Diverse task-driven modeling of macaque V4 reveals functional specialization towards semantic tasks

Santiago A. Cadena[1-3,7*], **Konstantin F. Willeke**[2-4], **Kelli Restivo**[5,6], **George Denfield**[5,6], **Fabian H. Sinz**[2,3-4,7†], **Matthias Bethge**[1,2,†], **Andreas S. Tolias**[5,6,†], **and Alexander S. Ecker**[7,8,†]

[1]Inst. for Theoretical Physics and Centre for Integrative Neuroscience, University of Tübingen, Germany
[2]Bernstein Center for Computational Neuroscience, Tübingen, Germany
[3]International Max Planck Research School for Intelligent Systems, Tübingen, Germany
[4]Institute for Bioinformatics and Medical Informatics, University Tübingen, Germany
[5]Center for Neuroscience and Artificial Intelligence, Baylor College of Medicine, Houston, TX, USA
[6]Department of Neuroscience, Baylor College of Medicine, Houston, TX, USA
[7]Institute of Computer Science and Campus Institute Data Science, University of Göttingen, Germany
[8]Max Planck Institute for Dynamics and Self-Organization, Göttingen, Germany
[†]Equal senior author contributions
[*]santiago.cadena@uni-tuebingen.de

ABSTRACT

Responses to natural stimuli in area V4 – a mid-level area of the visual ventral stream – are well predicted by features from convolutional neural networks (CNNs) trained on image classification. This result has been taken as evidence for the functional role of V4 in object classification. However, we currently do not know if and to what extent V4 plays a role in solving *other* computational objectives. Here, we investigated normative accounts of V4 by predicting macaque single-neuron responses to natural images from the representations extracted by 23 CNNs trained on different computer vision tasks including semantic, geometric, 2D, and 3D visual tasks. We found that semantic classification tasks do indeed provide the best predictive features for V4. Other tasks (3D in particular) followed very closely in performance, but a similar pattern of tasks performance emerged when predicting the activations of a network exclusively trained on object recognition. Thus, our results support V4's main functional role in semantic processing. At the same time, they suggest that V4's affinity to various 3D and 2D stimulus features found by electrophysiologists could be a corollary of a semantic functional goal.

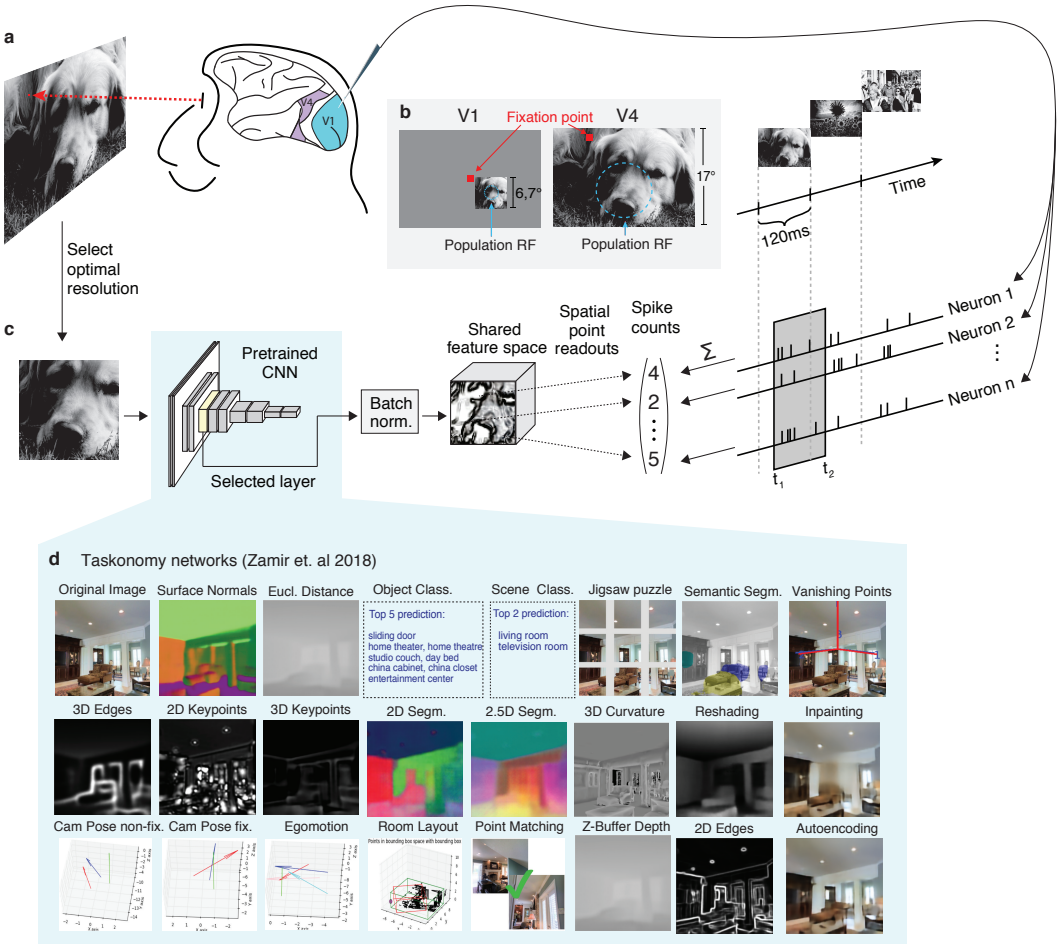

Figure 1. Experimental paradigm and diverse task modeling of neural responses. **a**, natural images were shown in sequence to two fixating rhesus macaques for 120ms while neural activity was recorded with a laminar silicon probe. Following careful spike-sorting, spike counts were extracted in time windows 40–160ms (V1) and 70–160ms (V4) after image onset. The screen covered $\sim 17^\circ \times 30^\circ$ of visual field with a resolution of $\sim 63\text{px}/^\circ$. For each area, we showed approx. 10k unique images once (*train set*). A random set of 75 images, identical for both areas, was repeated 40-55 times (*test set*). **b**, For V1 recordings (left), the fixation spot was centered on the screen. Square, gray-scale, 420px ImageNet images (6.7°) were placed at the center of each session's population receptive field (size: $\sim 2^\circ$, eccentricities: $2^\circ - 3^\circ$). In the V4 recordings sessions (right), the fixation spot was accommodated to bring the population receptive field as close to the center of the screen as possible (size: $\sim 8^\circ$, eccentricities: $8^\circ - 12^\circ$). All images were up-sampled and cropped to cover the whole screen. We isolated 458 (V1) and 255 (V4) neurons from 32 sessions of each area. **c**, Predictive model. Cropped input images covering 2.7° (V1) and 12° (V4) were resized and forwarded through the first l layers of a pretrained convolutional neural network (CNN) to produce features that are then batch-normalized and shared by all neurons. The input scale factor was a hyper-parameter, cross-validated on a held-out subset of the train set (*validation set*). The point readout[1] extracts features at a single spatial location and computes a regularized linear mapping to the neural responses for each neuron separately (see Methods). The readout and batch-normalization parameters were jointly learned to minimize the Poisson loss between predicted and observed response rates. **d**, *taskonomy* networks used for feature extraction (Adapted with permission from Zamir et al. (2018)[2]). We used the pretrained encoder CNNs of these tasks, which share a Resnet50 architecture[3], to build our models and compare their predictive abilities on V1 and V4.

Introduction

What is the functional role of area V4 in primate visual information processing? One line of evidence suggests that V4 is tuned in a high-dimensional space that facilitates the joint encoding of shape and surface characteristics of object parts[4,5] (e.g. sensitivity to luminance[6], texture[7] and chromatic contrasts[8], blurry boundaries[9], and luminance gradients[10]). Although very insightful, these experiments are constrained to a relatively small number of stimulus feature directions that potentially miss other important aspects of V4 function that could be unlocked with richer natural stimuli. Recent work showed that features extracted by convolutional neural networks (CNNs) trained on object classification provided strong predictive performance of V4 responses to natural stimuli[11,12]. This result has been interpreted as evidence that object recognition is one of the major goals of V4 processing. A natural question that arises is: Do other computational goals beyond object classification explain V4 responses equally well? Recent work using fMRI in humans has attempted to assign different functional goals to different regions of interest in the brain (including V4)[13–15], but it remains unclear whether single neurons express the same patterns of selectivity as the highly aggregated, indirect fMRI signal.

