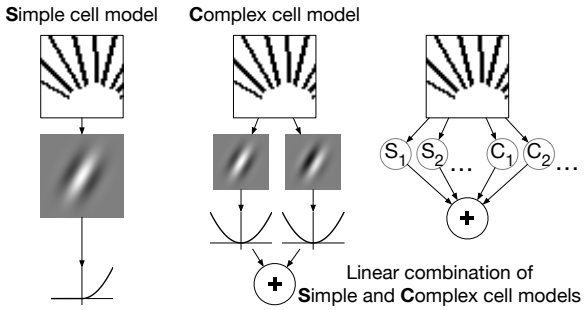


Fig. 2.9 The architecture of Gabor-based models. A simple cell model (left) takes the dot product of a Gabor filter and the input, and passes through the output through a half-wave squaring nonlinearity. A complex cell model (middle) takes the dot products of two Gabor filters with quadrature phase relationship, squares and sums the outputs. A linear combination of simple and complex cell models (right) takes some linear combination of some simple cell models (S) and some complex cell models (C). This figure is partially inspired by Figure 1 in Rust et al. [118] and Figure 1 in Carandini et al. [13].

consistently worse than Poisson ones in our experiments. Note that the term "GLM" has been used pretty loosely in the literature, and many models with similar structural components to those in the CNN are considered GLMs by many. We want to emphasize that the purpose of including GLMs in this study is not to compare CNNs and (all the variations of) GLMs in terms of performance but to find key components that make CNN models outperform commonly used models for V1 modeling. We call these models GLMs mainly because they are often formulated as GLMs in the literature. See Section 2.2.3.4 for the connection between CNNs and GLMs considered in this study. Check Section 3.3 of Zhang et al. [157] for details.

2.2.3.4 Connections among CNNs, Gabor models, and GLMs

As mentioned in the beginning of Section 2.2.3, the three classes of models considered in this study are connected and form a continuum as they all can be roughly formulated as vanilla one-hidden-layer neural networks [9], or one-hidden-layer multilayer perceptrons (MLPs):

$$\hat{r}(\vec{\boldsymbol{x}}) = \sum_{i=1}^{C} c_i z_i(\vec{\boldsymbol{x}}) + b, \tag{2.3a}$$

$$z_i(\vec{\boldsymbol{x}}) = f(a_i(\vec{\boldsymbol{x}})), \tag{2.3b}$$

$$a_i(\vec{\boldsymbol{x}}) = \langle \vec{\boldsymbol{x}}, \vec{\boldsymbol{w}}^{(i)} \rangle + b_i. \tag{2.3c}$$

A one-hidden-layer neural network computes the output $\hat{r}$ given (vectorized) input stimulus $\vec{\boldsymbol{x}}$ according to Eqs. (2.3). Overall, the output is a linear combination of C hidden

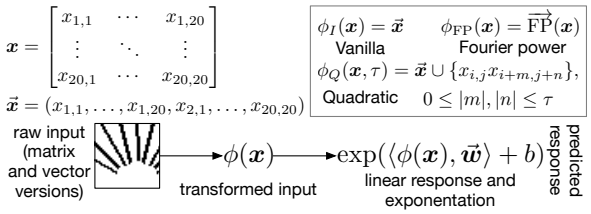


Fig. 2.10 The architecture of generalized linear models. The raw input stimulus $\boldsymbol{x}$ is first transformed into $\phi(\boldsymbol{x})$, where different $\phi(\cdot)$ are used for different GLM variants (inside the box). For vanilla GLMs, we use the identity transformation $\phi_I(\cdot)$; for Fourier power models, we use the Fourier power transformation $\phi_{\mathrm{FP}}(\cdot)$ (Section 3.3.2 of Zhang et al. [157]); for generalized quadratic models, we use the localized quadratic transformation $\phi_Q(\cdot, \tau)$ (Sections 4.3.2 and 3.3.3 of Zhang et al. [157]). The transformed input $\phi(\boldsymbol{x})$ is passed into a linear function $\langle \cdot, \vec{\boldsymbol{w}} \rangle + b$ and the output is exponentiated to give the predicted neural response. For details on the localized quadratic transformation ($\phi_Q(\cdot, \tau)$ in the figure), see Section 4.3.2 and Figure 5 of Zhang et al. [157].

units' output values z_i as shown in Eq. (2.3a). Each hidden unit's output is computed by applying some nonlinearity (also called activation function) f on the pre-activation value of the hidden unit a_i as shown in Eq. (2.3b), and pre-activation value a_i is a linear function of input specified by weights $\vec{\boldsymbol{w}}^{(i)}$ and bias b_i as shown in Eq. (2.3c).

Gabor models can be formulated as MLPs with constraints that weights $\vec{\boldsymbol{w}}^{(i)}$ must be Gabor functions. A simple cell model is a MLP with one hidden unit and half-wave squaring nonlinearity; a complex cell model is a MLP with two hidden units in quadrature phase relationship and squaring nonlinearity; a linear combination of simple and complex cell models is a MLP with multiple hidden units and mixed nonlinearities.

GLMs can be formulated as MLPs with an additional exponential nonlinearity on output. A vanilla GLM is a MLP with one hidden unit and no nonlinearity (linear); a Fourier power GLM is a MLP with multiple hidden units of fixed weights (Fourier basis functions) and squaring nonlinearity; A GQM is a MLP with multiple hidden units and squaring nonlinearity—the linear term in Eq. (5b) of Zhang et al. [157] can be absorbed into the quadratic one as long as the quadraic coefficient matrix is full rank. Empirically, we found the additional accelerating exponential nonlinearity to be unimportant for the modeling of our data, as Poisson GLMs with the additional accelerating nonlinearity performed similarly or marginally better, compared to Gaussian and softplus GLMs without such nonlinearity (Supplementary Materials of Zhang et al. [157]).

A CNN can be formulated as a MLP with ReLU ($x \mapsto \max(0, x)$) nonlinearity and an additional max pooling operation before the final output computation of Eq. (2.3b). Compared to other models, a CNN has additional constraints among the weights of hidden units—shared and spatially shifted in groups. For example, our baseline CNN can be considered as a MLP

with $12 \times 12 \times 9 = 1296$ hidden units, as each 9 by 9 filter in the CNN yields a feature map of $12 \times 12 = 144$ hidden units, and there are 9 filters in the CNN. For MLP hidden units derived from a common feature map, filter weights are shared and spatially shifted; for MLP hidden units derived from different feature maps, filter weights are independent. This group-wise sharing of hidden unit weights in CNN models is not present in GLMs, which we will compare in detail with CNNs in Section 2.2.5 as GLMs were the best-performing non-CNN models in our experiments.

Table 2.1 gives a summary of different models in terms of their structures, under the framework of one-hidden-layer neural network (or MLP). We classify nonlinearities into thresholding (half-wave squaring and ReLU) and non-thresholding (squaring) ones, because we found all the thresholding activation functions behaved essentially the same in our experiments (Section 2.2.5.2) and we think that being thresholding or not may be the most important aspect for a nonlinearity.

Table 2.1 Comparison of model structures for Gabor models, GLMs, and CNNs in the framework of one-hidden-layer MLP. First two columns specify the model class and subclass. The third column shows whether the models' corresponding MLPs have multiple hidden units or not. The fourth column shows the constraints among hidden units imposed by the models; "independent" means weights for different hidden units can vary independently, "shared" means weights for different hidden units are tied together (via convolution), "quadrature phase" means weights of the hidden unit pair are in quadrature phase relationship (specific to Gabor models), and "fixed" means weights are not learned but specified before training. The fifth column specifies the nonlinearity (activation function), with "none" meaning no nonlinearity (identity or linear activation function), and "mixed" meaning both thresholding and non-thresholding nonlinearities. The last column specifies additional structures imposed by the models.

Class	Subclass	Multiple units	constraints among units	nonlinearity	additional structures
Gabor	simple	No	—	thresholding	weights are Gabor
	complex	Yes	quadrature phase	non-thresholding	weights are Gabor
	combination	Yes	independent	mixed	weights are Gabor
GLM	vanilla	No	—	none	exponential output
	Fourier power	Yes	fixed (not learned)	non-thresholding	exponential output
	GQM	Yes	independent	non-thresholding	exponential output
CNN	—	Yes	independent + shared	thresholding	max pooling

2.2.4 Implementation Details

Here I just include the implementation details of CNN models for brevity; check Section 4 of Zhang et al. [157].

2.2.4.1 CNN models

Detailed model architecture Table 2.2 shows all the three CNN model architectures we evaluated against other models (Section 2.2.5), with the baseline CNN model (Figure 2.8) denoted `B.9` in the table. For a fair comparison between CNNs and other models (primarily GLMs; Gabor models inherently have too few parameters), in addition to the baseline CNN model `B.9`, we also evaluated two variants of the baseline model by changing its number of channels. Overall, the three CNN models match the three classes of GLMs (Section 4.3 of Zhang et al. [157]) in terms of model size.

Table 2.2 CNN model architectures explored in this work. Each row describes one CNN model architecture, with the first column showing its name (`B.n` where `n` is the number of channels), middle columns describing its computational components, and the last showing its number of parameters. Each CNN model first passes the input image through three computational components shown in the table—convolution (conv), nonlinearity, and pooling—and then linearly combine ("fully connected" in CNN jargon) output values of the pooling operation to give the model output. The baseline CNN (`B.9`) has its number of parameters shown in boldface. The number of parameters is computed by adding the number of parameters in the convolutional layer and that in the fully connected layer. For example, the baseline model `B.9` has $9 \times (9 \times 9 + 1) = 738$ parameters (9 for number of channels, 9 for kernel size, and 1 for bias) for the convolutional layer, and $9 \times 4 \times 4 + 1 = 145$ parameters (9 for number of channels, 4 for pooled feature map's size, and 1 for bias) for the fully connected layer, resulting in $738 + 145 = 883$ parameters.

Name	conv	nonlinearity	pooling	# of params
`B.2`	(kernel 9, channel `n`)	ReLU	(max pool, kernel 6, stride 2)	197
`B.4`				393
`B.9`				**883**

2.2.5 Results

2.2.5.1 CNN models outperformed others especially for higher-order neurons

Figure 2.11 shows the performance of CNN models vs. others on explaining our V1 neural data. Because the full stimulus set consists of different types of stimuli (OT, CN, CV, etc.; see Section 2.2.2.1), and the full population of neurons for each monkey consists of two subsets (OT neurons and HO neurons, which can be divided into finer subsets as well; see Section 2.2.2.2) that responded very differently to different types of stimuli, we trained all models using different stimulus subsets ("OT" stimuli and all stimuli; we also tried training only on "nonOT" stimuli, and that gave similar results to using all stimuli), and evaluated

each model in terms of its average CC^2_{norm} (Section 3.6 of Zhang et al. [157]) averaged over OT neurons and HO neurons (for results on finer subsets, see Section 2.2.5.2 and later). We do not show results of HO neurons trained on OT stimuli, as HO neurons by definition did not respond to OT stimuli well and the results might be unreliable.

We compare CNN models and other models at two different levels. At the individual model architecture level (solid bars in Figure 2.11), we compare specific CNN architectures (models with different numbers of channels) with Gabor models and GLMs. In this case, CNN models with more channels worked better and they outperformed their GLM counterparts (`B.2` vs. Fourier power GLMs, `B.4` vs. linear GLMs, and `B.9` vs. GQMs; see Section 2.2.4.1) across the board; GQMs had in general better performance than other GLMs, but still fell behind CNNs by a large margin. Gabor models performed similarly to GLMs or worse, and were outperformed by CNNs as well.

At the overall model category level (dashed lines in Figure 2.11), we compare CNN models as a whole to Gabor models as a whole as well as GLMs as a whole. To do this, for each model category, we constructed an "all" model for that category by choosing the best performing model architecture (in terms of performance on validation data for CNNs and GLMs, and in terms of performance on training data for Gabor models; testing data was never used during the model selection) for each individual neuron. By comparing the dashed lines, we have the following empirical observations about the three model classes.

CNNs outperformed other models especially for HO neurons with complex stimuli When stimuli were the same, the relative performance gap between CNN and other models was larger for HO neurons than OT neurons (middle and right columns of panels of Figure 2.11). For example, on Monkey A, the relative performance increase of the CNN over the GLM increased from 34.2 % for OT neurons to 52.2 % for HO neurons. When neurons to model were the same, the relative performance gap was larger for complex stimuli than simple stimuli (left and middle columns of panels of Figure 2.11). For example, on Monkey A, the relative performance increase of the CNN over the Gabor model increased from 27.3 % for "OT" stimuli to 48.5 % for all stimuli.

Priors on Gabor models helped especially with limited data When the stimuli were limited and simple, Gabor models outperformed GLMs, possibly due to the strong and neurophysiologically reasonable prior on Gabor models that filter weights can be described well by Gabor functions [63], and vice versa when the stimuli were relatively sufficient and rich (leftmost column of panels vs. other panels of Figure 2.11). One may hypothesize that multi-component Gabor models (`multi` ones) outperformed standard ones (`complex`

and `simple`) mostly due to having multiple orientations; this was not true as shown in Section 2.2.5.3.

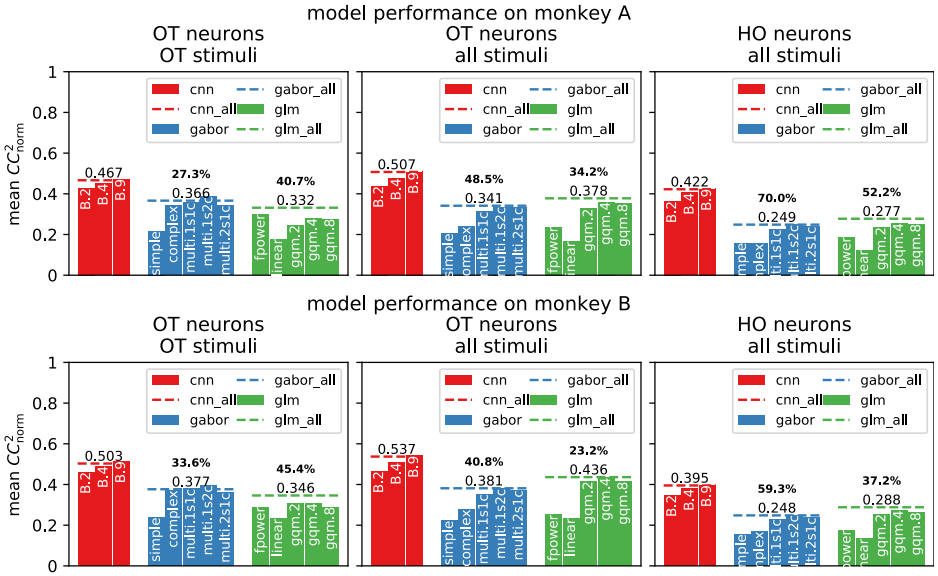


Fig. 2.11 CNN models vs. others on explaining V1 neural data. Two rows of panels show results for monkey A and monkey B respectively, and three columns of panels show how models performed on different neuron subsets ("OT" and "HO"), evaluated on different subsets of stimuli ("OT" and "all"). For each panel, the model performance is shown in CC^2_{norm} averaged over neurons in the neuron subset. For each category of models (`cnn`, `glm`, etc.), solid bars show model performance of different specific model architectures, and dashed lines (suffixed by `_all`) show the category's overall "best" performance by taking the best model architecture for each individual neuron (in terms of validation performance for CNNs and GLMs and training performance for Gabor models). Boldface numbers are the relative performance increases of the CNN classes over non-CNN classes (computed as ratios between dashed lines minus one). For CNN models (red), check Table 2.2 for their meanings. For Gabor models (blue), `complex` and `simple` mean complex cell and simple cell models; `multi.MsNc` means linear combinations of `M` simple and `N` complex model(s). For generalized linear models (green), `linear` means vanilla GLM; `fpower` means Fourier power GLM; `gqm.x` (`x` being one of `0`,`2`,`4`,`8`) means the quadratic GLM with locality `x`.

