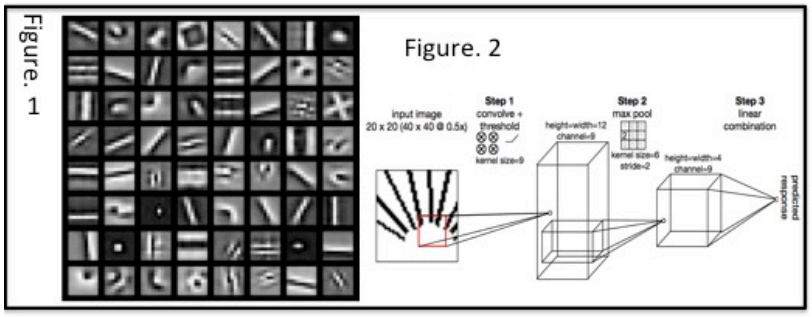
Characterizing V1 Neurons Using Convolutional Neural Networks and Project Pursuit Models with Interpretable Sparse Coding Kernels

Z. Wu, Y. Zhang, S. Tang, T.S. Lee,

Summary: Recent statistical modeling techniques, such as pursuit models and convolutional neural network, intended to recover visual neurons' receptive fields or preferred features by fitting neuron's responses to a large number of images. These models have decent performance in terms of prediction but tend to yield kernels with fairly noisy and uninterpretable features. This might due to the fact that the problem is under-constrained, and there are many possible local minima solutions. We hypothesize that features learned by unsupervised learning based on sparse coding principle (Olshausen and Field, 1996) might provide more interpretable kernels to build projection pursuit or CNN models for predicting neurons' responses. Our experiments show that this approach yielded more interpretable feature kernels and at the same time produced better prediction performance than CNN models (Zhang et al. 2018, J. Computational Neuroscience) with the same number or more parameters. Our experiments also suggest that our model is more robust against noise than the CNN models. Having a set of interpretable feature detectors provide a new approach for us to model and classify V1 neurons with complex tunings (Tang et al. 2017, Current Biology) based on their feature preference.

Motivation: In our earlier paper (Y. Zhang et al. 2018), as well as works of others (M. Bethge, 2017), convolutional neural networks (CNN) were used to fit neuronal responses, with state-of-the-art performance. However, the receptive field features or kernels recovered by CNN are typically noisy and lack interpretability, and not useful for revealing preferred features of the neurons. We conjecture that there exists a set of less noisy and more interpretable basis filters that span the same space as those spanned by the CNN filters, which might better describe the preference of the neurons. We tested this conjecture by fitting projection pursuit or CNN model with the set of feature dictionary derived from sparse coding principle fixed to be the kernels.

Method: We first learnt a set of complete and over-complete basis features using the method of convolutional sparse coding (Y. LeCun et al, 2010) and use them as the set of feature dictionary for CNN and projection pursuit modeling of the neural responses (J. Friedman et al, 1981). ***Convolutional sparse coding*** provides an efficient way to extract features for coding images with low redundancy. One set of learnt sparse coding dictionary is shown as in Fig.1. These features are found to resemble macaque V1 neurons' receptive fields. We thus find it reasonable that assume that these models of V1 receptive fields can be used as the first layer of the CNN so that the parameters can be used in the higher layer to use composition of these filters' responses to construct more complex feature functions to model the complexity of the V1 neurons we found. Subsequently we experiment this idea with two classes of methods: *projection pursuit regression (PPR*) and *convolutional neural networks (CNN).* ***PPR*** models input image as a sum of general smooth functions of linear sum of filters over this image; such filters are trained in an iterative manner. This method has been used as an approach to recover neurons' receptive fields (L. Liu et al, 2016). We proposed a new approach *convolutional matching pursuit regression (CMPR)*, shown as CMPR algorithm below, with two novel ideas, the first is to a set of fixed learned kernels, and the second is to convolve the image with the kernels, instead of taking the dot product as in original PPR. This allows fitting each type of filters to the entire image, solving the translational invariance issue. The second approach we experiment is



the *Fixed Kernel CNN (FKCNN)* using the same CNN model as in our previous paper (shown in Fig. 2). Instead of learning the kernel filter, FKCNN selects and fixes the kernels from the learnt dictionary and only learn the parameters of the fully connected layer.