Here, we investigated how well the representations learned by CNNs trained on 23 different visual tasks from the *taskonomy* project [2] predict single-cell responses to natural images recorded in macaque area V4 and V1 for comparison. We found that a diverse set of tasks explained V1 responses almost equally well. In contrast, scene and object classification provided better accounts for V4 responses than all the alternative tasks tested, solidifying V4's semantic functional role. However several 3D tasks like estimating 3D keypoints or surface normals explained as much as 90% of the gap between an untrained baseline and the best semantic task in V4. To understand whether this amounts to evidence for multiple viable functional goals in the ventral stream beyond semantic tasks, we computed the pattern of predictive performances of the tasks' feature spaces on target representations from a separate network solely trained on object classification, and compared it to our results with target neural data. We found that the predictive patterns on V4 and the features of this network were strongly correlated. These results suggest that V4's strong similarity to the features from different tasks (e.g. 3D) result from its primary role as an intermediate step towards solving object recognition.

Results

We built upon the *taskonomy* project[2], a recent large-scale effort of the computer vision community, in which CNN architectures consisting of *encoder-decoder* parts were trained to solve various visual tasks. The *encoder* provides a low-dimensional representation of the input images from which each task can be (nonlinearly) read-out by the *decoder*. We considered the encoder network of 23 of these tasks, which have been previously categorized into *semantic*, *geometric*, *2D*, and *3D* groups (listed in Fig. 1d) based on hierarchical clustering of their encoder representations[2]. We chose these networks because of two key features: 1) all of them were trained on the same set of images, and 2) all encoder networks have the same architecture (ResNet-50[3]). Any differences we observe across the learned representations are thus caused by the training objective targeted to solve a specific task.

To quantify the match between the representations extracted by intermediate layers of the *taskonomy* networks and V4 representations, we used these networks to built task-driven models[16,17] of single-neuron recordings in response to natural stimuli: We presented the images that were shown to the monkey to each pretrained network, then extracted the resulting output feature maps from several intermediate layers and fed these to a regularized linear-nonlinear (LN) readout that was

specific to each recorded neuron (Fig. 1c). This readout acted on the features at a single spatial location [1], preventing any additional nonlinear spatial integration beyond what has been computed by the task-trained network. For each *taskonomy* network, we built one model for each readout layer and optimized regularization parameters for each model. To ensure that the resulting correspondence between network layers and neural data is not merely driven by the confound between growing receptive field sizes and feature complexity along the network's depth, we optimized the resolution of the input images on held-out data from the training set (Supplementary Fig. S1). This prevented us from assigning V4 to a layer simply because of the matching receptive field coverage that could result from an arbitrary input resolution, but instead allowed us to find for each model the layer with the best aligned nonlinearities to the data.

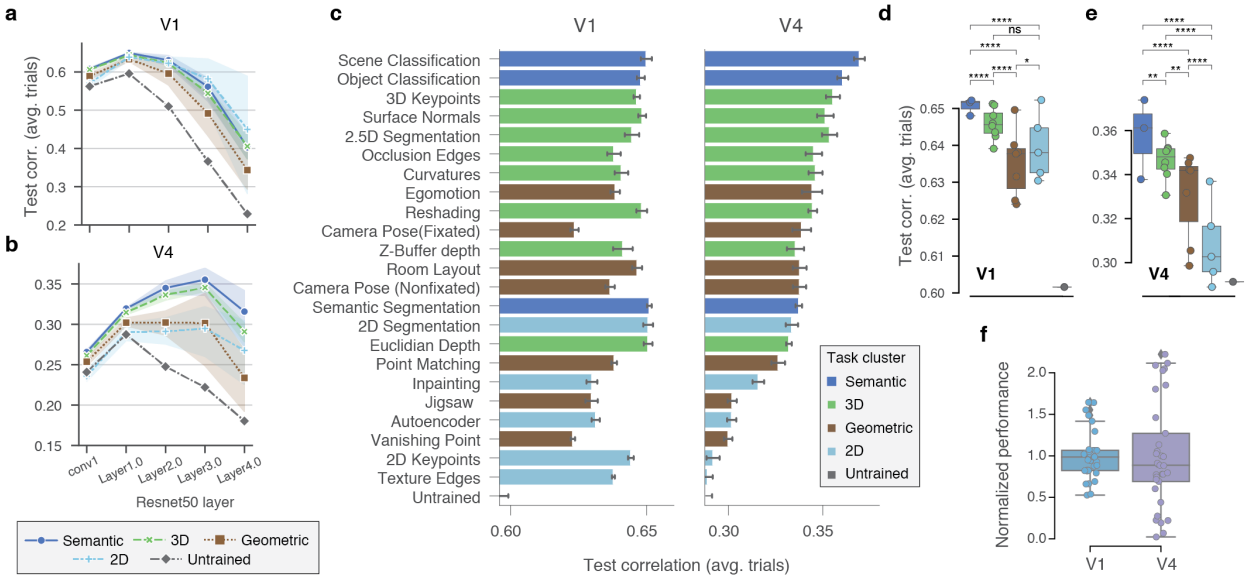

Figure 2. Comparison of diverse task-driven models on V1 and V4. **a,b**, Average performance of task clusters (see Fig. 1f) on V1 (**a**) and V4 (**b**) as a function of network layer. Error bands represent 95% confidence intervals across individual tasks (see Supplementary Fig. S2). Performance is measured as the average test-set correlation across neurons between model predictions and mean spike counts over repeated presentations. All tasks outperform an untrained, random baseline that shares the same architecture (dark gray). The best predictive features for V1 can be found at an early layer (Layer1.0) for all models, while deeper layers yielded peak performance for V4 with semantic and 3D task-clusters outperforming the others. **c**, The individual task-model performance on V1 (left) and V4 (right), maximized over layers and other hyper-parameters in the validation set. Error bars show 1 s.e. of the mean for five random initializations of the same model configuration. Task-models are sorted by performance on V4. Hues represent cluster types (inset). The baseline is the average performance of an untrained network. **d,e**, Comparisons between pairs of task-clusters. Individual task-models as circles, and box-plots depict their distribution for each task-cluster. The number of stars represent the *p-value* upper thresholds $(0.05, 10^{-2}, 10^{-3}, 10^{-3})$ of each comparison test between pairs of clusters. For each pair, we ran a pairwise Wilcoxon signed rank test between the group contrast performance of individual units $n = 458$ (V1), and $n = 255$ (V4), and applied Holm-Bonferroni correction to account for the six multiple comparisons. **f**, The normalized performance (increment over untrained baseline divided by the mean) for all individual tasks on V1 and V4. The variance across tasks in V4 is significantly different than in V1, rendering it more functionally specialized ($P = .0018$, $n = 23$, Levene's test for equality of variances)

We collected datasets of well-isolated single-cell responses from V4 and primary visual cortex (V1) for comparison. We measured the spiking activity of individual neurons in these areas from two awake, fixating rhesus macaques (M1, M2) using a 32-channel linear array spanning multiple cortical layers [17, 18], in response to tens of thousands of grayscale natural images presented in sequence over many trials (Fig. 1a). These images were sampled uniformly from the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC-2012) dataset[19] and displayed for 120 ms each without interleaving blanks (see Methods). Most of these images were shown only once (*train-set*) while a selection of 75 images was repeated multiple times (*test-set*). We isolated 458 V1 neurons from 15 (M1) and 17 (M2) sessions at eccentricities 2–3°; and 255 V4 neurons from 11 (M1) and 21 (M2) sessions at eccentricities 8–12°. For the V1 sessions, we centered the stimuli on the population receptive field of the neurons. For the V4 sessions, the stimuli covered the entire screen. We obtained image–response pairs by extracting spike counts in the windows 40–160 ms (V1) and 70–160 ms (V4) after image onset (Fig. 1a–c), which corresponded to the typical response latency of the neurons in the respective brain area. With these image–response pairs, we fitted our taskonomy-based task-driven models.

We evaluated the predictive performance of our fitted task-driven models and found consistent results to a body of prior neurophysiology work that corroborate our approach: First, when accounting for input scale (Supplementary S1), we found that the predictive performance on V4 peaks at a higher layer than V1 (Fig. 2a–b, Supplementary Fig. S2), a signature of hierarchical correspondence and increased complexity of V4 over V1 found in anatomical and latency studies[20]. This functional hierarchy was present in most cases, but could not be explained by the architecture alone: V1 and V4 were assigned to the same layer for networks trained on texture edges, jigsaw, vanishing points, and the untrained baseline (a network with matching architecture and random weights). Moreover, for several 2D tasks, the hierarchical assignment was not so strict as higher layers in the network retained high predictive power in comparison to other tasks. Second, we found that although V1's performance peaks at an early layer (Supplementary Fig. S2), it is still four linear-nonlinear steps away from input pixels, highlighting the nonlinear nature of its computations[17]. Finally, the optimal input resolutions (the image scale that leads to best performance) for V4 and V1 differed by roughly a factor of four (Fig. S3), which corresponds to the difference in average eccentricity of the neurons recorded in both areas (2–3° in V1 and 8–12° in V4) and roughly matches others' observations[21–24].