Finally, Figure 2.12 shows the fitting results of some neurons in different classes (see Section 2.2.2.2); for CNN models, we also show the learned filters and visualization results obtained by activation maximization [98]; these visualization results are images that activate fitted CNNs most. In most cases, Gabor models and GLMs failed to predict the high-responding parts of the tuning curves compared to CNNs.

In Supplementary Materials, we show that CNN models outperformed others even with less amount of data; we also show additional results on CNN models, such as comparison

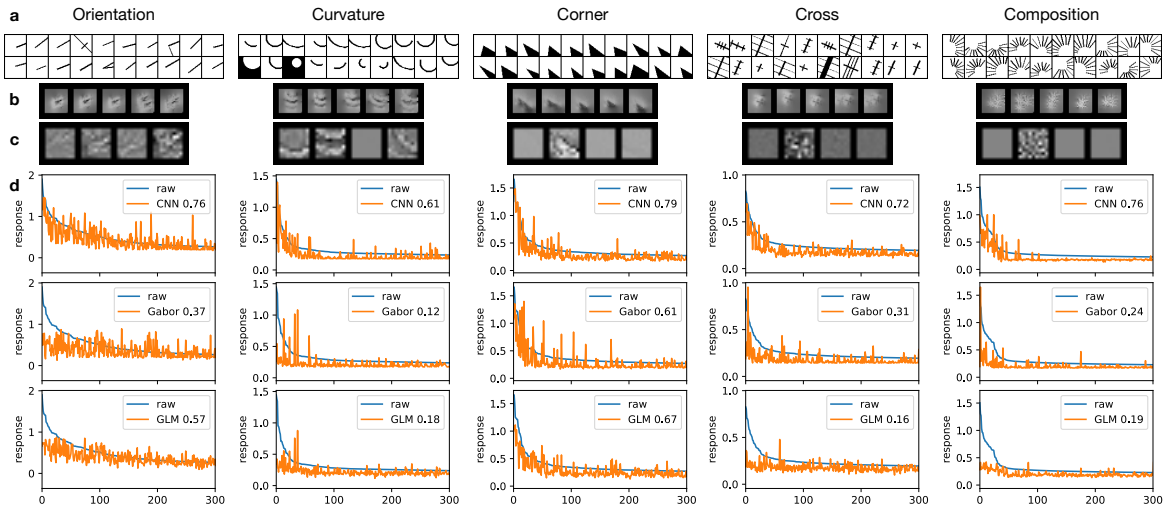


Fig. 2.12 Example neurons and their fitting results. For each of the five stimulus classes shown in different columns, we show the following four pieces of information regarding the fitting of a neuron that responded better to this class than the others (**a-d**). **a** The top 20 responding stimuli of the neuron; **b** the fitted CNN fully connected output layer's visualization results (over 5 random initalizations) obtained by activation maximization [98] implemented in `keras-vis` [78]; **c** the fitted CNN's four 9 by 9 convolutional filters (each scaled by the sum of squares of its associated weights in the fully connected layer); **d** the neuron's fitting results (over testing data) on three categories of models: CNN, Gabor and GLM, with model performance in terms of CC^2_{norm} given in the legends. As each category of models has multiple variants or architectures, we roughly speaking picked the overall best one for each category. We picked the 4-channel architecture `B.4` for CNN, as it performed almost the same as the baseline `B.9` (Figure 2.11) and allows easier visualization and interpretation; we picked `multi.1s2c` for Gabor, and `gqm.4` for GLM as they performed overall better than other variants. Check Figure 2.11 for the meanings of model names.

of different optimization configurations and comparison of different architectures (different numbers of layers, different kernel sizes, and so on). We will focus on the one-convolutional-layer CNN model `B.9` with 883 parameters for the rest of this study, because its performance was close to the best among all the CNN models we tried (Supplementary Materials) without having too many parameters, and its one-layer architecture is easier to analyze than those of similarly performing models.

2.2.5.2 What made CNNs outperform other models

As shown in Figure 2.11, the baseline CNN architecture alone (`B.9`) outperformed GLMs, which were the best non-CNN models in this study, by a large amount, especially for HO neurons. By comparing the row for the CNN and the rows for GLMs (particular

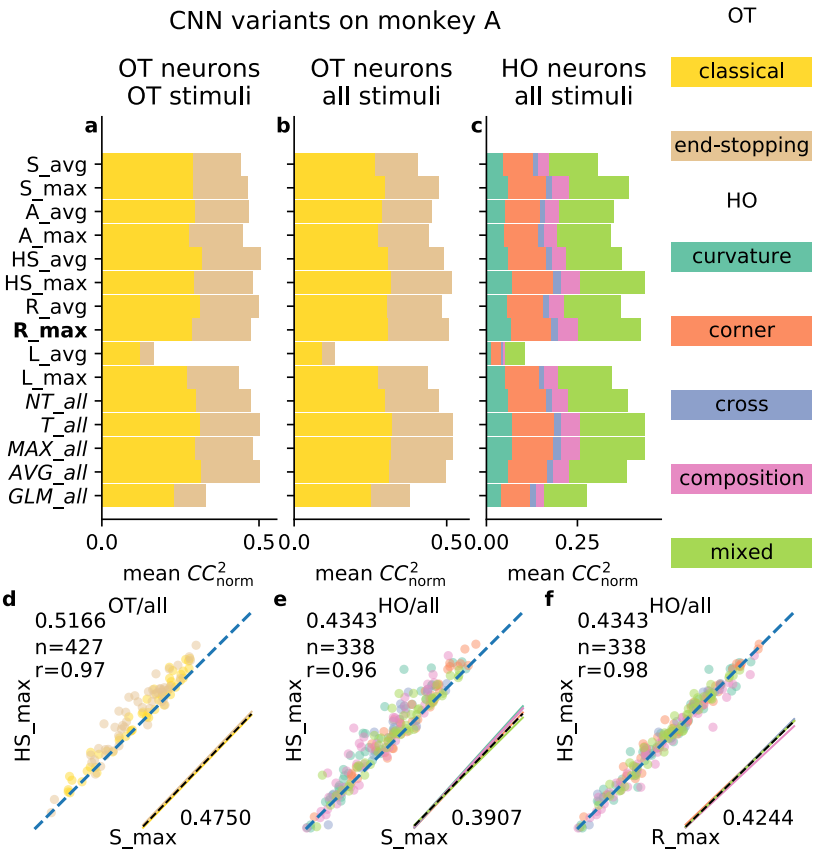


Fig. 2.13 Detailed comparison of CNN variants, monkey A. **a-c** ten variants of the baseline CNN (`B.9`), along with the "all" model for GLMs `GLM_all` (Figure 2.11) for reference. In addition, four "all" CNNs, each of which constructed from CNN models with some shared structural component (thresholding nonlinearity `T`, non-thresholding nonlinearity `NT`, max pooling `MAX`, or average pooling `AVG`), are shown as well. CNN variants are named `X_Y` where `X` and `Y` denote nonlinearity and pooling type, respectively (Section 2.2.5.2). The organization of panels is the same as that in Figure 2.11, except that only results for Monkey A are shown (see Figure 2.14 for Monkey B). Rows show different models, whose performance metrics (mean CC^2_{norm}) are decomposed into components of neuron subclasses, denoted by different colors (legend on the right). For each model in some panel, the length of each colored bar is equal to the average performance over that neuron subclass multiplied by the percentage of neurons in that subclass, and the length of all bars concatenated is equal to the average performance over all neurons. The baseline model has its name in bold, and "all" models in italics. **d,e** Neuron-by-neuron comparison of the a CNN variant with thresholding nonlinearity (`HS_max`) vs. one without (`S_max`) for OT neurons, all stimuli (**d**) and HO neurons, all stimuli (**e**). For **d**, **e**, and **f**, performance metrics (mean CC^2_{norm}) are shown at corners, Pearson correlation coefficients between models are shown at the top left, and regression lines for different neuron subclasses (colored solid) together with the regression line over all neurons (black dashed) are shown at the bottom right (scaled and shifted to the corner for clarity; otherwise these regression lines will clutter the dots that represent individual neurons). **f** Comparison of two thresholding nonlinearities, for HO neurons, all stimuli. Results with max pooling are shown, and average pooling gave similar results.

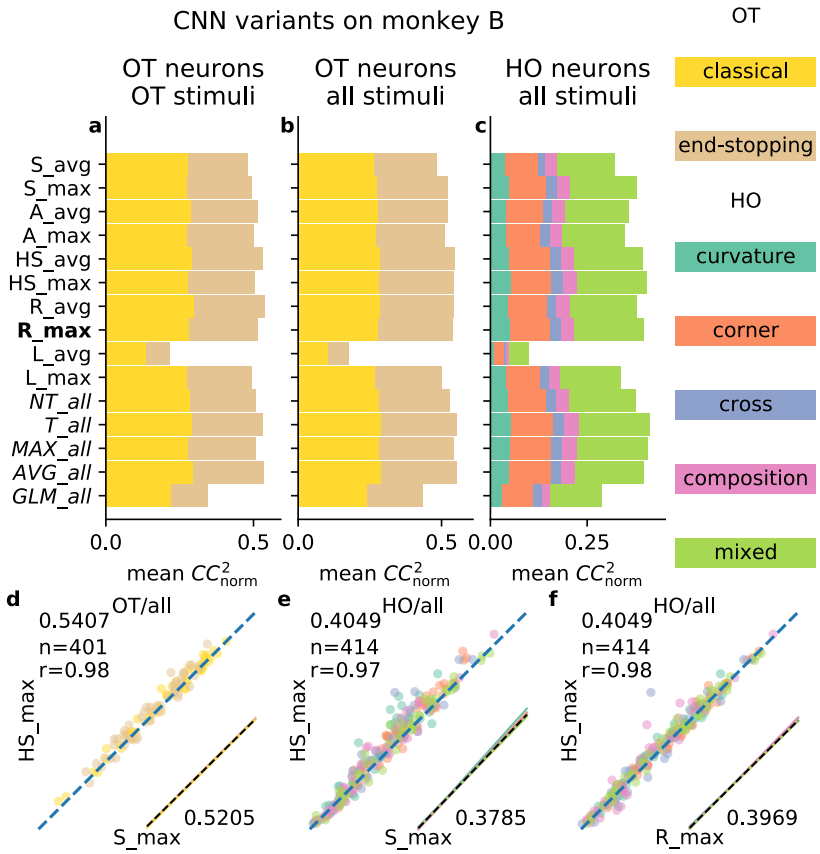


Fig. 2.14 Detailed comparison of CNN variants, monkey B. Check Figure 2.13.

the row for the GQM, as GQMs overall performed better than other GLM variants) in Table 2.1 (Section 2.2.3.4), we hypothesize that this performance gap was primarily due to the structural components present in the CNN but not in GLMs we studied: thresholding nonlinearity (ReLU), max pooling, and shared weights of hidden units (convolution). To test our hypothesis, we explored different variants of our baseline CNN architecture `B.9` in terms of its structural components. The results on thresholding nonlinearity and max pooling are given in this part, and those on convolution are given in the next part. While our GLMs possess an exponentiation nonlinearity which is not present in our CNNs, we found that the exponentiation gave little performance increase than without (Supplementary Materials).

To better understand the utilities of thresholding nonlinearity and max pooling, we explored various variants of the baseline CNN architecture in terms of nonlinearity and pooling scheme. Specifically, we tried all combinations of five different nonlinearities—ReLU (`R`), ReLU followed by squaring (half-squaring, `HS`), squaring (`S`), absolute value (`A`), linear (no nonlinearity, `L`)—and two different pooling schemes—max pooling (`max`), average (mean) pooling (`avg`)—with other structural components unchanged. Thus, we obtained ten different CNN variants (including the original one) and compared them with the "all" model for GLMs (picking the best model architecture for each neuron), or `GLM_all` as reference. Results are shown in Figure 2.13 and Figure 2.14, which have the same organization: panels a-c show the performance of all explored models as before, but with CC^2_{norm} over OT and HO neurons decomposed into average CC^2_{norm} for finer subsets inside OT and HO neurons (Section 2.2.2.2) to examine model performance in more detail; panels d-f show the neuron-by-neuron comparison of different pairs of models for highlighting. Overall, we have the following observations (letters in the parentheses denote the panels used for highlighting among d-f, if any).

- Thresholding nonlinearities outperformed non-thresholding ones (d,e).
- Thresholding nonlinearities performed similarly (f).
- No consistently better pooling type, but max pooling was more powerful in isolation.
- High correlation between per-neuron and average model performance (almost all panels).

Thresholding nonlinearities outperformed non-thresholding ones Compared to GLMs we explored in this work, one nonlinear structural component unique to CNNs is ReLU, a thresholding nonlinearity. To understand the usefulness of thresholding nonlinearities in general, we compared four CNN variants with thresholding nonlinearities (`R_max`, `R_avg`,

HS_max, HS_avg) with four without (A_max, A_avg, S_max, S_avg) and found that thresholding (R, HS) in general helped. This can be seen at two levels. At the level of individual architectures, those with thresholding generally performed better than those without (d, e, and rows 5-8 from the top vs. 1-4 in a-c of Figures 2.13 and 2.14). At the level of model categories, we combined all four thresholding models into one "all" model (T_all) and all four non-thresholding ones as well (NT_all), using the same method as we constructed "all" models in Figure 2.11; we found that thresholding helped as well. Our results suggest that the recorded V1 neurons actually take some thresholded versions of the raw input stimuli as their own inputs. There are at least two ways to implement this input thresholding. First, neurons may have some other upstream neurons as their inputs, each upstream neuron with its own thresholding nonlinearity as modeled in McFarland et al. [94], Vintch et al. [143]. Second, the thresholding may happen at the dendritic tree level, as suggested by Gollisch and Meister [46].

Thresholding nonlinearities performed similarly While the two thresholding nonlinearities (R and HS) showed better performance overall, we did not see much difference between the two (f, and HS_max vs. R_max, HS_avg vs. R_avg in a-c of Figures 2.13 and 2.14). This observation was consistent with Heeger [52], where the author claimed that these two types of thresholding nonlinearities are both consistent with physiological data and the brain might be using one as an approximation to implement the other.

No consistently better pooling type, but max pooling was more powerful in isolation While thresholding nonlinearities showed better performance consistently than non-thresholding ones as shown above, the results were mixed for two pooling schemes and depended on nonlinearities, combinations of neurons and stimuli, and monkeys (rows 1-8 from the top, as well as MAX_all vs. AVG_all that were constructed like T_all and NT_all above, in a-c of Figures 2.13 and 2.14). We suspect such mixed results were due to the complicated interaction between nonlinearity and pooling. In other words, the contributions of nonlinearity and pooling to model performance do not add linearly. Still, we think max pooling is a powerful computational component per se for modeling neural responses, as max pooling alone without any nonlinearity performed comparably with many other models with pooling and nonlinearity (L_max vs. others in a-c of Figures 2.13 and 2.14).

High correlation between per-neuron and average model performance Figures 2.13 and 2.14 show that different models performed differently. We found that the performance increase/decrease of one model over another one seemed to be *universal*, rather than class-

or neuron-specific. We can see this universality from several aspects when two models are compared neuron by neuron (d-f of Figures 2.13 and 2.14). First, there was a high correlation between the performance metrics of individual neurons (high Pearson correlation coefficients r). Second, we performed linear regression on each neuron subclass as well as on all neurons (colored solid lines and black dashed line in the lower right corner of each panel), and found all regression lines were very close.

2.2.5.3 Convolution was more effective than diverse filters

Apart from thresholding nonlinearity and max pooling explored in Section 2.2.5.2, CNN models have another unique structural component compared to other models in our study—shared weights among hidden units via convolution—as shown in Table 2.1. In contrast, other models with multiple hidden units (when these models are considered as MLPs; see Section 2.2.3.4) often have hidden units with independent and diverse weights without sharing ("independent" in Table 2.1). In this section, we explore the relative merits of these strategies for relating weights of different hidden units—shared weights via convolution vs. independent weights—in terms of model performance, not only for the CNN, but also for other model classes. The results are shown in Figure 2.15, with similar layout to Figures 2.13 and 2.14. We have the following observations (letters in the parentheses denote the panels used for highlighting).

- Multiple diverse filters alone did not help much (d vs. e).
- Convolution helped achieve better performance with the same number of parameters (f).

Multiple diverse filters alone did not help much To examine the impact of having multiple filters with diverse shapes, we explored two classes of models: Gabor models and CNN models. For Gabor models, we examined three single-filter variants—simple cell model (`Gabor_s`), complex cell model (`Gabor_c`), and the "single-component" Gabor model (`Gabor_single`) constructed from simple and complex cell models similarly to "all" models in Figure 2.11—and one multi-filter variant—one simple two complex (`Gabor_1s2c`; other multi-filter models performed worse as shown in Figure 2.11). For CNN models, we varied the number of channels of the baseline CNN `B.9` from 1 (`B.1`) through 18 (`B.18`).

While the multi-filter Gabor model outperformed both simple and complex cell models by a large margin (a-c,d of Figure 2.15), we found that the single-component model (`Gabor_single`), which takes the better one of simple cell and complex cell models for each neuron, worked almost as well as the multi-filter one (a-c,e of Figure 2.15). While

there was still some performance gap between `Gabor_single` and the `Gabor_1s2c`, the gap was relatively small and there was strong correlation between the two models in terms of per-neuron performance (Figure 2.15e). For each neuron, we further compared the learned filters of simple, complex, and multi-filter Gabor models, and found that in some extreme cases, the learned multi-filter Gabor model was degenerate in the sense that it had its simple component dominate its complex components or vice versa (Figure 2.15g; check the caption).

The results for one-channel CNN and the baseline 9-channel CNN are shown in the top two rows of Figure 2.15a-c, and we found that the performance increase (around 20 % to 50 %) was not proportional to the increase in the number of parameters (around 800 %, or 99 vs. 883 parameters). See Figure 2.11 and Supplementary Materials for more results on the model performance of CNN as we change the number of channels.

Convolution helped achieve better performance with the same number of parameters As shown in the previous part, having multiple independent filters of diverse shapes was not effective for increasing performance relative to the increase in model size it involved. However, we found that convolution was much more effective, achieving better model performance without increasing the number of parameters. To illustrate this, we compared the baseline CNN's average pooling (`R_avg`) variant, which linearly combines ReLU units, with a multilayer perceptron consisting of one hidden layer of 40 ReLU units (`MLP_40`). To make the two models match in the number of parameters, we performed principal component analysis to reduce the input dimensionality for the MLP to 20; therefore the MLP has $40 \times (20+1)+40+1 = 881$ parameters, roughly matching the CNN (883 parameters). The CNN outperformed the MLP by a relatively large margin (a-c,f of Figure 2.15). We also explored the trade-off between input dimensionality and number of hidden units for MLP, with the number of parameters roughly fixed (Figure 2.15h); given roughly the same number of parameters, the CNN, which has convolution, consistently outperformed MLPs of various configurations.

One may argue that the (average) pooling, which was difficult to avoid in our experiments as CNNs would otherwise have too many parameters, helped model performance as well; while such interpretation is possible, it is also helpful to simply consider convolution and pooling collectively as a modeling prior that helps neural response prediction with limited number of parameters and training data. The effectiveness of convolution and pooling could also be due to eye movements during neural data recording; as shown in our previous work [134], the eye movement was in general very small (the standard deviation of the distribution of eye positions during stimulus presentation was in general less than 0.05° in visual angle,

or 0.75 px in the 20 px by 20 px input space of the CNN) for our data, and such interpretation was less likely.

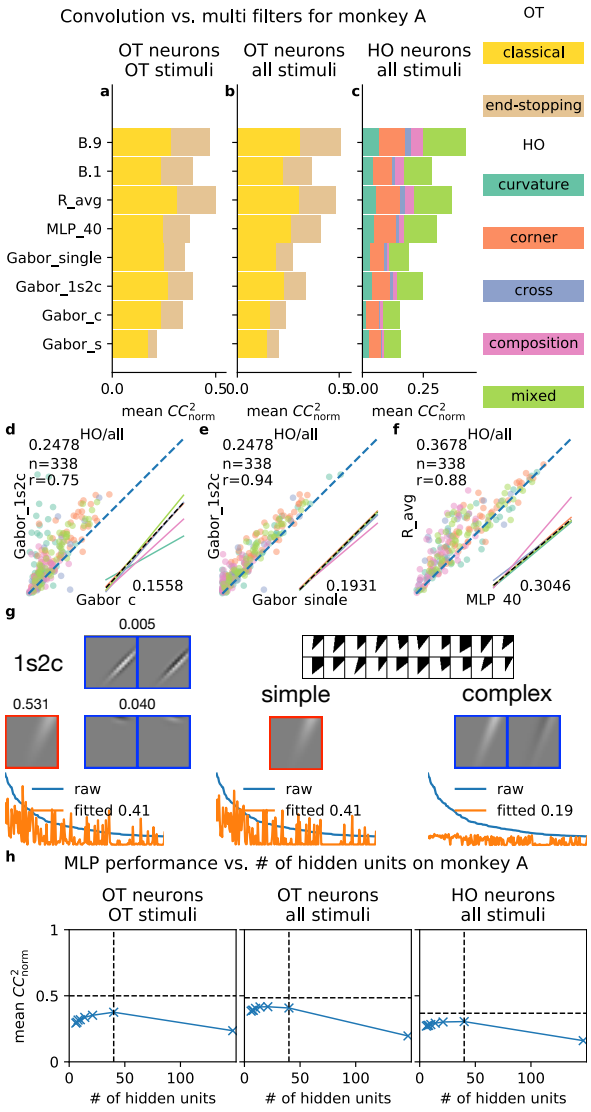


Fig. 2.15 Convolution seemed more important than diverse filters. **a-f** Comparison of single- vs. multi-component Gabor models (highlighted in **d**,**e**), comparison of single- vs. multi-channel CNN models, and comparison of models with and without convolution (highlighted in **f**). See Section 2.2.5.3 for details. These panels have similar formats to those in Figure 2.13. **g** Learned single- (`simple` and `complex`) and multi-component (`1s2c`) Gabor models fitted to a particular neuron's data. This neuron was tuned to corners as shown in the top right part of the panel. For the three models (left, middle, right), we show the learned filters (top) and fitting results (bottom). Simple cell components are shown with red borders, and complex cell components are shown with blue borders. For the multi-component model, we also show the weights of different components at the top of filters. In this case, the multi-component model was dominated by its simple component with weight 0.531, which was orders of magnitude larger than the weights of its complex components. **h** Performance vs. number of hidden units for MLP models. Vertical dashed lines denote the MLP model (`MLP_40`) in panels **a-c,f**, and horizontal dashed lines show performance metrics of the CNN `R_avg`. Only results for monkey A are shown and monkey B gave similar results.

2.3 Relationships to the proposed work

The first study [158] has shown the feasibility of learning horizontal functional connectivities among V1 neurons using the Boltzmann machine, a neural network model with local horizontal recurrent computation components. This study has demonstrated that the Boltzmann machine, which is inspired by neuroscience but often dismissed to be too abstract and different from the real brain, is in fact a useful and viable model for conceptualizing certain recurrent computations in the primary visual cortex. However, the model has many hand-crafted designs and is difficult to be trained end-to-end on V1 neural data at scale. The proposed study will try to incorporate recurrent computation components into standard CNNs with feedforward connections and learn two types of connections together in an end-to-end fashion to explain more V1 data and phenomena.

The second study [157] has established a correspondence between the (feedforward) CNN and biological reality in the context of V1 modeling by systematically evaluating and dissecting different CNN components. While this project is not directly related to recurrent circuits, it has demonstrated that predicting neural responses to natural and complex stimuli accurately is a useful objective metric for identifying neural network models with high correspondence with biological reality; in addition, it has shown the usefulness of various ablation, dissection, and visualization methods for comparing and understanding neural network models of different architectures. The metric and methods in this study have laid the foundations for developing and analyzing models with recurrent computation components in the proposed work.

Chapter 3

Existing work relevant to the proposal

This chapter is not meant to be read directly; it mostly provides background knowledge for the proposed work in Chapter 4.

3.1 Existing computational work on V1's nCRF effects

As mentioned in Section 1.1, V1's complex nonlinear response properties not predicted by standard models are collectively called *non-classical receptive field (nCRF) effects* [160] in this document. When carefully prepared artificial stimuli are used, V1's complex response properties (nCRF effects) can be classified into those that are triggered by stimuli shown within the CRF alone—response saturation [123], cross-orientation suppression [16], and contrast invariance in orientation tuning [127]—and those that are triggered by stimuli shown in and outside the CRF simultaneously—end-stopping, surround suppression, and surround facilitation; for a review, see Seriès et al. [124]. When natural stimuli are used, V1's nCRF effects result in a significant portion (no less than 50 %) of variance in V1 neural responses left unexplained by most existing models [27, 77, 11]. Existing experimental studies suggest that nCRF effects play an important role in V1 response to natural, complex stimuli [141, 51, 23].

Existing studies on the computational mechanisms of V1 nCRF effects [16, 113, 17, 122, 124, 153, 130, 131, 23] can be broadly classified into three types according to the research goals they have [124] as follows.

- *What* (functional): a mathematical (also called functional) description of the input-output transformation from the input stimulus to the output neural response. The main goal here is to predict the neural response given the stimulus as accurately as possible using a mathematical model (linear regression, neural network, etc.). To

be scientifically relevant, only those functional models involving relatively speaking biologically plausible mechanisms will be considered; for example, given two models that explain certain neural phenomena equally well, one that involves computation mechanisms that are considered existent in the brain (addition, division, thresholding, etc.) is more desirable than one involving less likely mechanisms (exponentiation, logarithm, etc.).

- *Why* (normative): a justification of the existence of nCRF effects in the first place. The main goal here is to understand why nCRF effects emerge given constraints of the brain and natural scene statistics of the environment, and what roles these nCRF effects play in visual information processing overall.

- *How* (mechanistic): a mechanistic, biophysical implementation of some functional model of V1 nCRF effects. The main goal here to understand how the brain can achieve the mathematical operations underlying nCRF effects using a neural circuit with biologically realistic components.

In the following, I will give an overview of existing studies, especially those on functional (*what*) and normative (*why*) models, as they are the primary research goals of this proposal. Most existing studies assume that V1 neurons can effectively perform some reconstruction of the input (Section 3.1.1) and they are the main inspiration for my proposed models in Section 4.2.3.2; other studies are discussed briefly in Section 3.1.2.

3.1.1 Reconstruction-enabled studies

In most existing models of nCRF effects, model V1 responses can be used to reconstruct the input stimulus. Specifically, given some input stimulus, the V1 responses of a (trained) model can be transformed back to the input stimulus approximately. In other words, V1 encodes the input stimulus approximately via some invertible transformation.

Note that the reconstruction property of model V1 responses is derived from different normative (*why*) assumptions in different lines of studies. In the sparse coding line of studies [100], the normative assumption puts more emphasis on using sparsely active neurons to reconstruct the input that occur frequently according to natural scene statistics; in the predictive coding line of studies [113], the normative assumption puts more emphasis on using neurons across different areas of the visual system to form a generative model (probability distribution) of the input stimulus according to natural scene statistics so that neural responses can be later used for visual inference in an analysis-by-synthesis approach; in the divisive

normalization line of studies, the normative assumption, if any [145, 23], also assumes that neurons form a generative model according to natural scene statistics.

At the functional (*what*) level, the transformation from input stimulus to V1 responses in these studies can be implemented via some *competition* among model neurons. Various forms of competition are ubiquitous in the brain [36, 14] and are of great research interest. While the word "competition" suggests suppression among neurons, facilitatory effects sometimes can also be explained by these models by suppression with less or negative strength.

Table 3.1 gives an overview of the existing reconstruction-enabled studies in terms of their normative (*why*) assumptions and functional (*what*) implementations, and the following sections describe these studies in more detail.

Table 3.1 Overview of existing reconstruction-enabled studies, in terms of their normative assumptions (why neurons behave this way, in the perspective of visual information processing), and their functional implementations (how to compute the response of model neurons using relatively speaking biologically plausible mechanisms). Note that for each line of studies, normative assumptions and functional implementations may come from different original studies as the research develops. The references for the origins of different normative assumptions and functional implementations are provided in the table.

Name	Normative assumptions	Functional implementations
Divisive normalization	V1 neurons model the input via a Gaussian scale mixture model [145, 23]	Competition between a neuron and its surround in the form of division [15]
Predictive coding	Neurons form a hierarchical probabilistic model, encoding residual errors between the input and the predictable [113]	Divisive input modulation [130, 131]
Sparse coding	Sparsely active neurons reconstruct the input linearly [100]	Pairwise competition [160]; pools of excitatory and inhibitory neurons [72]

3.1.1.1 Divisive normalization

Initially, divisive normalization is mostly a theory at the functional level to explain some nCRF effects within the (classical) receptive field [15, 16], assuming a competition mechanism between a neuron and its surrounding neurons in the form of division for computing the neural response. In general, a neuron with some divisive normalization mechanism computes the ratio between its own (intermediate) response and the summed responses of a pool of

neurons [15]. Later on, the divisive normalization theory has been extended to explain nCRF effects involving the surround [17, 125, 122].