The two best predictive task-models on V4, after optimizing over layers, were the two semantic classification tasks: scene classification, and – consistent with prior work[11,12] – object classification (test correlation with averaged trials of 0.369 and 0.361; Fig. 2c; right). Although these two networks learned similar representations at their mid-level layers[14], the superior performance of the scene classification model suggests that its task-relevant stimulus cues that go beyond individual object parsing (e.g. detecting background colors or textures) are relevant to V4 function. Semantic segmentation, on the other hand, did not yield a high performance (0.337) in comparison. A possible explanation is that solving this pixel-wise task with the *taskonomy* architecture requires retaining deep into the network more precise information about the exact location of semantic boundaries than is retained in V4, which has been shown to exhibit translation invariance to the exact location of shape borders[5,25].

In contrast to V4, we found that V1's top predictive models were more diverse and not specifically tied to semantic-related tasks – the top five predictive models also came from 3D, and 2D task groups: semantic segmentation, 2D segmentation, euclidean depth, scene classification, and surface normals (Fig. 2c, left). We found that V4 responses were harder to predict than V1's with our task-driven representations. For instance, the mean top five model performance in V1 (0.650 ± 0.001) was higher

and less variable than in V4 (0.358 ± 0.006). However, the stimulus-driven variability between the two areas was not significantly different (explainable variances of 0.31 ± 0.18 (V1) and 0.32 ± 0.9 (V4); two-sided t-test, $t(711) = -1.152, P = .2$). This performance discrepancy could be partly explained by differences in sparsity – measured by the selectivity index (SI; see Methods) – between V1 and V4 responses (selectivity indices of 0.40 ± 0.23 [V1] and 0.56 ± 0.26 [V4]; two-sided t-test, $t(711) = -9.09, P = 9.49 \cdot 10^{-19}$) as the SI of individual neurons was negatively correlated to the performance yielded by a top model ($\rho = -0.35, P < 1e-8$ on V4 neurons). Nonetheless, we found that most models in both areas were non-trivially effective as they widely outperformed the untrained (weights randomly initialized) baseline – the two exceptions comparable to the baseline on V4 were texture edges and 2D keypoints (Fig. 2c). These results highlight key differences between primate and mouse visual processing: in the mouse, a similar approach revealed that the performance of untrained features is much closer to that of features trained for an image classification task[26].

How important is the specific task objective over the untrained baseline to obtain better predictive performances? We found that the choice of task – or task cluster – in V1 was less detrimental to changes in performance than in V4 (Fig. 2 e,d,f). We quantified this functional specialization by computing the variance of the performance of all task-models normalized to their mean (excluding the untrained network). The variance for V4 was higher than V1 ($P < .002$, $n = 23$, Levene's test; Fig. 2f), rendering it more specialized. This can be largely explained by the fact that V4's optimal layer is deeper than V1's as more nonlinear representations likely have more divergent representations. These results back the traditional notion of generality in the features extracted by V1, as they support multiple downstream tasks, while they also reveal the specialized role of V4 in visual processing.

To unveil trends that apply beyond individual tasks, but that are consistent for functionally related task-groups (i.e. task clusters semantic, geometric, 3D, 2D), we compared the average performances of each task cluster. We found that the specialized role of V4 towards semantic tasks was also prevalent at this coarser level: there were significant differences between all pairs of clusters (Fig. 2 e; Supplementary Fig. S2) and the order of clusters was semantic, 3D, geometric, and 2D. For V1, the semantic group also came on top, followed by 2D and 3D (no significant differences), and finally the Geometric cluster (Fig. 2 d).

Our comparisons at the individual and cluster levels support that semantic tasks drive representations that best match ventral visual areas in the brain, especially area V4. However, semantic tasks only outperformed the next high-predictive tasks by a relatively small margin (Fig. 2c, right), making it difficult to dismiss them altogether as computational goals of V4 function. Moreover, task-models that extract features relevant for 3D understanding directly followed object classification and reached $\sim 90\%$ of the gap between the untrained and scene classification networks: these tasks were 3D keypoints, 2.5D segmentation, and surface normals estimation (see Methods). The 3D keypoints task, for example, aims to find points of interest that could be reliably detected even if an object in the scene is observed from different perspectives, and then to extract local surface features at these points. Detecting 3D borders is essential to solve this task because these keypoints tend to be around object corners[27], likely capturing useful information for downstream invariant object recognition. Most classical approaches characterizing V4 function are mainly focused on its tuning properties to flat (2D) stimuli[5], but recent studies suggest that many cells in V4 are tuned to solid-shape properties as well[28]. In line with these findings, our results also support the affinity of V4 for 3D representations.

The fair success of other (3D) tasks at explaining V4 responses in addition to semantic tasks – coupled with their consistency with neurophysiologists' observations – raises the hypothesis

that there might be other computational goals that V4 – or the ventral stream at large – could be optimized to solve. To check if other tasks are capturing relevant aspects of V4's nonlinear responses that diverge from those capture by the semantic tasks, we replicated our previous analyses using all *taskonomy* networks to predict the outputs of the relevant layers of a model where object classification is the only ground-truth task driving the representations – namely, a Resnet50[3] trained on ImageNet[19] (see Methods). We chose it because ImageNet-trained networks have proven to be the most effective to predict V1 and V4 responses[17,29]. We found that the prediction patterns across tasks for V4 responses and Resnet50 (layer3.0) activations was strongly correlated (Fig. 3, right). This suggests that our observations of V4's affinity to 3D representations are largely explained by a single semantic task objective. On the other hand, the tasks predictions on Resnet50 activations (layer1.0) and V1 responses were only weakly correlated (Fig. 3, left). This result supports our previous observations highlighting V4's specialized role in extracting semantic information over V1's generality to support multiple downstream tasks.

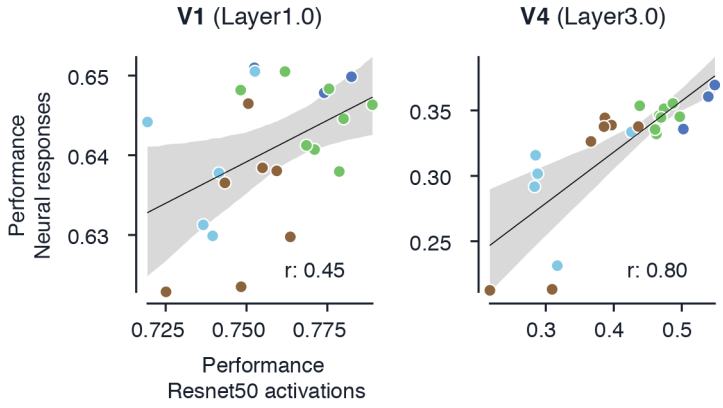

Figure 3. Predictive performance of task-model features on neural data vs. ImageNet-trained Resnet50. Each point in both panels represents a task-model and their task clusters are colored according to Fig. 2C. For V1 (left), we used Layer1.0 of the representations of the network to predict the very same layer of Resnet50 (x-axis). The correlation between the ability of these representations to predict V1 data and ImageNet features was weak ($r = 0.45$). On the other hand, for V4 (right) and using Layer3.0, we found a much stronger correlation ($r = 0.8$). These results suggest that for a semantically specialized area like V4, object recognition alone can largely explain the order and pattern of performances across tasks.

Discussion

We explored multiple normative accounts beyond object classification of V4 and V1 single neuron responses to natural stimuli using CNNs with matching architectures trained on the same dataset but different computer vision tasks. We found that semantic objectives drive representations that best match V4 and V1 responses, solidifying existing evidence provided by goal-driven models of the ventral stream using features pretrained for object classification. Although tasks related to 3D processing reached comparably good performances in V4, we found that their success is mainly a corollary of their similarity to semantic-driven representations. This implies that we can predict – to a large extend – V4's affinity to a candidate task-driven representation by evaluating its similarity to semantic representations. In particular, our work predicts a strong relationship between 3D visual processing and V4 function, shifting the long-standing focus of V4's functional role in flat shape processing, and supporting recent discoveries showing that V4 and early layers of CNNs

trained on object classification exhibit solid shape coding[28]. In addition to probing similarity to semantic-trained networks, our work suggests that we can use CNNs trained on object or scene classification to probe multiple stimulus features *in-silico* – as opposed to the expensive *in-vivo* alternatives – to find tuning directions that are relevant for the neurons.