At the normative (*why*) level, the divisive normalization operations used to explain surround-related nCRF effects can be derived by modeling natural images using a Gaussian scale mixture (GSM) distribution [145]. Furthermore, extensions of the GSM, such the mixture of GSMs (MGSM), have shown that the divisive normalization operations for surround-related nCRF effects should be context-dependent to better match neural data; in particular, the strength of divisive normalization should be stronger when there is more redundancy between the visual signal in the center and that in the surround and weaker when there is less [21–23].

3.1.1.2 Predictive coding

Normatively, the predictive coding model [113] proposes that neurons across different visual areas form a hierarchy that encodes the probabilistic distribution of natural stimuli, and neurons from one area try to predict the input they receive. Functionally, neural responses of each visual area are driven by possibly three types of cortical connections: feedforward connections encoding the residual errors between the prediction made by the neurons and the input, feedback connections encoding the prediction made by neurons of higher areas, as well as lateral inhibitory connections among the competing neurons in this area.

The original predictive coding model [113] can explain nCRF effects such as end-stopping. Later on, with additional nonlinearities and more biological computing rules such as divisive input modulation [133], predictive coding models from [130, 131] are able to produce more nCRF effects such as cross-orientation and surround suppression, as well as to produce other tuning properties of V1.

3.1.1.3 Sparse coding

Sparse coding [101] has been initially proposed as a normative theory, hypothesizing that information in the brain is represented by a relatively small fraction of neurons that are simultaneously active. Various studies [100, 8, 139, 138] have shown that the receptive fields of V1 simple cells are adapted to natural scene statistics following the sparse coding principle. Apart from explaining V1 cells, the sparse coding theory has been shown to be consistent with many other experimental studies as well [101, 141, 142, 29].

Functionally, the sparse coding theory can be implemented in various ways. One particular implementation [160] involving pairwise competition of model neurons in a dynamical system has been shown to match physiological data well in terms of nCRF effects, both for

individual neurons and at a population level. Other implementations are possible, such as having separate populations of excitatory and inhibitory neurons [72]. However, the predictive power of these alternative implementations on nCRF effects have yet to be demonstrated.

3.1.2 Other studies of V1 nCRF effects

Apart from the three types of studies examined above, V1 nCRF effects can also be derived from other mathematical (functional) operations and normative principles. In Zetzsche and Röhrbein [154], nCRF effects have been demonstrated in a two-layer neural network, which has rectification nonlinearity and was optimized to reduce the dependencies among output units. In Zetzsche and Nuding [153], model neurons with nCRF effects have been shown to be selective to *intrinsically two-dimensional (i2D) signals* [152], which are the most informative ones in natural scenes [153].

3.2 Existing work on using CNNs for neural data modeling

In recent years, neural network models have been used to study various visual areas. For early visual areas, Boltzmann machine and its variants have found their success in modeling population response patterns for ganglion cells [48, 121, 43] and predicting functional connectivity of V1 disparity-tuned neurons [158]; convolutional neural networks (CNNs) have recently been found to be more effective than more classical methods for modeling retinal [71] and V1 neurons [95, 11, 73]. For higher visual areas, CNNs, especially deep ones, have been used for explaining inferotemporal cortex (IT) [150, 79, 151]; they form the only model class in computational neuroscience that can predict IT responses to novel stimuli with reasonable accuracy [70].

There are at least two different ways to model V1 neurons and neural data in general using (convolutional) neural network models: data-driven and transfer learning as described in the following.

3.2.1 Data-driven approach

In the data-driven approach, neural network models are simply trained from scratch to fit the neural data, using only input stimuli and their associated neural responses; this is the approach taken in my recent study [157] and many other very recent ones [71, 95, 73]. For CNNs, the data-driven approach using has been mostly applied to early visual neurons such as retinal ganglion cells [95] and V1 neurons [71, 11, 73, 157]. Most studies use fairly typical CNNs that have multiple convolutional blocks—each of which performs convolution that is

optionally followed by nonlinear activation, batch normalization [61], and pooling—followed by a fully connected readout layer (possibly with a nonlinearity layer at the very end of the model) that linearly combines output features of the last convolutional block to get predicted neural responses. All CNN models in the literature outperformed corresponding baseline models such as linear-nonlinear (LN) models [19] and generalized linear models (GLMs) [107].

3.2.2 Transfer learning approach

In the transfer learning (also called goal-driven) approach [11, 151, 79], there are two phases of training for fitting neural data; in the pre-training phase, a neural network model is trained on a task relevant to the function of neurons (e.g., training a deep CNN on image classification for modeling IT neurons as in Yamins and DiCarlo [151]) using large-scale labeled data sets such as ImageNet [116]; in the fine-tuning phase, outputs of fitted units in the trained model are (linearly) combined as predicted neural responses. The transfer learning approach has achieved most of its success in explaining neural data from higher visual areas such as V4 and IT [151, 69]; for fMRI data, the transfer learning approach provide effective models for lower visual areas V1 through V3 as well [69].

3.3 Existing work on recurrent computation components in neural data modeling

While there has been considerable success in predicting neural data using CNNs via data-driven and transfer learning approaches (Section 3.2), there are two key structural components missing in the CNNs typically used in existing studies: local horizontal recurrent connections between neurons in the same area [65] and long-range feedback recurrent connections between neurons in different areas [39]. These connections greatly contribute to the complexity of the visual system, and may be essential for the success of the visual systems in reality; for example, there are evidences that recurrent connections are crucial for object recognition under noise, clutter, and occlusion [102, 129, 112].

The usage of recurrent connections in neural network models is relatively new in both artificial intelligence and neuroscience communities. In the artificial intelligence community, feedback recurrent connections from higher layers of a CNN can help sharpen and refine representations of lower layers [86, 12, 62] to achieve better image recognition performance under occlusion and noise [146, 147, 159]; various neurally inspired neural network modules with local horizontal recurrent connections [25, 88] have been shown to have similar or better

performance than standard recurrent neural network modules, such as LSTM [55] and GRU [20], in various tasks. In the neuroscience community, explaining neural data quantitatively using neural networks with recurrent computation components is still in its infancy; Nayebi et al. [97] have shown that deep CNNs with custom recurrent computation components can explain dynamics of neural activity in V4 and IT better than feedforward CNNs, and Lotter et al. [90] have shown that PredNet [91], a neural network model for video prediction based the predictive coding principle [113] with bottom-up and top-down connections, is able to capture a range of seemingly disparate phenomena observed in the visual cortex.

Chapter 4

Proposed work plan and preliminary results

4.1 Overview

In the proposed final project, I plan to advance our understanding of recurrent circuits of the primary visual cortex and the visual system in general, in a two-part investigation.

First (Section 4.2), I will try to find candidate models for recurrent circuits of V1, by designing and evaluating different neural network models with recurrent computation components for predicting V1 neural responses to natural images as well as predicting other phenomena observed in V1 neurophysiology studies. The rationale here is that better-performing models for V1 data may have more similarities to the biological reality [157, 151]. Compared to feedforward neural network models used in typical data-driven and transfer learning methods (Section 3.2), the candidate models to be explored will feature new designs from two complementary aspects: model architecture (recurrent vs. feedforward computation components) and training methodology (loss functions, number of training phases, etc.). Preliminary results show that models with these two new designs can perform as well as state-of-the-art approaches with fewer parameters and less data (Section 4.2.3).

Second (Section 4.3), I propose to explain the role of recurrent computation components in these high-performing models for explaining V1 data, by first simplifying models with various model compression techniques without sacrificing much performance and then analyzing the simplified models using various tools for neural network analysis as well as existing knowledge about V1. In particular, I propose to use recurrent computation components in my candidate models to 1) provide alternative and more-detailed explanation of contextual modulation [23], a well-known phenomena in V1, and 2) explain familiarity

effect in early visual areas [56], a newly found phenomena in V2 and early visual areas in general. The end product of achieving any of the two above goals would be a valuable contribution to NIPS or other high-profile neuroscience conferences and journals.

Overall, the proposed work will yield two deliverables. First, more interpretable recurrent models that can predict V1 neural responses to natural images and various V1 phenomena accurately will be developed. Second, correspondence between model components and biological reality will be established, and new insights about the roles of recurrent circuits in V1 and visual signal processing in general will be provided in the context of contextual modulation and familiarity effect.

4.2 Part 1: finding candidate models for recurrent circuits of V1

In this part of the proposed study, I will try to find candidate models for recurrent circuits of V1, by designing and evaluating different neural network models with recurrent computation components for predicting V1 neural responses to natural images as well as predicting other phenomena observed in V1 neurophysiology studies. The rationale here is that better-performing models for V1 data may have more similarities to the biological reality [157, 151]. Compared to feedforward neural network models used in typical data-driven [11, 73, 157] and transfer learning [11] methods (Section 3.2), the candidate models to be explored will feature new designs from the following two complementary aspects.

- Novel model architecture with recurrent computation components. As mentioned in Section 3.3, two key structural components are missing in the CNNs typically used in existing studies for fitting V1 data: local horizontal recurrent connections between neurons in the same area [65] and long-range feedback recurrent connections between neurons in different areas [39]. Recurrent computation components conceptually corresponding to these structural components will be explored in the models to be developed and evaluated.

- Better training methodology in terms of auxiliary tasks. and optimization schemes. As discussed in Section 4.2.1, potential performance improvement is possible by combining and enhancing the strengths of the standard data-driven and transfer learning approaches without changing their feedforward model architectures. First, given that V1 neurons are typically considered to process low-level image features such as oriented edges, additional auxiliary or pre-training tasks, especially those self-supervised

ones with less semantics information such as image reconstruction and video prediction, might be used together with the standard supervised image classification task to provide better constraints for learning feature representations useful for predicting V1 data; identifying auxiliary tasks useful for predicting V1 data will provide insights on the role of V1 in visual information processing [151]. Second, there might be some alternative optimization scheme that fuses the two-phase (pre-training followed by fine-tuning) scheme used in the transfer learning approach and the one-phase (trained from scratch) scheme used in the data-driven approach.

Visually, the two new designs focus on different and complementary parts of the model training pipeline (Figure 4.1). See the caption for details.

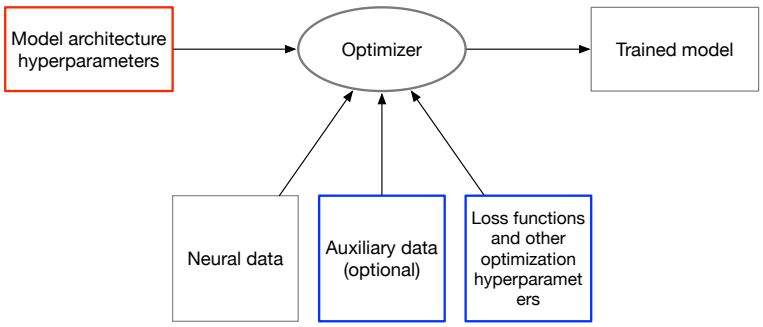


Fig. 4.1 The model training pipeline for predicting neural data. Conceptually the trained model is the output of an optimizer that takes four components as the input: model architecture hyperparameters, neural data, optional auxiliary data, and optimization hyperparameters such as loss functions (here loss functions are considered part of the optimization process, not that of the model). The first new design (red) improves the model architecture component by introducing recurrent computation components; the second new design (blue) improves training methodology by changing optimization hyperparameters and optionally auxiliary data.

To find candidate models for recurrent circuits of the primary visual cortex, I will develop and test neural network models with recurrent computation components against feedforward models on various V1 data sets under different metrics. Apart from measuring model performance by the correlation between predicted and ground-truth V1 responses to natural images [11, 73], I will also compare models under other metrics applicable to V1 data, in particular surround modulation ratio [23] and familiarity suppression index [56], as they will be investigated later on in the second part of the proposed study to understand the role of recurrent circuits in V1. These performance metrics derived under different motivations and from different data sets will put a strong constraint on plausible model architectures with recurrent computation components for modeling V1 [151].

4.2.1 Why fusing data-driven and transfer learning approaches with better training methodology

While transfer learning [151, 79] has achieved great success in explaining neural responses of IT area neurons (see Section 3.2 for details), this approach cannot completely replace the simpler data-driven approach [157, 11] that trains models from scratch to fit V1 data. In particular, across different studies on V1, there is no clear winner of the two approaches.

- In Cadena et al. [11], the transfer learning approach performed overall better[1] than the data-driven approach on V1 responses to rapidly changing sequences of natural images.
- In Zhang et al. [157], the data-driven approach performed overall better (around 7.5 % higher in terms of explained explainable variance; see Figure 11 of Zhang et al. [157]) than the transfer learning approach on calcium imaging data of V1 to a large number of complex shapes [134].
- In the following baseline experiment (Section 4.2.1.1) on the V1 data [76] in Coen-Cagli et al. [23], the state-of-the-art data-driven CNN [11, 73] performed better overall (around 11.5 % higher in terms of raw Pearson correlation coefficient averaged over neurons) but with mixed results on individual neurons, compared to my data-driven approach on VGG networks [126].
- Finally, pre-training task in the standard transfer learning approach is limited to image classification, which is not directly related to V1 neurons. Given that V1 neurons are typically considered to process low-level image features such as oriented edges, I feel additional auxiliary or pre-training tasks, especially those self-supervised ones with less semantics information such as image reconstruction and video prediction, might be used instead or together with image classification to provide better constraints for learning feature representations useful for predicting V1 data.

Therefore, it would be desirable to combine and enhance the strengths of the two approaches together in a single framework that would predict V1 responses with higher performance than either of the two approaches above.

[1] While the authors considered the differences to be statistically insignificant ($p = 0.09$), I think the p-value in this case was small and it would be smaller and become significant with more neurons in the data.

4.2.1.1 Baseline experiment showing the potential usefulness of fusing data-driven and transfer learning approaches

This experiment is designed to illustrate the insufficiency of the transfer learning approach that achieves state-of-the-art results for IT neurons [151, 79]. Here, I compare the state-of-the-art data-driven [11, 73] and transfer learning [11] approaches for modeling V1 data. In my experiment, the data-driven CNN has an overall structure similar to that in Cadena et al. [11] with three convolutional layers, a factored fully connected layer [73], and some changes in kernel sizes of convolutional layers for adaptation to our data; the transfer learning approach evaluated here is similar to that in Cadena et al. [11] with experiment details following those in Zhang et al. [157]. For other experiment details, check Section 4.2.5.

Results Results are shown in Figure 4.2. Within the limit of my hyperparameter tuning, I found that the data-driven approach performed better than the transfer-learning approach, but the former did not outperform the latter consistently over each neuron; the results suggest that fusing two approaches together may further improve the state-of-the-art performance for predicting V1 responses.