In contrast to similar efforts applied to measurements of aggregated neural activity (i.e. fMRI data) from multiple visual areas[13,30], we evaluated the *intermediate* representations of *taskonomy* networks – not just their final layers – to evaluate their affinities to single-cell responses. We indeed found that intermediate layers provided the best match to the data – regardless of the task – and that diverse tasks offered comparable top performances in V1 and mostly semantic tasks were optimal in V4. Recent goal-driven efforts to characterize ventral stream responses (including V1 and V4) have found that unsupervised objectives lead to comparable predictive performance to object classification[31]. Although these self-supervised methods contain no explicit semantic training objective, their inherent data augmentation strategies enforce similar invariances as required for object recognition[32] and their learned representations predict semantic labels well[33]. Thus, it is to some extent expected that self-supervised methods fare similar to the object classification task.

Taken together, our results provide evidence for semantic tasks as a normative account of area V4 and explain why particular tasks (e.g. 3D) have strong affinities to V4 representations. Promising directions to improve the predictive power of V4 responses require training with richer and larger datasets and exploring different architectures.

Methods

Ethics statement

All behavioral and electrophysiological data were obtained from two healthy, male rhesus macaque (*Macaca mulatta*) monkeys aged 15 and 16 years and weighing 16.4 and 9.5 kg, respectively, during the time of study. All experimental procedures complied with guidelines of the NIH and were approved by the Baylor College of Medicine Institutional Animal Care and Use Committee (permit number: AN-4367). Animals were housed individually in a large room located adjacent to the training facility, along with around ten other monkeys permitting rich visual, olfactory and auditory interactions, on a 12h light/dark cycle. Regular veterinary care and monitoring, balanced nutrition and environmental enrichment were provided by the Center for Comparative Medicine of Baylor College of Medicine. Surgical procedures on monkeys were conducted under general anesthesia following standard aseptic techniques. To ameliorate pain after surgery, analgesics were given for seven days. Animals were not sacrificed after the experiments.

Electrophysiological recordings

We performed non-chronic recordings using a 32-channel linear silicon probe (NeuroNexus V1x32-Edge-10mm-60-177). The surgical methods and recording protocol were described previously[18]. Briefly, form-specific titanium recording chambers and headposts were implanted under full anesthesia and aseptic conditions. The bone was originally left intact and only prior to recordings, small trephinations (2 mm) were made over medial primary visual cortex at eccentricities ranging from 1.4 to 3.0 degrees of visual angle. Recordings were done within two weeks of each trephination. Probes were lowered using a Narishige Microdrive (MO-97) and a guide tube to penetrate the dura. Care was taken to lower the probe slowly, not to penetrate the cortex with the guide tube and to minimize tissue compression).

Data acquisition and spike sorting

Electrophysiological data were collected continuously as broadband signal (0.5Hz–16kHz) digitized at 24 bits. Our spike sorting methods mirror those in[17,18], code available at https://github.com/aecker/moksm. We split the linear array of 32 channels into 14 groups of 6 adjacent channels (with a stride of two), which we treated as virtual electrodes for spike detection and sorting. Spikes were detected when channel signals crossed a threshold of five times the standard deviation of the noise. After spike alignment, we extracted the first three principal components of each channel, resulting in an 18-dimensional feature space used for spike sorting. We fitted a Kalman filter mixture model to track waveform drift typical for non-chronic recordings[34,35]. The shape of each cluster was modeled with a multivariate t-distribution ($df = 5$) with a ridge-regularized covariance matrix. The number of clusters was determined based on a penalized average likelihood with a constant cost per additional cluster[36]. Subsequently, we used a custom graphical user interface to manually verify single-unit isolation by assessing the stability of the units (based on drifts and health of the cells throughout the session), identifying a refractory period, and inspecting the scatter plots of the pairs of channel principal components.

Visual stimulation and eye tracking

Visual stimuli were rendered by a dedicated graphics workstation and displayed on a 16:9 HD widescreen LCD monitor (23.8”) with a refresh rate of 100 Hz at a resolution of 1920×1080 pixels and a viewing distance of 100 cm (resulting in $\sim 63 px/^\circ$). The monitors were gamma-corrected to have a linear luminance response profile. A camera-based, custom-built eye tracking system verified that monkeys maintained fixation within $\sim 0.95^\circ$ around a $\sim 0.15^\circ$-sized red fixation target. Offline analysis showed that monkeys typically fixated much more accurately. After monkeys maintained fixation for 300 ms, a visual stimulus appeared. If the monkeys fixated throughout the entire stimulus period, they received a drop of juice at the end of the trial.

Receptive field mapping and stimulus placing

We mapped receptive fields relative to a fixation target at the beginning of each session with a sparse random dot stimulus. A single dot of size 0.12° of visual field was presented on a uniform gray background, changing location and color (black or white) randomly every 30 ms. Each fixation trial lasted for two seconds. We obtained multi-unit receptive field profiles for every channel using reverse correlation. We then estimated the population receptive field location by fitting a 2D Gaussian to the spike-triggered average across channels at the time lag that maximizes the signal-to-noise-ratio. During V1 recordings, we kept the fixation spot at the center of the screen and centered our natural image stimulus at the mean of our fit on the screen (Fig. 1b). During V4 recordings, the natural image stimulus covered the entire screen. We accommodated the fixation spot so that the mean of the population receptive field was as close to the middle of the screen as possible. Due to the location of our recording sites in both monkeys, this equated to locating the fixation spot close to the upper border of the screen, shifted to the left (Fig. 1c).

Natural image stimuli

We sampled a set of 24075 images from 964 categories (~ 25 images per category) from ImageNet[37], converted them to gray-scale, and cropped them to keep the central 420×420px. All images had 8 bit intensity resolution (values in [0,255]). We then sampled 75 as our *test-set*. From the remaining 24000 images, we sampled 20% as *validation-set*, leaving 19200 as *train-set*. We used the same sets of images for V1 and V4 recordings. During a recording session, we recorded ~ 1000 successful

trials, each consisting of uninterrupted fixation for 2.4 seconds including 300ms of gray screen (128 intensity) at the beginning and end of the trial, and 15 images shown consecutively for 120ms each with no blanks in between. Each trail contained either train and validation, or test images. We randomly interleaved trials throughout the session so that our test-set images were shown 40-50 times. The train and validation images were sampled without replacement throughout the session, so each train / validation image was effectively shown once or not at all. In V1 sessions, the images were shown at their original resolution and size covering 6.7° (screen resolution of 63 degrees per visual angle). The rest of the screen was kept gray (128 intensity). In V4 sessions, the images were upscaled preserving their aspect ratio with bicubic interpolation to match the width of the screen (1920px). We cropped out the upper and bottom 420px bands to cover the entire screen. As a result, we effectively stimulated both the classical and beyond the classical receptive fields of both areas. Once the neurons were sorted, we counted the spikes associated to each image presentation in a specific time window following the image onset. These windows were 40-160ms (V1) and 70-160ms (V4).

Explainable variance

We estimated the fraction of the stimulus-driven variability by computing the ratio between the total variance of the flattened responses minus the variance of the observation noise, over the total variance (Eq. 1). We estimated the variance of the observation noise by averaging across images the variance across repetitions of responses: $\sigma^2_{noise} = \mathrm{E}_j[\mathrm{Var}_t[r_t|x_j]]$ where t indexes the trials (repeats) and x_j represents a unique image.

$$\mathrm{EV} = \frac{\mathrm{Var}[r] - \sigma^2_{noise}}{\mathrm{Var[r]}} \tag{1}$$

Measuring Sparseness

We computed the selectivity index[38] (*SI*) as a measure of sparseness for every neuron on its test-set average responses. To do this, we first plotted the fraction of images whose responses where above a threshold, as a function of normalized thresholds. We considered 100 threshold bins ranging from the minimum to the maximum response values. The area under this curve (A) is close to zero for sparse neurons, and close to 0.5 for a uniform distribution of responses. Thus, following Quiroga et. al (2007)[38], we computed the selectivity index as $SI = 1 - 2A$. *SI* approaches 0 for a uniform distribution, and 1 the sparser the neuron is. In this study, we reported the mean and standard deviation of *SI* for both brain areas and found sparser responses in V4 than in V1 (see Results).

Image preprocessing and resizing

An important step of our modeling pipeline was to adjust the size and resolution of the input images to our computational models (Supplementary Fig. S1). In V1, we effectively cropped the central 2.65° (167px) at its original 63px/° resolution and downsampled with bicubic interpolation to different target resolutions: 3.5, 7.0, 14, 21, 24.5, and 28px/°. For practical and legacy reasons[17], in our codebase we first downsampled the images to a resolution of 35px/°, followed by cropping and another downsampling step to obtained the target sizes and resolutions just reported. In V4, we cropped the images up to the bottom central 12°, corresponding to 168px at the original 14px/° resolution (63px/° × 420/1920), in accordance with the neuron's RF positions. These images were similarly downsampled to multiple target resolutions: 1.4, 2.8, 5.6, 8.4, and 11.2px/°.

Model architecture

Our models of cell responses consisted of two main parts: A pretrained core that outputs nonlinear features of input images, and a *spatial point readout*[1] that maps these features to each neuron's responses. We built separate model instances for each visual area, input image resolution, task-dependent pretrained CNN, intermediate convolutional layer, regularization strength, and random initialization. Input images $\mathbf{x}$ were forwarded through all layers up to the chosen layer l, to output a tensor of feature maps $l(\mathbf{x}) \in \mathbb{R}^{w \times h \times c}$ (**w**idth, **h**eight, **c**hannels). Importantly, the parameters of the pretrained network were always kept fixed. We then applied batch-normalization[39] (Eq. 2), with trainable parameters for scale (γ) and shift (β), and running statistics mean (μ) and standard deviation (σ). These parameters were held fixed at test time (i.e. when evaluating our model). Lastly, we rectified the resulting tensor to obtain the final nonlinear feature space ($\Phi(\mathbf{x})$) shared by all neurons, with same dimensions as l. The normalization of CNN features ensured that the activations of each feature map (channel) have zero mean and unit variance (before rectification), facilitating meaningfully regularized readout weights for all neurons with a single penalty – having input features with different variances would implicitly apply different penalties on their corresponding readout weights.

$$\mathrm{BN}(x) = \gamma \cdot \frac{x - \mu}{\sigma} + \beta \tag{2}$$

The goal of the readout was to find a linear-nonlinear mapping from $\Phi(\mathbf{x})$ to a single scalar firing rate for every neuron. Previous approaches have attempted to 1) do dimensionality reduction on this tensor and regress from this components (e.g. partial least squares)[11]; 2) learn a dense readout with multiple regularization penalties over space and features[17]; and 3) factorize the 3D readout weights into a lower-dimensional representation consisting of a spatial mask matrix and a vector of feature weights[40]. In this work we used the recently proposed spatial point readout[1] – also called *Gaussian readout* by the authors – that goes a step further and restricts the spatial mask to a single point. Per neuron, it computes a linear combination of the feature activations at a spatial position, parametrized as (x, y) relative coordinates (the middle of the feature map being $(0, 0)$). Training this readout poses the challenge of maintaining gradient flow when optimizing the objective function. In contrast to previous approaches that tackle this challenge by recreating multiple subsampled versions of the feature maps and learn a common relative location for all of them[41], the *Gaussian readout* learns the parameters of a 2D Gaussian distribution $\mathcal{N}(\mu_n, \Sigma_n)$ and samples a location during each training step for every n^{th} neuron. Σ_n is initialized large enough to ensure gradient flow, and is then shrunken during training to have a more reliable estimate of the mean location μ_n. At inference time (i.e. when evaluating our model), the readout is deterministic and uses position μ_n. Although this framework allows for rotated and elongated Gaussian functions, we found that for our monkey data, an isotropic formulation of the covariance – parametrized by a single scalar σ_n^2 – was sufficient (i.e. offer similar performance as the fully parametrized Gaussian). Thus, the total number of parameters per neuron of the readout were $c + 4$ (channels, bivariate mean, variance, and bias). Finally, the resulting dot product between the features of $\Phi(\mathbf{x})$ at the chosen location with an L_1 regularized weight vector $\mathbf{w}_n \in \mathbb{R}^c$ was then followed by f, a point-wise nonlinear function ELU[42] offset by one ($\mathrm{ELU} + 1$) to make responses positive (Eq. 3).

$$\hat{r}_n(\mathbf{x}) = f\left(\sum_k \Phi_{\mu_{n,x}, \mu_{n,y}, k}(\mathbf{x}) w_{n,k} + b_n\right) \tag{3}$$

Beyond offering competitive performance compared to the factorized readout alternative with far less parameters, the most important motivation to use a single point readout was to make sure that all spatial nonlinear computations happen in the pretrained core feature extractor. We could draw mistaken claims about the nonlinear power of a feature space by computing new ones in the readout that combine rectified features computed at different spatial positions. For example, a readout that rectifies features produced by multiple simple cells with similar orientation at different locations can easily approximate phase invariance (i.e. complex cells)[43].

Model training

We trained every model to minimize the summed Poisson loss across N neurons between observed spike counts r and our predicted spike rate $\hat{r}$ (Eq. 4, first term) in addition to the L_1 regularization of the weights (Eq. 4, second term) with respect to the batch-normalization, and readout parameters.

$$\mathscr{L} = \sum_{i=1}^{N} (\hat{r}_n - r_n \log \hat{r}_n) + \lambda \sum_{n,k} |w_{nk}| \tag{4}$$

Since neurons across session from the same visual area didn't necessarily *see* the same images (they were differently drawn over sessions), during each training step, we cycled through all sessions of the same visual area, sampling for each of them a fixed batch size of image-response pairs without replacement and kept track of the gradients of the loss with respect of our trainable parameters. Once a cycle was through, the gradients were added to execute an update of the weights of the weights based on the Adam optimizer[44] – an improved version of stochastic gradient descent. The initial learning rate was $3 \cdot 10^{-4}$ and momentum 0.1. We continued to exhaust image-response pair batches from all sessions until the longest session was exhausted to count a full epoch. Once all image-response pairs had been drawn from a session, we restarted sampling batches from all available image-response pairs.

Every epoch, we temporarily switched our model into evaluation mode (i.e. we froze the batch-normalization running statistics), and computed the Poisson loss on the entire single trial *validation-set*. We then used early stopping to decide whether to decay the learning rate: we scaled the learning rate by a factor of 0.3 once the validation loss did not improve over five consecutive epochs. Before decaying the learning rate, we restored the weights to the best ones up to that point (in terms of validation loss). We ran the optimization until four early stopping steps were completed. On average, this resulted in ~ 50 training epochs (or ~ 40 minutes on one of our GPUs) per model instance.

Taskonomy networks

The *taskonomy* networks[2] are encoder-decoder CNNs trained on multiple computer vision tasks. The original goal of the authors was to identify a taxonomy of tasks that would facilitate efficient transfer learning based on the encoder representations of these networks. Importantly for our study, all these networks were trained by the authors on the same set of images, which have labels for all tasks. These images consisted of 120k indoor room scenes. In this work, we used the encoder architecture of these networks, which was based on a slightly modified version of Resnet50[3] that excluded average-pooling, and replaced the last stride 2 convolution with stride 1. However, these modifications did not change the number of output features of the intermediate layers we considered, keeping our *taskonomy* -based results fairly comparable with those of the original Resnet50.

The Resnet50 architecture – originally developed to solve ImageNet[37] – is made up of a series of hierarchical stages that include 1) an initial strided convolutional layer (`conv1`) followed by batch normalization, rectification, and max-pooling; 2) four processing layers, with 3,4,6, and 3 residual blocks, respectively; and 3) a final average pooling layer that maps features to the number of classes. Each residual block amounts to the rectified sum of two pathways: one that simply projects the input to the end (i.e. skip connection), and a second that consists of three successive convolutional layers with sizes 1, 3, and 1. In this work we trained models on the output of the first convolutional layer (`conv1`), and the output of the first residual block of each processing layer (i.e. `layer1.0`, `layer2.0`, `layer3.0`, `layer4.0`). The corresponding number of output feature maps (channels) for these layers was 64, 256, 512, 1024, and 2048, respectively.

We used several *taskonomy* encoder networks, listed in (Fig. 1c). The structure of the representations of these networks was presented by the authors via a metric of similarity across tasks: with agglomerative clustering of the tasks based on their transferring-out behavior, they built a hierarchical tree of tasks (see Figure 13 of their paper[2]). They found that the tasks can be grouped into 2D, 3D, low dimensional geometric, and semantic tasks based on how close (i.e. how similar) they are on the tree. We now briefly describe the tasks (for more details, see Supplementary Material from[2]):

2D tasks. *Autoencoding* PCA finds a low-dimensional latent representation of the data. *Edge Detection* responds to changes in texture. It is the output of a Canny edge detector without nonmax suppression to enable differentiation. *Inpainting* reconstructs missing regions in an image. *Keypoint Detection(2D)* both detects locally important regions in an image (keypoints), and extracts descriptive features of them that are invariant across multiple images. The output of SURF[45] was the ground-truth output of this task. *Unsupervised 2D Segmentation* uses as ground-truth the output of Normalized cuts[46] which tries to segment images into perceptually similar groups.