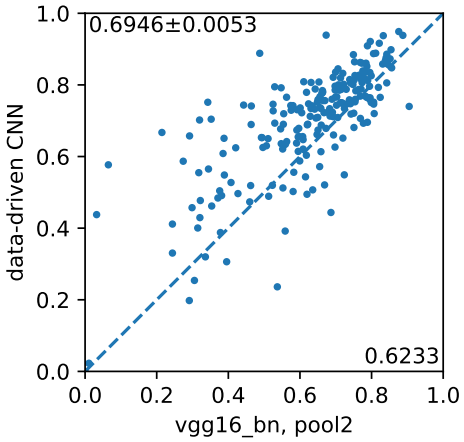


Fig. 4.2 Neuron-by-neuron comparison of a data-driven CNN (results averaged over 10 times with different initialization seeds) and a transfer learning approach using a batch normalized VGG16's `pool2` layer. Performance is measured in Pearson correlation coefficient and average correlation coefficients over all neurons are shown at corners; for the data-driven CNN, the standard deviation of the 10 average correlation coefficients obtained with different seeds is also shown. For the transfer learning approach, other layers or VGG network variants gave worse results.

4.2.2 Proposed methods

Formally, all the explored models will be trained by minimizing the following cost function for each input stimulus $\vec{\boldsymbol{x}}$ and its associated neural response vector $\vec{\boldsymbol{r}}$, with respect to the model parameter vector $\vec{\boldsymbol{\theta}}$:

$$L(\vec{\boldsymbol{\theta}};\vec{\boldsymbol{x}},\vec{\boldsymbol{r}},\vec{\boldsymbol{y}}^{(1)},\ldots,\vec{\boldsymbol{y}}^{(K)}) = \alpha_0 l_0(f_0(\vec{\boldsymbol{x}};\vec{\boldsymbol{\theta}}),\vec{\boldsymbol{r}}) + \sum_{i=1}^{K} \alpha_i l_i(f_i(\vec{\boldsymbol{x}};\vec{\boldsymbol{\theta}}),\vec{\boldsymbol{y}}^{(i)}). \tag{4.1}$$

In Eq. (4.1), $\vec{\boldsymbol{x}}$ is the (vectorized) input stimulus; $\vec{\boldsymbol{r}} \in \mathbb{R}^M$ is the neural response vector for the input stimulus for a data set of M neurons; $\vec{\boldsymbol{y}}^{(1)},\ldots,\vec{\boldsymbol{y}}^{(K)}$ are the label vectors, which are reduced to scalars for tasks like image classification and proper vectors for tasks like image reconstruction and object detection, of K relevant auxiliary tasks; $f_0, f_1, \ldots, f_K$ are $K+1$ neural network models that all take $\vec{\boldsymbol{x}}$ as input and outputs the predicted neural response vector and the predicted output responses of the K auxiliary tasks, respectively, with the parameter vector $\vec{\boldsymbol{\theta}}$ (partially) shared among $K+1$ models; likewise, $l_0, l_1, \ldots, l_K$ are $K+1$ loss functions that measure the differences between the predicted responses from $f_0, f_1, \ldots, f_K$ and the ground-truth labels $\vec{\boldsymbol{r}}, \vec{\boldsymbol{y}}^{(1)}, \ldots, \vec{\boldsymbol{y}}^{(K)}$; $\alpha_1, \ldots, \alpha_K$ are the weighting factors for the K auxiliary tasks relative to the neural response loss term l_0, which typically has a trivial weighting factor of $\alpha_0 = 1$. Eq. (4.1) defines the optimization objective function for a single input stimulus with its associated neural responses and labels; in practice, the actual objective function to be optimized is the average of Eq. (4.1) over all stimuli in the training data plus some regularization terms such as ℓ_2 weight decay. In cases where some stimuli are only available for the neural data or auxiliary tasks, e.g. transfer learning, the terms corresponding to the missing stimuli are simply dropped.

Potential improvements on different terms of Eq. (4.1) will be explored in the proposed study. The first new design, novel model architecture, will explore variations of $f_0, f_1, \ldots, f_K$, and the second new design, better training methodology, will explore variations of $l_0, l_1, \ldots, l_K$ and adjusting $\alpha_0, \alpha_1, \ldots, \alpha_K$ as well as learning rates during training in some principled way.

The cost function presented in Eq. (4.1) fuses and extends standard data-driven and transfer learning approaches. It is reduced to the standard data-driven approach if $K = 0$, and it is equivalent to the standard transfer learning approach if we adopt a two-phase training scheme, where we set the weight factor α_0 for neural response loss term to be zero in the first phase and later set the weight factors $\alpha_1, \ldots, \alpha_K$ for auxiliary tasks to be zero in the second phase.

Compared to the naive data-driven approach, my fused approach makes use of labels of relevant auxiliary tasks as additional supervision signals during training, and these additional

supervision signals may reduce the chance of overfitting, which is common for most neural data sets that are relatively small due to technological limitations. Compared to the transfer learning approach with pre-trained models on image classification, my fused approach has two differences: first, my approach allows flexible training schemes that balance the neural response loss term and auxiliary task terms in different ways (fixed α_j's in one phase, two sets of α_j's in two phases, etc.), while the transfer learning approach minimize two sets of terms separately in two phases; second, my approach generalizes the single image classification task in common implementations [11, 151, 79] of the transfer learning approach to multiple relevant tasks that jointly constrain the neural response prediction part of the model.

Table 4.1 Comparison of four different methods for predicting V1 neural data. For each method, we list its model architecture and its associated auxiliary or pre-training tasks, if any. In the "Architecture" column, "recurrent" means that the model has recurrent computation components such as LSTM [55], and "semi-recurrent" means that the model has computation components that are technically feedforward but can be roughly considered as performing some recurrent computations unrolled over time. Check Figure 4.3 as well.

Method	Architecture	Auxiliary or pre-training tasks
Data-driven [11, 73, 157] in Section 4.2.1.1	feedforward	—
Transfer learning [11] in Section 4.2.1.1	feedforward	image classification; supervised
Models in Section 4.2.3.2	semi-recurrent	image reconstruction; self-supervised
Transfer learning with PredNet [91] in Section 4.2.3.1	recurrent	video prediction; self-supervised

Table 4.1 gives a summary of four different methods evaluated and compared in this chapter for fitting V1 neural data in terms of auxiliary tasks used and model architectures.

I will preliminarily demonstrate the usefulness of recurrent computation components and other alternative auxiliary tasks in Section 4.2.3. In particular, the first Experiment (Section 4.2.3.1) used PredNet [91], which is trained on video data in a self-supervised fashion, to fit V1 neural data in a transfer learning setting. Results show that PredNet performed as well as VGG networks, which are used in state-of-the-art transfer learning approaches, with fewer model parameters, less training data, and less supervision. Together with the ability of PredNet for explaining many neural phenomena [90], the preliminary results suggests the plausibility to develop compact recurrent models for explaining V1 neural responses to natural images and other V1 phenomena.

As shown in Figure 4.3, the four V1 modeling methods evaluated and compared here (two in Section 4.2.1.1 and two in Section 4.2.3) are just four individual points in the vast conceptually 2D space of model architecture and auxiliary task(s), with optimization scheme (which is also to be explored) ignored. Other recently developed recurrent models, such as

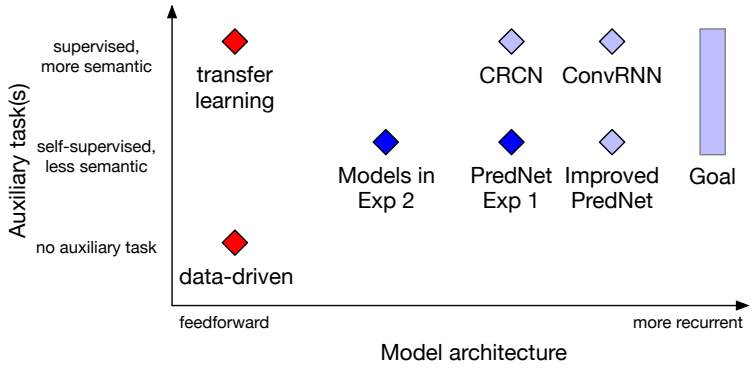


Fig. 4.3 Different methods for predicting V1 neural data visualized in the 2D space of model architecture (horizontal axis) and auxiliary task(s) (vertical axis). Compare with Table 4.1. Standard data-driven and transfer learning approaches are marked as red dots; newly tested recurrent models in Sections 4.2.3.1 and 4.2.3.2 are marked as dark blue dots; untested new recurrent models, CRCN [2], improved PredNet [3], and ConvRNN [97], are marked as light blue dots; the ultimate goal of this proposal in terms of model searching is marked as a light blue rectangle spanning multiple auxiliary tasks of multiple levels with highly recurrent computation components.

CRCN [2], an improved PredNet [3], and ConvRNN [97], are also in this 2D space but their usefulness for explaining V1 neural data have yet to be tested due to limited time.

The ultimate goal of this proposal (the rectangle in the far right of Figure 4.3) in terms of model searching is to develop recurrent models that can be trained end-to-end to predict V1 neural responses accurately with constraints from multiple auxiliary tasks of different levels of supervision and semantics information. These candidate models for recurrent circuits of V1 will be analyzed and interpreted according to the proposed plan in Section 4.3.

4.2.3 Preliminary experiments and results

Here I will demonstrate the usefulness of recurrent computation components and other alternative auxiliary tasks via two Experiments (Sections 4.2.3.1 and 4.2.3.2). The first Experiment demonstrated the usefulness of recurrent models together with self-supervision pre-training for predicting V1 responses in a transfer-learning setting; the second Experiment demonstrated the usefulness of image reconstruction as an auxiliary task in a data-driven setting. Notably, the first Experiment (Section 4.2.3.1) used PredNet [91], which is trained on video data in a self-supervised fashion, to fit V1 neural data in a transfer learning setting. Results show that PredNet performed as well as VGG networks, which are used in state-of-the-art transfer learning approaches, with fewer model parameters, less training data, and less supervision. Together with the ability of PredNet for explaining many neural phenomena

[90], the preliminary results suggests the plausibility to develop compact recurrent models for explaining V1 neural responses to natural images and other V1 phenomena.

4.2.3.1 Experiment 1: recurrent models trained with self-supervision performed as well as feedforward ones in a transfer learning setting

As an initial test on the performance of recurrent models for explaining V1 data, I tested PredNet [91] on the V1 data [76] in Coen-Cagli et al. [23]. The pre-trained L_0 variant PredNet in Lotter et al. [91], which was trained on video data in an unsupervised manner, was used to extract feature representations of input stimuli. Each input stimulus was presented to the network for eleven time steps (with the same input) and the network's responses of E and R units across four layers during time steps 2–11 were extracted, with eight feature representations (four layers and two types of units) per time step for each input stimulus. The V1 responses were fit with the extracted feature representations in a transfer-learning approach similar to those in Section 4.2.1.1 and Zhang et al. [157].

As shown in Figure 4.4, PredNet[2] performed similarly to VGG networks. The performance parity between PredNet and VGG networks may be due to the recurrent computation components in the former given that the tested PredNet has much fewer model parameters, less training data, and less supervision than VGG networks. Together with the ability of PredNet for explaining many neural phenomena [90], the preliminary results suggests the plausibility to develop compact recurrent models for explaining V1 neural data.

4.2.3.2 Experiment 2: usefulness of image reconstruction as an auxiliary task in a data-driven setting

Two-phase training paradigm The first experiment is designed to illustrate the usefulness of *image reconstruction* as an alternative auxiliary task for modeling V1 data, with a two-phase training paradigm like that of the transfer learning approach but *without additional labeled data* (only images with associated neural responses in the neural data set are used in the pre-training phase). While typical transfer learning implementations [79, 151] use image classification as the pre-training task, other choices are possible for modeling V1 neurons, which presumably do not directly participate in object recognition as IT neurons where transfer learning works best. One alternative choice to image classification is image reconstruction, which is an essential optimization criterion in many unsupervised learning studies on natural images [100, 82, 132, 113]. There are at least two sets of arguments

[2] the third layer's R units, averaged across time steps 2–4 were used; adjacent layers and units showed similar performance.

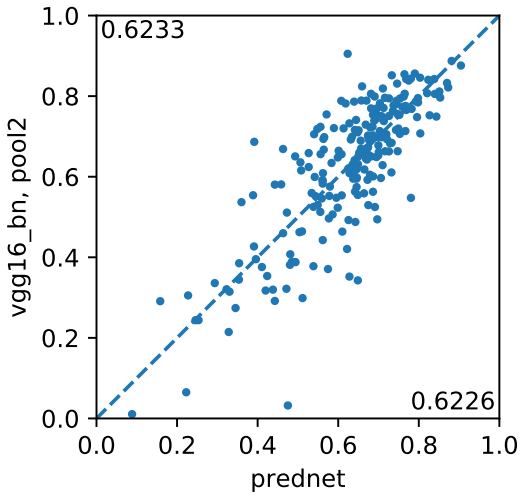


Fig. 4.4 Neuron-by-neuron comparison of PredNet and VGG. The layout of the figure is similar to that of Figure 4.2.

for using image reconstruction as an auxiliary task for modeling V1: first, most existing studies on V1 nCRF effects (Section 3.1) assume conceptually that V1 neurons perform some reconstruction of the input; second, V1 neurons are typically considered to process low-level image features such as oriented edges [28], and image reconstruction intuitively feels like a task that can be handled by V1 neurons better than image classification.

Overall, the model explored in this experiment is a one-convolutional-layer CNN, with the parameters for the convolutional part initialized to those of a pre-trained model on sparse coding, which performs well in explaining V1 nCRF effects [160]. See Figure 4.5.

In the pre-training phase ($\alpha_0 = 0, \alpha_1 = 1$ in Eq. (4.1)), a one-convolutional-layer sparse coding network model was trained on a large set (100,000) of image patches of size 25 px by 25 px cropped from images in the neural data set; the architecture of the model follows Section 3 of Kavukcuoglu et al. [66], which is arguably the state-of-the-art feedforward model for approximating the standard sparse coding. Given some input stimulus, the model passes the input through a convolutional layer of 64 channels of 9 by 9 kernels, a tanh nonlinearity layer, and a scaling layer in sequence to generate the approximate sparse representation I of the input. Other experiment details, such as image preprocessing, follow those in Kavukcuoglu et al. [66] as much as possible.

In the fine-tuning phase ($\alpha_0 = 1, \alpha_1 = 0$ in Eq. (4.1)), we attach the one-convolutional-layer sparse coding network model in the pre-training phase with a batch normalization layer [61], which handles feature scaling, a factored fully connected layer [73], and a soft-thresholding nonlinearity layer $x \mapsto \log(1+\exp(x))$ [73] (called *softplus* in Dugas et al. [35]) in sequence to generate the predicted neural responses. Model parameters that correspond

to those in the sparse coding network model are initialized by the pre-trained model; all parameters, instead of only additional parameters in the transfer learning approach, in the model are optimized during training as in the naive data-driven approach. For other experiment details, see Section 4.2.5.