3D tasks. *Keypoint Detection (3D)* are like the 2D counterpart, but derived from 3D data, accounting for scene geometry. The output of the NARF algorithm[27] was the ground-truth output of this task. *Unsupervised 2.5D Segmentation* uses the same algorithm as 2D, but the labels are not only computed from RGB image, but also jointly from aligned depth, and surface normal images. It thus has access to ground-truth 3D information. *Surface Normal Estimation* are trained directly on the ground-truth surface normal vectors of the 3D meshes of the scene. *Curvature Estimation* extracts principal curvatures at each fix point of the mesh surface. *Edge Detection (3D)* (Occlusion Edges) are the edges where an object in the foreground obscures the background. It depends on 3D geometry and it is invariant to changes in color and lighting. In *Reshading*, the label for an RGB image is the shading function that results from having a single light point at the camera origin, multiplied by a constant albedo (amount of diffuse reflection of light radiation). *Depth Estimation, Z-Buffer*. *Depth Estimation, Euclidian* measures the distance between each pixel to the camera's optic center.

Geometric tasks. *Relative Camera Pose Estimation, Non-Fixated* predicts the relative six degrees of freedom (yaw, pitch, roll, x, y, z) of the camera pose between two different views with same optical centers. *Relative Camera Pose Estimation, Fixated* is a simpler variant of the previous one where the center pixel of the two inputs is always the same physical 3D point – yielding only five degrees of freedom. *Relative Camera Pose Estimation, Triplets (Egomotion)* matches camera poses for input triplets with a fixed center point. *Room Layout Estimation* estimates and aligns 3D bounding boxes around parts of the scene. *Point Matching* learns useful local feature descriptors that

facilitate matching scene points across images. *Content Prediction (Jigsaw)* unscrambles a permuted tiling of the image. *Vanishing Point Estimation* predicts the analytically computed vanishing points corresponding to an x, y, and z axis.

Semantic tasks. *Object Classification* uses knowledge distillation from a high-performing network trained on ImageNet[37] where its activations serve as a supervised signal (within the manually selected 100 object classes appearing in the *taskonomy* dataset). *Scene Classification* follows a similar approach, but uses a network trained on MITPlaces[47] with 63 applicable indoor workplace and home classes for supervised annotation of the dataset. *Semantic Segmentation* also follows the same supervised annotation procedure using a network trained on COCO[48] dataset with 17 applicable classes.

Finally, we included a control with matching architecture (Resnet50) but with random initialization.

Model configurations

In this study, we fitted a large set of task-models ($> 10,000$) that include all viable combinations of 1) brain areas (2: V1, V4); 2) input resolutions (5); 3) pretrained CNNs (23 *taskonomy* , 1 random); 4) intermediate convolutional layers (5), 5) L_1 regularization strengths (1-3 for most models); and 6) random initialization (5 seeds). Because of the receptive field size of higher layers in all networks, only large enough input resolutions were permitted in those cases. We used only 1-3 regularization penalties for most model configurations because we found that the optimal parameters from a fine-grained search of a single model where also appropriate for the corresponding layers of the *taskonomy* networks – actual optimal penalties led to negligibly differences in validation performance (within the noise of random seeds). The specific values we cross-validated over (after the fine-grained search) were: $\lambda_{\text{conv1}} = \{0.33, 1, 3\}, \lambda_{\text{layer1.0}} = \{3\}, \lambda_{\text{layer2.0}} = \{3, 6\}, \lambda_{\text{layer3.0}} = \{3, 9\}, \lambda_{\text{layer4.0}} = \{6, 12\}$.

Performance evaluation

The purpose of our work was to thoroughly compare the effectiveness of diverse task representations at predicting V1 and V4, rather than to establish a stat-of-the-art performance of the two visual areas. We thus simply computed the correlation between a model's predictions with the average response over multiple presentations. Any potential method to normalize these values according to estimates of the trial-to-trial variability would similarly scale these measures and not affect the ranking of the models.

Linear prediction of Resnet50 activations

We extracted the outputs of layers Layer1.0 and Layer3.0 of Resnet50[3] trained on ImageNet[19] to all of our training and testing images and kept all features (depth) at the center spatial point of these tensors. We did the same for all *taskonomy* networks. We then used standard linear regression using the output features for training images (L1 and L2 regularizations did not change results much) to predict the Resnet50 activations from each of the representations from the corresponding layer of the *taskonomy* networks. We then evaluated this fits on the test set responses by computing the correlation between true Resnet50 activations and those predicted by linearly reading out from each *taskonomy* network.

Data availability

The datasets of V1 and V4 recordings will be made available at (https://gin.g-node.org/) upon publication.

Code availability

Our coding framework uses general tools like Pytorch[49], Numpy[50], scikit-image[51], matplotlib[52], seaborn[53], DataJoint[54], Jupyter[55], and Docker[56]. We also used the following custom libraries and code: `neuralpredictors` (https://github.com/sinzlab/neuralpredictors) for torch-based custom functions for model implementation, `nnfabrik` (https://github.com/sinzlab/nnfabrik) for automatic model training pipelines using DataJoint, `nnvision` (https://github.com/sinzlab/nnvision) for specific model definitions, `ptrnets` (https://github.com/sacadena/ptrnets) for readily available pretrained CNNs and access to their intermediate layers. Example code to train models on our data will be made available upon publication.

Acknowledgements

The research was supported by the German Federal Ministry of Education and Research (BMBF) via the Competence Center for Machine Learning (FKZ 01IS18039A); the German Research Foundation (DFG) grant EC 479/1-1 (A.S.E.), the Collaborative Research Center (SFB 1233, Robust Vision) and the Cluster of Excellence "Machine Learning – New Perspectives for Science" (EXC 2064/1, project number 390727645); the Bernstein Center for Computational Neuroscience (FKZ 01GQ1002); the National Eye Institute of the National Institutes of Health under Award Numbers R01EY026927 (A.S.T.), DP1 EY023176 (A.S.T.), and NIH-Pioneer award DP1-OD008301 (A.S.T), and the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/Interior Business Center (DoI/IBC) contract number D16PC00003. This work was also supported by the National Institute of Mental Health under Award Number T32EY00252037. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/IBC, or the U.S. Government. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

References

1. Lurz, K.-K. *et al.* Generalization in data-driven models of primary visual cortex. *bioRxiv* (2020).
2. Zamir, A. R. *et al. Taskonomy: Disentangling task transfer learning* in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), 3712–3722.
3. He, K., Zhang, X., Ren, S. & Sun, J. *Deep residual learning for image recognition* in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), 770–778.
4. Pasupathy, A., Popovkina, D. V. & Kim, T. Visual functions of primate area V4. *Annual Review of Vision Science* **6,** 363–385 (2020).

5. Pasupathy, A. & Connor, C. E. Shape representation in area V4: position-specific tuning for boundary conformation. *Journal of neurophysiology* **86,** 2505–2519 (2001).

6. Bushnell, B. N., Harding, P. J., Kosai, Y., Bair, W. & Pasupathy, A. Equiluminance cells in visual cortical area V4. *Journal of Neuroscience* **31,** 12398–12412 (2011).

7. Kim, T., Bair, W. & Pasupathy, A. Neural coding for shape and texture in macaque area V4. *Journal of Neuroscience* **39,** 4760–4774 (2019).

8. Conway, B. R., Moeller, S. & Tsao, D. Y. Specialized color modules in macaque extrastriate cortex. *Neuron* **56,** 560–573 (2007).

9. Oleskiw, T. D., Nowack, A. & Pasupathy, A. Joint coding of shape and blur in area V4. *Nature communications* **9,** 1–13 (2018).

10. Hanazawa, A. & Komatsu, H. Influence of the direction of elemental luminance gradients on the responses of V4 cells to textured surfaces. *Journal of Neuroscience* **21,** 4490–4497 (2001).

11. Yamins, D. L. *et al.* Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the national academy of sciences* **111,** 8619–8624 (2014).

12. Pospisil, D. A., Pasupathy, A. & Bair, W. 'Artiphysiology'reveals V4-like shape tuning in a deep network trained for image classification. *Elife* **7,** e38242 (2018).

13. Wang, A., Tarr, M. & Wehbe, L. *Neural Taskonomy: Inferring the Similarity of Task-Derived Representations from Brain Activity* in *Advances in Neural Information Processing Systems* (eds Wallach, H. *et al.*) **32** (Curran Associates, Inc., 2019). https://proceedings.neurips.cc/paper/2019/file/f490c742cd8318b8ee6dca10af2a163f-Paper.pdf.

14. Dwivedi, K. & Roig, G. *Representation similarity analysis for efficient task taxonomy & transfer learning* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), 12387–12396.

15. Conwell, C., Prince, J. S., Alvarez, G. A. & Konkle, T. *What can 5.17 billion regression fits tell us about artificial models of the human visual system?* in *SVRHM 2021 Workshop@ NeurIPS* (2021).

16. Yamins, D. L. & DiCarlo, J. J. Using goal-driven deep learning models to understand sensory cortex. *Nature neuroscience* **19,** 356–365 (2016).

17. Cadena, S. A. *et al.* Deep convolutional models improve predictions of macaque V1 responses to natural images. *PLoS computational biology* **15,** e1006897 (2019).

18. Denfield, G. H., Ecker, A. S., Shinn, T. J., Bethge, M. & Tolias, A. S. Attentional fluctuations induce shared variability in macaque primary visual cortex. *Nature communications* **9,** 1–14 (2018).

19. Russakovsky, O. *et al.* Imagenet large scale visual recognition challenge. *International journal of computer vision* **115,** 211–252 (2015).

20. Felleman, D. J. & Van Essen, D. C. Distributed hierarchical processing in the primate cerebral cortex. *Cerebral cortex (New York, NY: 1991)* **1,** 1–47 (1991).

21. Gattass, R., Gross, C. & Sandell, J. Visual topography of V2 in the macaque. *Journal of Comparative Neurology* **201,** 519–539 (1981).

22. Gattass, R., Sousa, A. & Gross, C. Visuotopic organization and extent of V3 and V4 of the macaque. *Journal of neuroscience* **8,** 1831–1845 (1988).

23. Dumoulin, S. O. & Wandell, B. A. Population receptive field estimates in human visual cortex. *Neuroimage* **39,** 647–660 (2008).

24. Freeman, J. & Simoncelli, E. P. Metamers of the ventral stream. *Nature neuroscience* **14,** 1195–1201 (2011).

25. El-Shamayleh, Y. & Pasupathy, A. Contour curvature as an invariant code for objects in visual area V4. *Journal of Neuroscience* **36,** 5532–5543 (2016).

26. Cadena, S. A. *et al. How well do deep neural networks trained on object recognition characterize the mouse visual system?* in *Advances in Neural Information Processing (NeurIPS) Neuro-AI Workshop* (2019). https://openreview.net/forum?id=rkxcXmtUUS.

27. Steder, B., Rusu, R. B., Konolige, K. & Burgard, W. *NARF: 3D range image features for object recognition* in *Workshop on Defining and Solving Realistic Perception Problems in Personal Robotics at the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)* **44** (2010).

28. Srinath, R. *et al.* Early emergence of solid shape coding in natural and deep network vision. *Current Biology* **31,** 51–65 (2021).

29. Schrimpf, M. *et al.* Brain-score: Which artificial neural network for object recognition is most brain-like? *BioRxiv,* 407007 (2020).

30. Dwivedi, K., Bonner, M. F., Cichy, R. M. & Roig, G. Unveiling functions of the visual cortex using task-specific deep neural networks. *PLoS computational biology* **17,** e1009267 (2021).

31. Zhuang, C. *et al.* Unsupervised neural network models of the ventral visual stream. *Proceedings of the National Academy of Sciences* **118** (2021).

32. Geirhos, R. *et al.* On the surprising similarities between supervised and self-supervised models. *arXiv preprint arXiv:2010.08377* (2020).

33. Chen, T., Kornblith, S., Norouzi, M. & Hinton, G. *A simple framework for contrastive learning of visual representations* in *International conference on machine learning* (2020), 1597–1607.

34. Calabrese, A. & Paninski, L. Kalman filter mixture model for spike sorting of non-stationary data. *Journal of neuroscience methods* **196,** 159–169 (2011).

35. Shan, K. Q., Lubenov, E. V. & Siapas, A. G. Model-based spike sorting with a mixture of drifting t-distributions. *Journal of neuroscience methods* **288,** 82–98 (2017).

36. Ecker, A. S. *et al.* State dependence of noise correlations in macaque primary visual cortex. *Neuron* **82,** 235–248 (2014).

37. Deng, J. *et al. Imagenet: A large-scale hierarchical image database* in *2009 IEEE conference on computer vision and pattern recognition* (2009), 248–255.

38. Quiroga, R. Q., Reddy, L., Koch, C. & Fried, I. Decoding visual inputs from multiple neurons in the human temporal lobe. *Journal of neurophysiology* **98,** 1997–2007 (2007).

39. Ioffe, S. & Szegedy, C. *Batch normalization: Accelerating deep network training by reducing internal covariate shift* in *International conference on machine learning* (2015), 448–456.

40. Klindt, D. A., Ecker, A. S., Euler, T. & Bethge, M. Neural system identification for large populations separating" what" and" where". *arXiv preprint arXiv:1711.02653* (2017).

41. Sinz, F. H. *et al.* Stimulus domain transfer in recurrent models for large scale cortical population prediction on video. *BioRxiv,* 452672 (2018).

42. Clevert, D.-A., Unterthiner, T. & Hochreiter, S. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289* (2015).

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

44. Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

45. Bay, H., Ess, A., Tuytelaars, T. & Van Gool, L. Speeded-up robust features (SURF). *Computer vision and image understanding* **110,** 346–359 (2008).

46. Shi, J. & Malik, J. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence* **22,** 888–905 (2000).

47. Zhou, B., Lapedriza, A., Xiao, J., Torralba, A. & Oliva, A. Learning deep features for scene recognition using places database (2014).

48. Lin, T.-Y. *et al. Microsoft coco: Common objects in context* in *European conference on computer vision* (2014), 740–755.

49. Paszke, A. *et al.* in *Advances in Neural Information Processing Systems 32* (eds Wallach, H. *et al.*) 8024–8035 (Curran Associates, Inc., 2019). http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

50. Harris, C. R. *et al.* Array programming with NumPy. *Nature* **585,** 357–362. https://doi.org/10.1038/s41586-020-2649-2 (Sept. 2020).

51. Van der Walt, S. *et al.* scikit-image: image processing in Python. *PeerJ* **2,** e453 (2014).

52. Hunter, J. D. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering* **9,** 90–95 (2007).

53. Waskom, M. *et al. mwaskom/seaborn: v0.8.1 (September 2017)* version v0.8.1. Sept. 2017. https://doi.org/10.5281/zenodo.883859.

54. Yatsenko, D. *et al.* DataJoint: managing big scientific data using MATLAB or Python. *BioRxiv,* 031658 (2015).

55. Kluyver, T. *et al. Jupyter Notebooks – a publishing format for reproducible computational workflows* in *Positioning and Power in Academic Publishing: Players, Agents and Agendas* (eds Loizides, F. & Schmidt, B.) (2016), 87–90.

56. Merkel, D. Docker: lightweight linux containers for consistent development and deployment. *Linux journal* **2014,** 2 (2014).