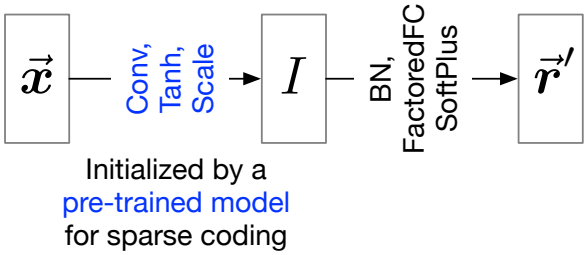
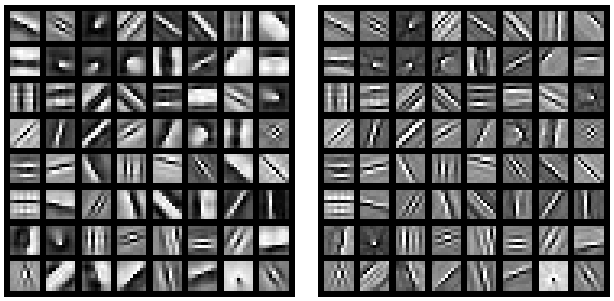


Fig. 4.5 Diagram of the proposed model in Experiment 2, two-phase variation. Check Section 4.2.3.2 for details. The notation overall follow that in Eqs. (4.2).

Results With two phases of training, my model performed with an average correlation coefficient of 0.4031 ± 0.03, with the standard deviation computed over 5 different initialization seeds. For comparison, if the pre-training phase was dropped and the convolutional part of the model was initialized randomly instead of by a pre-trained model, the performance dropped to 0.3288 ± 0.01. While right now I cannot make similar performance gains on a CNN architecture similar to that in the standard data-driven method (Section 4.2.1.1, the current results shows that image reconstruction can be used as an auxiliary task for modeling V1 data in a data-driven setting. In addition, the pre-trained convolutional sparse coding model is visualized in Figure 4.6.



(a) (b)

Fig. 4.6 Parameters in the learned pre-trained model. Panel (a) shows the learned convolutional sparse coding dictionary, and panel (b) shows the feedforward kernels for approximating the sparse representation.

One-phase training paradigm The second experiment is designed to illustrate the feasibility of a simple one-phase end-to-end training paradigm as proposed in Eq. (4.1) with all α_j's fixed throughout the training. Compared to the two-phase training paradigm used in transfer learning and the previous two-phase experiment (Section 4.2.3.2), a one-phase training paradigm is conceptually simpler and requires less manual intervention of the training process. Given some input stimulus, the model first passes the input through a convolutional layer of channels of 9 by 9 kernels and a batch normalization layer to generate the intermediate feature map. Then the feature map is processed through two routes: in the neural response prediction route (blue in Figure 4.7), the model passes the intermediate feature map through a factored fully connected layer and a soft-thresholding nonlinearity layer as in the models of other experiments; in the image reconstruction route (red in Figure 4.7), the model passes the intermediate feature map through a transposed convolutional layer (also called deconvolutional layer in the literature) to invert the feature map back to the input space, trying to reconstruct the input stimulus. Formally, for each vectorized image $\vec{\boldsymbol{x}}$ and its associated neural response vector $\vec{\boldsymbol{r}}$, the model minimizes the following Eq. (4.2a) that is a special case of Eq. (4.1):

$$L(\vec{\boldsymbol{\theta}};\vec{\boldsymbol{x}},\vec{\boldsymbol{r}}) = l_0(f_0(\vec{\boldsymbol{x}};\vec{\boldsymbol{\theta}}),\vec{\boldsymbol{r}}) + \alpha\|f_1(\vec{\boldsymbol{x}};\vec{\boldsymbol{\theta}}) - \vec{\boldsymbol{x}}\|_2^2, \tag{4.2a}$$

$$I = \mathrm{BN}(\mathrm{Conv}(\vec{\boldsymbol{x}};\vec{\boldsymbol{\theta}});\vec{\boldsymbol{\theta}}), \tag{4.2b}$$

$$f_0(\vec{\boldsymbol{x}};\vec{\boldsymbol{\theta}}) = \mathrm{SoftPlus}(\mathrm{FactoredFC}(I;\vec{\boldsymbol{\theta}});\vec{\boldsymbol{\theta}}), \tag{4.2c}$$

$$f_1(\vec{\boldsymbol{x}};\vec{\boldsymbol{\theta}}) = \mathrm{ConvT}(I;\vec{\boldsymbol{\theta}}). \tag{4.2d}$$

In Eqs. (4.2), the intermediate feature map I as in Eq. (4.2b) is shared between the neural response prediction route f_0 as in Eq. (4.2c) and the image reconstruction route f_1 as in Eq. (4.2d); finally, two routes are combined in Eq. (4.2a) with a weighting factor α; l_0 denotes the Poisson loss function as used in Klindt et al. [73]; $\mathrm{Conv}, \mathrm{ConvT}, \mathrm{FactoredFC}, \mathrm{SoftPlus}, \mathrm{BN}$ denote convolutional layer, transposed convolutional layer, factorized fully connected layer, soft-thresholding nonlinearity layer, and batch normalization layer, respectively. For other experiment details, see Section 4.2.5.

Results Results are shown in Figure 4.8. I tried different values for the weighting factor α in Eq. (4.2a). As shown in Figure 4.8a, my models ($\alpha \neq 0$) outperformed the baseline ($\alpha = 0$) over a large range of values for α. In addition, the performance gain was also reflected in the learned convolutional filters (Figure 4.8c vs. Figure 4.8b). While right now I cannot make similar performance gains on a CNN architecture similar to that in the two-phase variation, this performance gap should be non-essential, as both Experiments make use

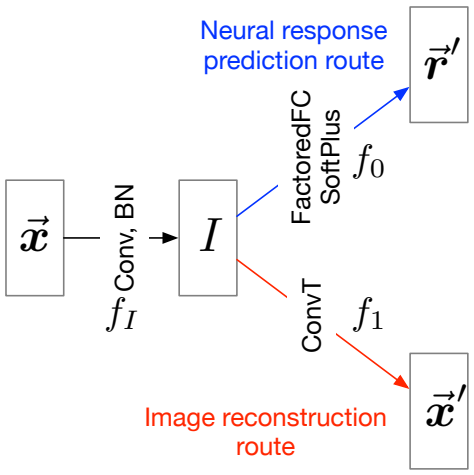


Fig. 4.7 Diagram of the proposed model in Experiment 2, one-phase variation. Check Section 4.2.3.2 and Eqs. (4.2) for details.

of the same training data without using additional labeled data as in the transfer learning approach. With additional improvements on the training procedure—using different learning rates for different layers, dynamically adjusting the weighting factor along different stages of training via some learning algorithm. etc.—an end-to-end one-phase training paradigm should achieve comparable or better performance than the two-phase training paradigm in Section 4.2.3.2. Overall, the current results are a proof-of-concept that image reconstruction can be used as an auxiliary task for modeling V1 data under a one-phase training paradigm in a data-driven setting.

4.2.4 Proposed work to do

Based on existing preliminary results, I plan to work in the following directions next, focusing on those in Section 4.2.4.1.

4.2.4.1 High priority tasks

More recurrent model architectures under more metrics Existing neural network models with recurrent components [97, 90, 91, 23] have not been rigorously tested with V1 data under various metrics. I propose to test neural network models with existing or newly designed recurrent computation components against feedforward models on various V1 data sets under different metrics, such as correlation between predicted and ground-truth responses [11, 73], surround modulation ratio [23] and familiarity suppression index [56]. In line with goal-driven modeling in Yamins and DiCarlo [151], these metrics derived under different

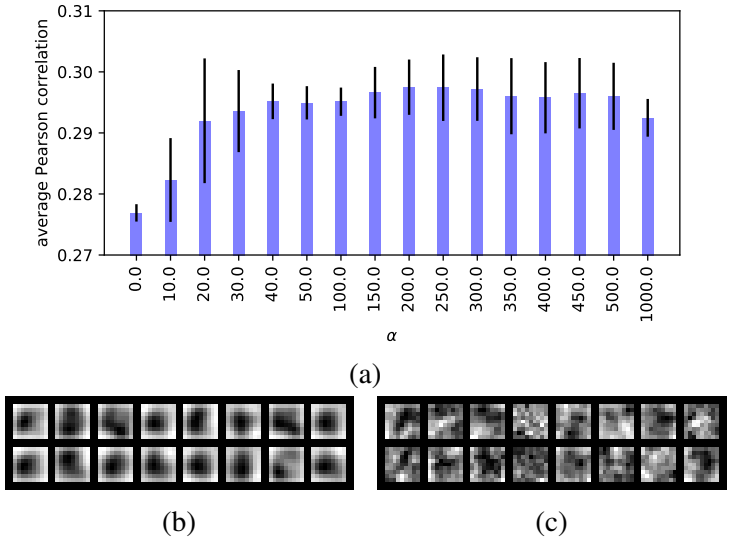


Fig. 4.8 Results of Experiment 2, one-phase variation. Panel (a) shows the results for different values of the weighting factor α, and standard deviations were computed over 9 different initialization seeds. Panels (b) and (c) visualize the convolutional layers for models with $\alpha = 0$ and $\alpha = 200$, respectively.

motivations and from different data sets will put a strong constraint on plausiable model architectures with recurrent computation components for modeling V1.

More demonstration on the usefulness of self-supervision tasks While PredNet under a transfer learning setting showed competitive performance with the state of the art, the currently experimented models that demonstrated the usefulness of image construction task in a data-driven setting are clearly very simple and did not exihibit performance similar to that of the state of the art. The effectiveness of image reconstruction or other auxiliary tasks in data-driven, transfer learning, or other scenarios need to be further demonstrated.

Potential neuroscience insights of auxiliary tasks While using mutliple auxiliary tasks may help neural prediction, it is also important to understand the relative contributions of different auxiliary tasks to the learned feature representations. As discussed in Yamins and DiCarlo [151], while feature representations learned by image classification generalize to many other visual tasks [32], there may exist visual tasks requiring separate, independent optimizations; these tasks may help us better understand the function of V1 and visual cortex in general.

4.2.4.2 Others

Alternative loss functions measuring the difference between the stimulus and the reconstruction The ℓ_2 squared loss function used in Experiment 2 (Section 4.2.3.2) may be replaced by other loss functions for better results. One possibility is to quantize the outputs and use the cross-entropy loss used in multi-class classification problems; existing studies show that ℓ_2 squared loss tends to get more blurry results as the trained model tends to predict the mean pixel intensity only [106, 31, 155]. Another possibility is to use some adversarial losses [106, 33] inspired by generative adversarial networks [47]. Finally, loss functions that capture perceptual distortions between images [5, 81] may be explored as well.

Other auxiliary tasks Other than image reconstruction, other auxiliary tasks that may be more efficient for learning useful feature representations of the stimulus can be used, such as cross-channel prediction [156], context inpainting [106], relative position prediction [30], etc.

Multiple auxiliary tasks Doersch and Zisserman [31] has shown that having multiple auxiliary tasks can improve performance for various visual tasks. What tasks to choose and how to combine them in a unified model for improving neural response prediction (rather than computer vision tasks) remain open questions.

Better optimization schemes Apart from the one-phase training scheme used in the data-driven approach and the two-phase scheme used in the transfer learning approach, some middle ground may exist between the two optimization schemes to better utilize supervision from neural data and auxiliary tasks more effectively.

4.2.5 Data, model training, and evaluation details for certain experiments

Data I used the V1 data [76] in Coen-Cagli et al. [23] for model evaluation. Specifically, 540 natural images, which contains 270 pairs of small (1° in visual angle) and large (6.7°) ones, and the responses of 221 V1 neurons to these images were used to train and test different models. The set of 540 images along with their associated neural responses is split into training, validation, and testing sets in the proportions of $80\% \times 80\% = 64\%$, $80\% \times 20\% = 16\%$, and 20%, as in Zhang et al. [157].

Model training The model training procedure follows that in Klindt et al. [73] as much as possible. Poisson loss function is used to quantify the difference between predicted

neural responses and ground-truth responses; other training details—optimizer, learning rate scheduling, early stopping, additional regularizations, etc.—are mostly the same as those in the publicly available code[3] of Klindt et al. [73].

Evaluation The model evaluation procedure exactly follows that in Klindt et al. [73]; the raw Pearson correlation coefficient is computed between the predicted neural responses on the testing set and the ground-truth responses (averaged over trials) for each of 221 neurons, and these 221 Pearson correlation coefficients are averaged to quantify the model's performance (the higher the better).

4.3 Part 2: analyzing and interpreting models

After getting high-performing neural network models with recurrent computation components following the plan proposed in Section 4.2. I will first simplify these models and then analyze them in detail, to explain the performance gain of neural network models with recurrent computation components from the perspective of neuroscience and that of machine learning.

- **Model simplification**. While deeper and more complex models may achieve better performance [97, 161], they are naturally more difficult to analyze. To make subsequent analyses easier, I will first try to simplify models in various ways, such as by reducing the number of layers and parameters [4, 114], sparsifying model parameters [49], and replacing more complicated components like LSTM [55] with simpler ones such as GRU [20] or non-recurrent components. The simplification process used should induce no or little performance decrease [18].

- **Model analysis**. To understand models from both the perspective of neuroscience and that of machine learning, visualization, dissection, and ablation methods [157, 98, 99] will be applied to the simplified models that capture the essence of original ones; in addition, I will try to map components of (simplified) models to those of neural circuits [25, 88].

As mentioned in Section 4.1, I will focus on explaining the role of recurrent ciruits in the following two phenomena. The end product of the proposed study would be a valuable contribution to NIPS or other high-profile neuroscience conferences and journals.

- **Alternative and better-performing explanation of contextual modulation in V1 responses to natural images**. Contextual modulation is a well-known phenomenon

[3] https://github.com/david-klindt/NIPS2017

in V1 [124], and Coen-Cagli et al. [23] have tried explaining contextual modulation for natural images, showing that a Gaussian scale mixture model involing some gating mechanism explained the data better than typical divisive normalization models without gating. However, while gating mechanism in general can be important for recurrent circuits as shown in many studies [97, 25, 88], alternative gating mechanisms may provide better prediction of the available data. In addition, the model introduced in Coen-Cagli et al. [23] requires complex training procedures and is not useable for modeling arbitrary V1 data.

- **Explaining familiarity effect in early visual areas and its relationship to that in higher visual areas**. While familiarity effect [6] is a well-known phenomenon in higher visual areas, recently our lab has found its existence in early visual areas (V2) as well [56]; interestingly, in that study, the familiarity effect in early visual areas emerged faster than that in higher ones. The results in that study suggest new theories of familiarity effect, which is commonly attributed to higher visual areas such as inferotemporal cortex (ITC). Existing models on familiarity effect [90, 87] do not address the familiarity effect in early visual areas or its faster emergence than higher visual areas.