Supplementary Figures

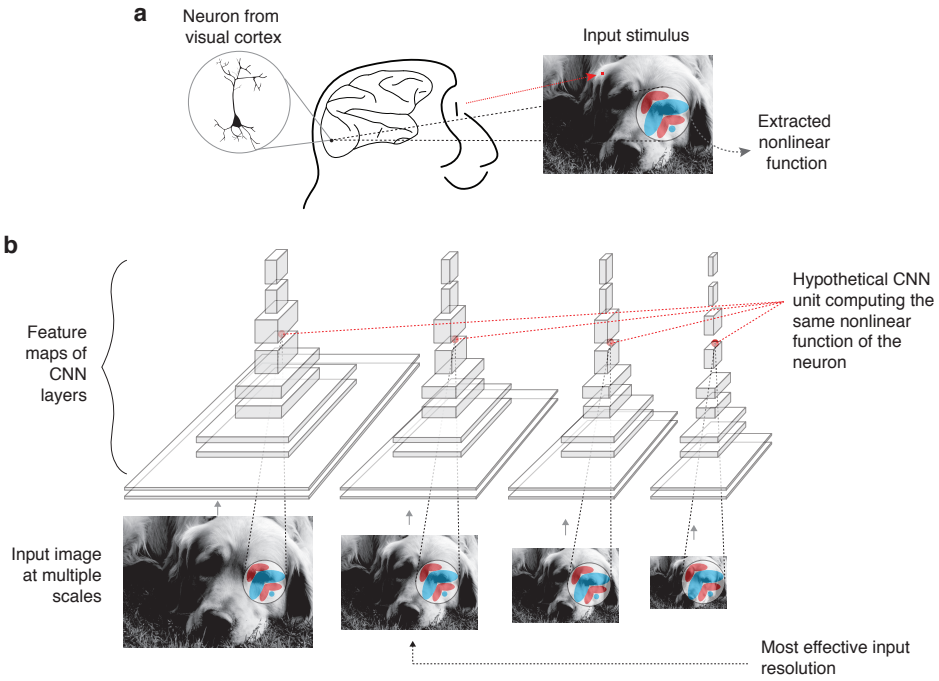

Figure S1. A case for input scale optimization. **a**, A single excitatory neuron from visual cortex, recorded from a head-anchored monkey sitting at a certain distance from a screen and fixating on a spot; extracts a nonlinear function of the input stimulus with a specific receptive field coverage. **b**, A pretrained deep convolutional neural network (CNN) extracts several nonlinear feature maps at each of its intermediate layers. A single output unit of a feature map computes a nonlinear function on its analytical receptive field with a fixed size in pixels. Even if the real neuron's nonlinear function was exactly matched to that of a CNN unit, we would have troubles finding it if we were to forward the input image at the wrong input resolution (in terms of pixels per visual angle). It is oftentimes difficult to predict *a priori* the optimal resolution at which a certain layer extracts the right nonlinearities that best match our responses, especially when the receptive field sizes of neurons are difficult to estimate for higher visual areas, and when recording beyond the foveal region of the visual field. We thus treated the input resolution as a hyperparameter that we cross-validate on the validation set. This facilitates removing the confound between the degree of nonlinearity and receptive field growth when trying to establish hierarchical correspondence between CNN layers and the biological visual system.

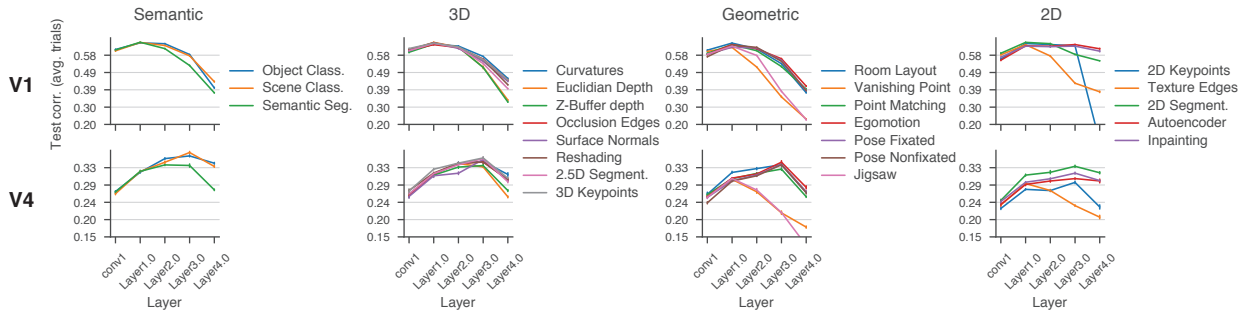

Figure S2. Individual task-model performances on V1 (upper row) and V4 (bottom row) as a function of network layer organized in columns by the task-clusters[2]. The task-model labels are shared between V1 and V4, and placed to the right of each column. Each line represents the average performance over seeds of the mean performance over neurons of the best task-model configuration in the validation set. That means that these lines represent the test set performance after pooling over input scales, and hyper-parameters (i.e. regularization penalty). Bars represent 95% confidence intervals of 1 s.e. of the mean for five seeds. We measured performance as the average test score over single units ($n_{V1} = 458$, $n_{V4} = 255$) calculated as the correlation between model predictions and mean responses over repetitions.

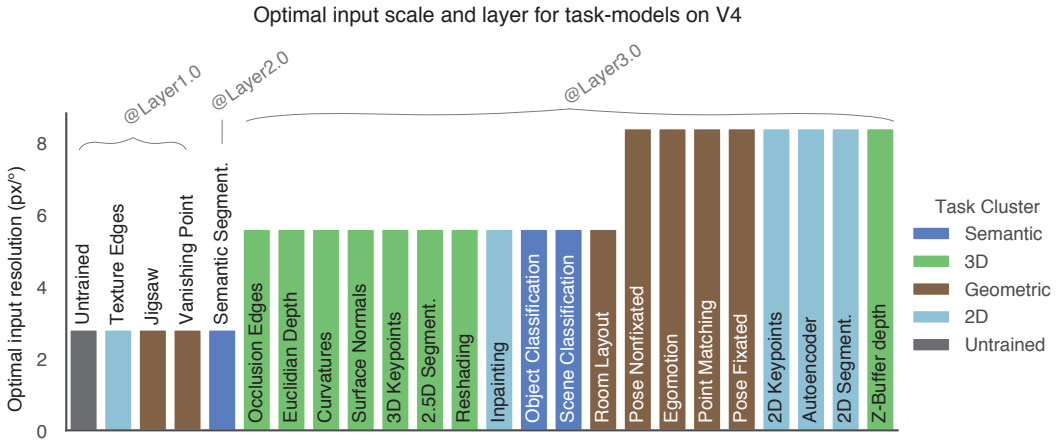

Figure S3. Optimal input scale and layers for task-models on V4. In contrast to area V1 where the optimal layer and scale was shared among all task-models (Layer1.0 and 21px/°), there was variability of the optimal layer in the V4 task-models. In some models, including the untrained network, Layer1.0 with a low input resolution was optimal. The top performing models, including the two semantic classification, and most of 3D tasks (see Fig. 2c) chose an intermediate resolution ($\sim$ 5.6px/°) at Layer3.0. Interestingly, most geometric and 2D tasks yielded optimal performances at the same layer, but at a higher resolution.

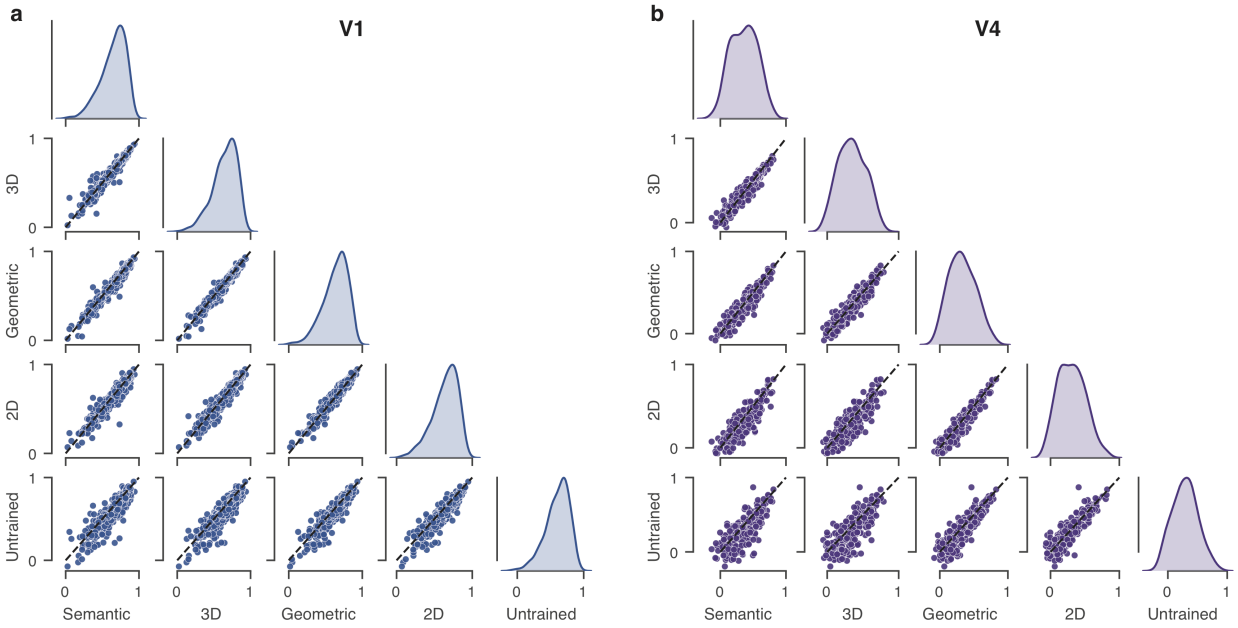

Figure S4. Comparison of the average task-cluster performance on single-neurons in V1 (**a**) and V4 (**b**). The dotted line in each pairwise comparison represents the identity and the panels in the main diagonal shows the performance distribution of each task-cluster. A pairwise Wilcoxon signed rank test reveal that the differences between task-clusters were significant (see Fig. 2d,e)