References

[1] Adelson, E. H. and Bergen, J. R. (1985). Spatiotemporal energy models for the perception of motion. *J. Opt. Soc. Am. A*, 2(2):284–299.

[2] Anonymous (2019a). Contextual recurrent convolutional model for robust visual learning. In *Submitted to International Conference on Learning Representations*. under review.

[3] Anonymous (2019b). A model cortical network for spatiotemporal sequence learning and prediction. In *Submitted to International Conference on Learning Representations*. under review.

[4] Ba, J. and Caruana, R. (2014). Do deep nets really need to be deep? In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2654–2662.

[5] Ballé, J., Laparra, V., and Simoncelli, E. P. (2016). End-to-end optimized image compression. *CoRR*, abs/1611.01704.

[6] Baylis, G. C. and Rolls, E. T. (1987). Responses of neurons in the inferior temporal cortex in short term and serial recognition memory tasks. *Experimental Brain Research*, 65(3):614–622.

[7] Belhumeur, P. N. and Mumford, D. (1992). A Bayesian treatment of the stereo correspondence problem using half-occluded regions. In *1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 506–512. IEEE Comput. Soc. Press.

[8] Bell, A. J. and Sejnowski, T. J. (1997). The "independent components" of natural scenes are edge filters. *Vision Research*, 37(23):3327 – 3338.

[9] Bishop, C. M. (2006). *Machine learning and pattern recognition*. Information Science and Statistics. Springer.

[10] Bosking, W. H., Crowley, J. C., and Fitzpatrick, D. (2002). Spatial coding of position and orientation in primary visual cortex. *Nature Neuroscience*, 5(9):874–882.

[11] Cadena, S. A., Denfield, G. H., Walker, E. Y., Gatys, L. A., Tolias, A. S., Bethge, M., and Ecker, A. S. (2017). Deep convolutional models improve predictions of macaque v1 responses to natural images. *bioRxiv*.

[12] Cao, C., Liu, X., Yang, Y., Yu, Y., Wang, J., Wang, Z., Huang, Y., Wang, L., Huang, C., Xu, W., Ramanan, D., and Huang, T. S. (2015). Look and think twice: Capturing top-down visual attention with feedback convolutional neural networks. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 2956–2964. IEEE Computer Society.

[13] 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.

[14] Carandini, M. and Heeger, D. J. (2012a). Normalization as a canonical neural computation. *Nature Reviews Neuroscience*, 13(1):51–62.

[15] Carandini, M. and Heeger, D. J. (2012b). Normalization as a canonical neural computation. *Nature Reviews Neuroscience*, 13(1):51–62.

[16] Carandini, M., Heeger, D. J., and Movshon, J. A. (1997). Linearity and normalization in simple cells of the macaque primary visual cortex. *Journal of Neuroscience*, 17(21):8621–8644.

[17] Cavanaugh, J. R., Bair, W., and Movshon, J. A. (2002). Selectivity and spatial distribution of signals from the receptive field surround in macaque v1 neurons. *Journal of Neurophysiology*, 88(5):2547–2556. PMID: 12424293.

[18] Cheng, Y., Wang, D., Zhou, P., and Zhang, T. (2017). A survey of model compression and acceleration for deep neural networks. *CoRR*, abs/1710.09282.

[19] Chichilnisky, E. J. (2001). A simple white noise analysis of neuronal light responses. *Network: Computation in Neural Systems*, 12(2):199–213.

[20] Cho, K., van Merrienboer, B., Gülçehre, Ç., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078.

[21] Coen-Cagli, R., Dayan, P., and Schwartz, O. (2009). Statistical Models of Linear and Nonlinear Contextual Interactions in Early Visual Processing. In Bengio, Y., Schuurmans, D., Lafferty, J. D., Williams, C. K. I., and Culotta, A., editors, *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada.*, pages 369–377. Curran Associates, Inc.

[22] Coen-Cagli, R., Dayan, P., and Schwartz, O. (2012). Cortical Surround Interactions and Perceptual Salience via Natural Scene Statistics. *PLoS computational biology*, 8(3):e1002405.

[23] Coen-Cagli, R., Kohn, A., and Schwartz, O. (2015). Flexible gating of contextual influences in natural vision. *Nature Neuroscience*, 18(11):1648–1655.

[24] Cohen, M. R. and Maunsell, J. H. R. (2009). Attention improves performance primarily by reducing interneuronal correlations. *Nature Neuroscience*, 12(12):1594–1600.

[25] Costa, R. P., Assael, I. A. M., Shillingford, B., de Freitas, N., and Vogels, T. P. (2017). Cortical microcircuits as gated-recurrent neural networks. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 271–282.

[26] Daugman, J. G. (1985). Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *J. Opt. Soc. Am. A*, 2(7):1160–1169.

[27] David, S. V. and Gallant, J. L. (2005). Predicting neuronal responses during natural vision. *Network: Computation in Neural Systems*, 16(2-3):239–260.

[28] Dayan, P. and Abbott, L. F. (2001). *Theoretical Neuroscience*. Computational and Mathematical Modeling of Neural Systems.

[29] DeWeese, M. R., Wehr, M., and Zador, A. M. (2003). Binary spiking in auditory cortex. *Journal of Neuroscience*, 23(21):7940–7949.

[30] Doersch, C., Gupta, A., and Efros, A. A. (2015). Unsupervised visual representation learning by context prediction. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1422–1430. IEEE Computer Society.

[31] Doersch, C. and Zisserman, A. (2017). Multi-task self-supervised visual learning. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2070–2079. IEEE Computer Society.

[32] Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 647–655. JMLR.org.

[33] Donahue, J., Krähenbühl, P., and Darrell, T. (2016). Adversarial feature learning. *CoRR*, abs/1605.09782.

[34] Douglas, R. J. and Martin, K. A. C. (2007). Recurrent neuronal circuits in the neocortex. *Current Biology*, 17(13):R496–R500.

[35] Dugas, C., Bengio, Y., Bélisle, F., Nadeau, C., and Garcia, R. (2000). Incorporating second-order functional knowledge for better option pricing. In Leen, T. K., Dietterich, T. G., and Tresp, V., editors, *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*, pages 472–478. MIT Press.

[36] Duncan, J., Humphreys, G., and Ward, R. (1997). Competitive brain activity in visual attention. *Current Opinion in Neurobiology*, 7(2):255 – 261.

[37] Ecker, A. S., Berens, P., Keliris, G. A., Bethge, M., Logothetis, N. K., and Tolias, A. S. (2010). Decorrelated Neuronal Firing in Cortical Microcircuits. *Science*, 327(5965):584–587.

[38] Elder, J. H. and Goldberg, R. M. (2002). Ecological statistics of Gestalt laws for the perceptual organization of contours. *Journal of Vision*, 2(4):324–353.

[39] Felleman, D. J. and Van Essen, D. C. (1991). Distributed hierarchical processing in the primate cerebral cortex. *Cerebral Cortex*, 1(1):1–47.

[40] Field, D. J., Hayes, A., and Hess, R. F. (1993). Contour integration by the human visual system: Evidence for a local "association field". *Vision Research*, 33(2):173–193.

[41] Fleet, D. J., Wagner, H., and Heeger, D. J. (1996). Neural encoding of binocular disparity: Energy models, position shifts and phase shifts. *Vision Research*, 36(12):1839–1857.

[42] Ganguli, D. and Simoncelli, E. P. (2010). Implicit encoding of prior probabilities in optimal neural populations. In Lafferty, J. D., Williams, C. K. I., Shawe-Taylor, J., Zemel, R. S., and Culotta, A., editors, *Advances in Neural Information Processing Systems 23*, pages 658–666. Curran Associates, Inc.

[43] Ganmor, E., Segev, R., and Schneidman, E. (2011). Sparse low-order interaction network underlies a highly correlated and learnable neural population code. *Proceedings of the National Academy of Sciences*, 108(23):9679–9684.

[44] Geisler, W. S., Perry, J. S., Super, B. J., and Gallogly, D. P. (2001). Edge co-occurrence in natural images predicts contour grouping performance. *Vision Research*, 41(6):711–724.

[45] Geman, S. and Geman, D. (1984). Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741.

[46] Gollisch, T. and Meister, M. (2010). Eye smarter than scientists believed: Neural computations in circuits of the retina. *Neuron*, 65(2):150–164.

[47] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. C., and Bengio, Y. (2014). Generative adversarial networks. *CoRR*, abs/1406.2661.

[48] Granot-Atedgi, E., Tkačik, G., Segev, R., and Schneidman, E. (2013). Stimulus-dependent Maximum Entropy Models of Neural Population Codes. *PLoS computational biology*, 9(3):e1002922.

[49] Han, S., Mao, H., and Dally, W. J. (2015). Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. *CoRR*, abs/1510.00149.

[50] Hartline, H. K. (1938). 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.

[51] Haslinger, R., Pipa, G., Lima, B., Singer, W., Brown, E. N., and Neuenschwander, S. (2012). Context Matters: The Illusive Simplicity of Macaque V1 Receptive Fields. *PLoS ONE*, 7(7):e39699.

[52] Heeger, D. J. (1992). Half-squaring in responses of cat striate cells. *Visual Neuroscience*, 9(5):427–443.

[53] Hegdé, J. and Van Essen, D. C. (2007). A comparative study of shape representation in macaque visual areas v2 and v4. *Cerebral Cortex*, 17(5):1100–1116.

[54] Hinton, G. E. and Sejnowski, T. J. (1986). Learning and Relearning in Boltzmann Machines. In *Parallel Distributed Processing: Foundations*. MIT Press.

[55] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

[56] Huang, G., Ramachandran, S., Lee, T. S., and Olson, C. R. (2018). Neural correlate of visual familiarity in macaque area v2. *Journal of Neuroscience*.

[57] Huang, J., Lee, A. B., and Mumford, D. (2000). Statistics of range images. In *2000 IEEE Conference on Computer Vision and Pattern Recognition*, pages 324–331. IEEE Comput. Soc.

[58] Hubel, D. H. and Wiesel, T. N. (1959). Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, 148(3):574–591.

[59] 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.

[60] Hubel, D. H. and Wiesel, T. N. (1968). Receptive fields and functional architecture of monkey striate cortex. *The Journal of Physiology*, 195(1):215–243.

[61] Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Bach, F. R. and Blei, D. M., editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 448–456. JMLR.org.

[62] Jetley, S., Lord, N. A., Lee, N., and Torr, P. H. S. (2018). Learn to pay attention. *CoRR*, abs/1804.02391.

[63] Jones, J. P. and Palmer, L. A. (1987). An evaluation of the two-dimensional Gabor filter model of simple receptive fields in cat striate cortex. *Journal of Neurophysiology*, 58(6):1233–1258.

[64] Kapadia, M. K., Ito, M., Gilbert, C. D., and Westheimer, G. (1995). Improvement in visual sensitivity by changes in local context: Parallel studies in human observers and in V1 of alert monkeys. *Neuron*, 15(4):843–856.

[65] Kapadia, M. K., Westheimer, G., and Gilbert, C. D. (2000). Spatial distribution of contextual interactions in primary visual cortex and in visual perception. *Journal of Neurophysiology*, 84(4):2048–2062. PMID: 11024097.

[66] Kavukcuoglu, K., Sermanet, P., Boureau, Y. L., Gregor, K., Mathieu, M., and LeCun, Y. (2010). Learning Convolutional Feature Hierarchies for Visual Recognition. In Lafferty, J. D., Williams, C. K. I., Taylor, J. S., Zemel, R. S., and Culotta, A., editors, *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada.*, pages 1090–1098. Curran Associates, Inc.

[67] Kelly, R. C., Smith, M. A., Kass, R. E., and Lee, T. S. (2010a). Accounting for network effects in neuronal responses using L1 regularized point process models. In Lafferty, J. D., Williams, C. K. I., Shawe-Taylor, J., Zemel, R. S., and Culotta, A., editors, *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada.*, pages 1099–1107. Curran Associates, Inc.

[68] Kelly, R. C., Smith, M. A., Kass, R. E., and Lee, T. S. (2010b). Local field potentials indicate network state and account for neuronal response variability. *Journal of Computational Neuroscience*, 29(3):567–579.

[69] Khaligh-Razavi, S.-M. and Kriegeskorte, N. (2014). Deep supervised, but not unsupervised, models may explain it cortical representation. *PLOS Computational Biology*, 10(11):1–29.

[70] Kietzmann, T. C., McClure, P., and Kriegeskorte, N. (2017). Deep Neural Networks In Computational Neuroscience. *bioRxiv*, page 133504.

[71] Kindel, W. F., Christensen, E. D., and Zylberberg, J. (2017). Using deep learning to reveal the neural code for images in primary visual cortex. *ArXiv e-prints*, q-bio.NC.

[72] King, P. D., Zylberberg, J., and DeWeese, M. R. (2013). Inhibitory Interneurons Decorrelate Excitatory Cells to Drive Sparse Code Formation in a Spiking Model of V1. *Journal of Neuroscience*, 33(13):5475–5485.

[73] Klindt, D., Ecker, A. S., Euler, T., and Bethge, M. (2017). Neural system identification for large populations separating "what" and "where". In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 3509–3519.

[74] Knill, D. C. and Richards, W. (1996). *Perception as Bayesian inference*. Cambridge University Press.

[75] Koch, C., Marroquin, J., and Yuille, A. (1986). Analog "neuronal" networks in early vision. *Proceedings of the National Academy of Sciences*, 83(12):4263–4267.

[76] Kohn, A. and Coen-Cagli, R. (2015). Multi-electrode recordings of anesthetized macaque v1 responses to static natural images and gratings.

[77] Köster, U. and Olshausen, B. (2013). Testing our conceptual understanding of V1 function. *ArXiv e-prints*, q-bio.NC.

[78] Kotikalapudi, R. (2017). keras-vis: Keras visualization toolkit. https://github.com/raghakot/keras-vis.

[79] Kriegeskorte, N. (2015). Deep Neural Networks: A New Framework for Modeling Biological Vision and Brain Information Processing. *Annual Review of Vision Science*, 1(1):417–446.

[80] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Bartlett, P. L., Pereira, F. C. N., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, pages 1106–1114.

[81] Laparra, V., Berardino, A., Ballé, J., and Simoncelli, E. P. (2017). Perceptually optimized image rendering. *J. Opt. Soc. Am. A*, 34(9):1511–1525.

[82] Lee, H., Ekanadham, C., and Ng, A. Y. (2007). Sparse deep belief net model for visual area V2. In Platt, J. C., Koller, D., Singer, Y., and Roweis, S. T., editors, *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*, pages 873–880. Curran Associates, Inc.

[83] Li, G. and Zucker, S. W. (2010). Differential Geometric Inference in Surface Stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1):72–86.

[84] Li, M., Liu, F., Jiang, H., Lee, T. S., and Tang, S. (2017). Long-term two-photon imaging in awake macaque monkey. *Neuron*, 93(5):1049–1057.e3.

[85] Li, W. and Gilbert, C. D. (2002). Global Contour Saliency and Local Colinear Interactions. *Journal of Neurophysiology*, 88(5):2846–2856.

[86] Li, X., Jie, Z., Feng, J., Liu, C., and Yan, S. (2018). Learning with rethinking: Recurrently improving convolutional neural networks through feedback. *Pattern Recognition*, 79:183–194.

[87] Lim, S., McKee, J. L., Woloszyn, L., Amit, Y., Freedman, D. J., Sheinberg, D. L., and Brunel, N. (2015). Inferring learning rules from distributions of firing rates in cortical neurons. *Nature Neuroscience*, 18:1804–1810.

[88] Linsley, D., Kim, J., Veerabadran, V., and Serre, T. (2018). Learning long-range spatial dependencies with horizontal gated-recurrent units. *CoRR*, abs/1805.08315.

[89] Liu, Y., Bovik, A. C., and Cormack, L. K. (2008). Disparity statistics in natural scenes. *Journal of Vision*, 8(11):19–19.

[90] Lotter, W., Kreiman, G., and Cox, D. (2018). A neural network trained to predict future video frames mimics critical properties of biological neuronal responses and perception. *ArXiv e-prints*.

[91] Lotter, W., Kreiman, G., and Cox, D. D. (2016). Deep predictive coding networks for video prediction and unsupervised learning. *CoRR*, abs/1605.08104.

[92] Marr, D. and Poggio, T. (1976). Cooperative Computation of Stereo Disparity. *Science*, 194(4262):pp. 283–287.

[93] McCullagh, P. and Nelder, J. (1989). *Generalized Linear Models, Second Edition*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis.

[94] McFarland, J. M., Cui, Y., and Butts, D. A. (2013). Inferring Nonlinear Neuronal Computation Based on Physiologically Plausible Inputs. *PLoS computational biology*, 9(7):e1003143.

[95] McIntosh, L. T., Maheswaranathan, N., Nayebi, A., Ganguli, S., and Baccus, S. A. (2017). Deep Learning Models of the Retinal Response to Natural Scenes. *ArXiv e-prints*, q-bio.NC.

[96] Menz, M. D. and Freeman, R. D. (2003). Stereoscopic depth processing in the visual cortex: a coarse-to-fine mechanism. *Nature Neuroscience*, 6(1):59–65.

[97] Nayebi, A., Bear, D., Kubilius, J., Kar, K., Ganguli, S., Sussillo, D., DiCarlo, J. J., and Yamins, D. L. K. (2018). Task-driven convolutional recurrent models of the visual system. *CoRR*, abs/1807.00053.

[98] Olah, C., Mordvintsev, A., and Schubert, L. (2017). Feature visualization. *Distill*. https://distill.pub/2017/feature-visualization.

[99] Olah, C., Satyanarayan, A., Johnson, I., Carter, S., Schubert, L., Ye, K., and Mordvintsev, A. (2018). The building blocks of interpretability. *Distill*. https://distill.pub/2018/building-blocks.

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

[101] Olshausen, B. A. and Field, D. J. (2004). Sparse coding of sensory inputs. *Current Opinion in Neurobiology*, 14(4):481 – 487.

[102] O'Reilly, R., Wyatte, D., Herd, S., Mingus, B., and Jilk, D. (2013). Recurrent processing during object recognition. *Frontiers in Psychology*, 4:124.

[103] Paninski, L. (2004). Maximum likelihood estimation of cascade point-process neural encoding models. *Network: Computation in Neural Systems*, 15(4):243–262.

[104] Park, I. M., Archer, E., Priebe, N., and Pillow, J. W. (2013). Spectral methods for neural characterization using generalized quadratic models. In Burges, C. J. C., Bottou, L., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 2454–2462.

[105] Park, I. M. and Pillow, J. W. (2011). Bayesian Spike-Triggered Covariance Analysis. In Taylor, J. S., Zemel, R. S., Bartlett, P. L., Pereira, F. C. N., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain.*, pages 1692–1700.

[106] Pathak, D., Krähenbühl, P., Donahue, J., Darrell, T., and Efros, A. A. (2016). Context encoders: Feature learning by inpainting. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2536–2544. IEEE Computer Society.

[107] Pillow, J. W., Shlens, J., Paninski, L., Sher, A., Litke, A. M., Chichilnisky, E. J., and Simoncelli, E. P. (2008). Spatio-temporal correlations and visual signalling in a complete neuronal population. *Nature*, 454(7207):995–999.

[108] Poggio, G. F. and Fischer, B. (1977). Binocular interaction and depth sensitivity in striate and prestriate cortex of behaving rhesus monkey. *Journal of Neurophysiology*, 40(6):1392–1405.

[109] Poggio, G. F., Gonzalez, F., and Krause, F. (1988). Stereoscopic mechanisms in monkey visual cortex: binocular correlation and disparity selectivity. *The Journal of Neuroscience*, 8(12):4531–4550.

[110] Prazdny, K. (1985). Detection of binocular disparities. *Biological Cybernetics*, 52(2):93–99.

[111] Prenger, R., Wu, M. C. K., David, S. V., and Gallant, J. L. (2004). Nonlinear V1 responses to natural scenes revealed by neural network analysis. *Neural Networks*, 17(5-6):663–679.

[112] Rajaei, K., Mohsenzadeh, Y., Ebrahimpour, R., and Khaligh-Razavi, S.-M. (2018). Beyond core object recognition: Recurrent processes account for object recognition under occlusion. *bioRxiv*.

[113] Rao, R. P. N. and Ballard, D. H. (1999). Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 2(1):79–87.

[114] Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. (2014). Fitnets: Hints for thin deep nets. *CoRR*, abs/1412.6550.

[115] Rowekamp, R. J. and Sharpee, T. O. (2017). Cross-orientation suppression in visual area V2. *Nature Communications*, 8:15739.

[116] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252.

[117] Rust, N. C. and Movshon, J. A. (2005). In praise of artifice. *Nature Neuroscience*, 8(12):1647–1650.

[118] Rust, N. C., Schwartz, O., Movshon, J. A., and Simoncelli, E. P. (2005). Spatiotemporal Elements of Macaque V1 Receptive Fields. *Neuron*, 46(6):945–956.

[119] Samonds, J. M., Potetz, B. R., and Lee, T. S. (2009). Cooperative and Competitive Interactions Facilitate Stereo Computations in Macaque Primary Visual Cortex. *The Journal of Neuroscience*, 29(50):15780–15795.

[120] Samonds, J. M., Potetz, B. R., Tyler, C. W., and Lee, T. S. (2013). Recurrent Connectivity Can Account for the Dynamics of Disparity Processing in V1. *The Journal of Neuroscience*, 33(7):2934–2946.

[121] Schneidman, E., Berry, M. J., Segev, R., and Bialek, W. (2006). Weak pairwise correlations imply strongly correlated network states in a neural population. *Nature*, 440(7087):1007–1012.

[122] Schwartz, O. and Simoncelli, E. P. (2001). Natural signal statistics and sensory gain control. *Nature Neuroscience*, 4(8):819–825.

[123] Sclar, G., Maunsell, J. H., and Lennie, P. (1990). Coding of image contrast in central visual pathways of the macaque monkey. *Vision Research*, 30(1):1 – 10.

[124] Seriès, P., Lorenceau, J., and Frégnac, Y. (2003). The "silent" surround of V1 receptive fields: theory and experiments. *Journal of Physiology-Paris*, 97(4-6):453–474.

[125] Simoncelli, E. P. and Schwartz, O. (1998). Modeling Surround Suppression in V1 Neurons with a Statistically Derived Normalization Model. In Kearns, M. J., Solla, S. A., and Cohn, D. A., editors, *Advances in Neural Information Processing Systems 11, [NIPS Conference, Denver, Colorado, USA, November 30 - December 5, 1998]*, pages 153–159. The MIT Press.

[126] Simonyan, K. and Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. *ArXiv e-prints*, cs.CV.

[127] Skottun, B. C., Bradley, A., Sclar, G., Ohzawa, I., and Freeman, R. D. (1987). The effects of contrast on visual orientation and spatial frequency discrimination: a comparison of single cells and behavior. *Journal of Neurophysiology*, 57(3):773–786. PMID: 3559701.

[128] Smith, M. A. and Kohn, A. (2008). Spatial and Temporal Scales of Neuronal Correlation in Primary Visual Cortex. *The Journal of Neuroscience*, 28(48):12591–12603.

[129] Spoerer, C. J., McClure, P., and Kriegeskorte, N. (2017). Recurrent convolutional neural networks: A better model of biological object recognition. *Frontiers in Psychology*, 8:1551.

[130] Spratling, M. W. (2010). Predictive Coding as a Model of Response Properties in Cortical Area V1. *Journal of Neuroscience*, 30(9):3531–3543.

[131] Spratling, M. W. (2011). A single functional model accounts for the distinct properties of suppression in cortical area V1. *Vision Research*, 51(6):563–576.

[132] Spratling, M. W. (2012). Unsupervised Learning of Generative and Discriminative Weights Encoding Elementary Image Components in a Predictive Coding Model of Cortical Function. *Neural Computation*, 24(1):60–103.

[133] Spratling, M. W., De Meyer, K., and Kompass, R. (2009). Unsupervised Learning of Overlapping Image Components Using Divisive Input Modulation. *Computational Intelligence and Neuroscience*, 2009:1–19.

[134] Tang, S., Lee, T. S., Li, M., Zhang, Y., Xu, Y., Liu, F., Teo, B., and Jiang, H. (2018). Complex Pattern Selectivity in Macaque Primary Visual Cortex Revealed by Large-Scale Two-Photon Imaging. *Current Biology*, 28(1):38–48.e3.

[135] Tappen, M. F. and Freeman, W. T. (2003). Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters. In *Proceedings of the Ninth IEEE International Conference on Computer Vision*, pages 900–906.

[136] Theunissen, F. E., David, S. V., Singh, N. C., Hsu, A., Vinje, W. E., and Gallant, J. L. (2001). Estimating spatio-temporal receptive fields of auditory and visual neurons from their responses to natural stimuli. *Network: Computation in Neural Systems*, 12(3):289–316.

[137] Touryan, J., Felsen, G., and Dan, Y. (2005). Spatial Structure of Complex Cell Receptive Fields Measured with Natural Images. *Neuron*, 45(5):781–791.

[138] van Hateren, J. H. and Ruderman, D. L. (1998). Independent component analysis of natural image sequences yields spatio-temporal filters similar to simple cells in primary visual cortex. *Proceedings of the Royal Society of London B: Biological Sciences*, 265(1412):2315–2320.

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

[140] Victor, J. D., Mechler, F., Repucci, M. A., Purpura, K. P., and Sharpee, T. (2006). Responses of v1 neurons to two-dimensional hermite functions. *Journal of Neurophysiology*, 95(1):379–400.

[141] Vinje, W. E. and Gallant, J. L. (2000). Sparse coding and decorrelation in primary visual cortex during natural vision. *Science*, 287(5456):1273–1276.

[142] Vinje, W. E. and Gallant, J. L. (2002). Natural stimulation of the nonclassical receptive field increases information transmission efficiency in v1. *Journal of Neuroscience*, 22(7):2904–2915.

[143] Vintch, B., Movshon, J. A., and Simoncelli, E. P. (2015). A convolutional subunit model for neuronal responses in macaque v1. *Journal of Neuroscience*, 35(44):14829–14841.

[144] von Helmholtz, H. (1896). *Handbuch der physiologischen Optik*. Leipzig Voss.

[145] Wainwright, M. J. and Simoncelli, E. P. (1999). Scale Mixtures of Gaussians and the Statistics of Natural Images. In Solla, S. A., Leen, T. K., and Müller, K. R., editors, *Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999]*, pages 855–861. The MIT Press.

[146] Wang, H., Lin, X., Zhang, Y., and Lee, T. S. (2017a). Learning robust object recognition using composed scenes from generative models. *CoRR*, abs/1705.07594.

[147] Wang, J., Xie, C., Zhang, Z., Zhu, J., Xie, L., and Yuille, A. L. (2017b). Detecting semantic parts on partially occluded objects. *CoRR*, abs/1707.07819.

[148] Welling, M. and Hinton, G. E. (2002). A New Learning Algorithm for Mean Field Boltzmann Machines. In *International Conference on Artificial Neural Networks 2002*, pages 351–357.

[149] Wu, M. C. K., David, S. V., and Gallant, J. L. (2006). Complete Functional Characterization of Sensory Neurons by System Identification. *Annual Review of Neuroscience*, 29(1):477–505.

[150] Yamins, D., Hong, H., Cadieu, C. F., and DiCarlo, J. J. (2013). Hierarchical Modular Optimization of Convolutional Networks Achieves Representations Similar to Macaque IT and Human Ventral Stream. In Burges, C. J. C., Bottou, L., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 3093–3101.

[151] Yamins, D. L. K. and DiCarlo, J. J. (2016). Using goal-driven deep learning models to understand sensory cortex. *Nature Neuroscience*, 19(3):356–365.

[152] Zetzsche, C. and Barth, E. (1990). Fundamental limits of linear filters in the visual processing of two-dimensional signals. *Vision Research*, 30(7):1111 – 1117.

[153] Zetzsche, C. and Nuding, U. (2005). Nonlinear and higher-order approaches to the encoding of natural scenes. *Network: Computation in Neural Systems*, 16(2-3):191–221.

[154] Zetzsche, C. and Röhrbein, F. (2001). Nonlinear and extra-classical receptive field properties and the statistics of natural scenes. *Network: Computation in Neural Systems*, 12(3):331–350.

[155] Zhang, R., Isola, P., and Efros, A. A. (2016a). Colorful image colorization. In Leibe, B., Matas, J., Sebe, N., and Welling, M., editors, *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III*, volume 9907 of *Lecture Notes in Computer Science*, pages 649–666. Springer.

[156] Zhang, R., Isola, P., and Efros, A. A. (2017a). Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 645–654. IEEE Computer Society.

[157] Zhang, Y., Lee, T. S., Li, M., Liu, F., and Tang, S. (2018). Convolutional neural network models of v1 responses to complex patterns. *Journal of Computational Neuroscience*.

[158] Zhang, Y., Li, X., Samonds, J. M., and Lee, T. S. (2016b). Relating functional connectivity in v1 neural circuits and 3d natural scenes using boltzmann machines. *Vision Research*, 120:121 – 131. Vision and the Statistics of the Natural Environment.

[159] Zhang, Z., Xie, C., Wang, J., Xie, L., and Yuille, A. L. (2017b). Deepvoting: An explainable framework for semantic part detection under partial occlusion. *CoRR*, abs/1709.04577.

[160] Zhu, M. and Rozell, C. J. (2013). Visual Nonclassical Receptive Field Effects Emerge from Sparse Coding in a Dynamical System. *PLoS computational biology*, 9(8):e1003191.

[161] Zoph, B. and Le, Q. V. (2016). Neural architecture search with reinforcement learning. *CoRR*, abs/1611.01578.