Experiment and Result: The dataset was the responses of 1142 V1 neurons to 9500 pattern stimuli, including edges, corners, curvatures, crosses, and some other (shown in Fig. 3) for two monkeys and recording their visual neurons' firing rate. Using cross-validation, we fitted our models to the neural responses to a set of input images, and then tested on different sets, using the Pearson correlation of the predicted response and the actual response as an objective metric. We did not include neurons with fewer than 50 stimuli above 0.2 in df/f singal level. Table. 1 shows the average correlation across 781 neurons for each model. CNN 4 is the baseline CNN model with 4 kernel filters this model has the same number of parameter as FKCNN model but performs worse. FKCNN is performing as good as CNN 9, which is CNN with 9 kernel filters, but the number of parameters and training time for FKCNN is significantly less.

Algorithm 2 CMPR Algorithm

procedure CMPR(X, y, D)
▷ input set of images X, corresponding neural response y,
and feature dictionary D learned from sparse coding.
$\psi(X) = \sum_{m=1}^{M} ploy(F_m * X, 2)$
$r_i \leftarrow y_i$
$m \leftarrow 0$
while $m \leq M$ **do**
▷ M: Number of layers (filters) expected
for $F \in D$ **do**
$I(F) = 1 - \sum_{i=1}^{n}(r_i - ploy(F * X, 2))^2 / \sum_{i=1}^{n} r_i^2$
optimize the parameters of quadratic function.
calculate the loss $L(F)$ (i.e. training performance).
select $F_m \leftarrow argmin L(F)$
$r_i \leftarrow r_i - ploy(F_m * X, 2)$
$m \leftarrow m + 1$

Figure.3

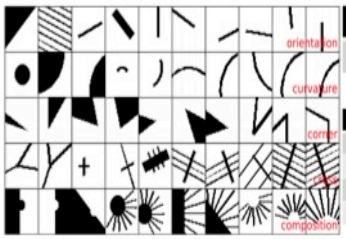


Table. 1

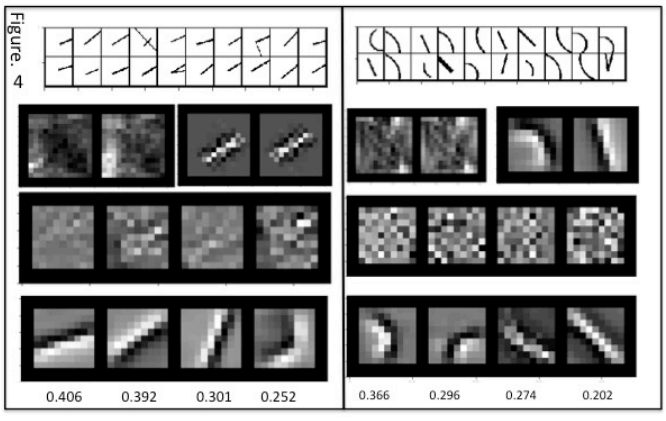
Model:	CPPR	CMPR	CNN 4	CNN 9	FKCNN
Performance:	0.203	0.242	0.374	0.418	0.423

Table. 2

Model:	FKCNN:	CNN 4:	CNN 9
Original	0.423	0.374	0.418
10% noise	0.258	0.173	0.205
20% noise	0.179	0.125	0.143
30% noise	0.121	0.100	0.110

Interpretability: We found that CNN built on transfer learning using interpretable filters outperformed the state-of-art models. Two neurons in Fig. 4 shows the top 20 preferred stimuli on the top row; on the second row, two kernels on the left are from CPPR and two kernels on the right are from CMPR; the third row are the four kernels in CNN model; fourth row are the top 4 most contributed kernels in FKCNN model, the importance is ranked from left to right. The filters are selected based on how much they contribute to the performance, i.e. the first one is the one, without which the model's performance suffers the most, and so on. Apart from the fact that such filters are visually more interpretable and significantly closer to the neuron's top responding stimuli, our model FKCNN is shown to be more robust against noise as in Table. 2 where 10% noise means we add 10% of "salt and pepper" noise pixels into the test images.

Figure. 4



Neuron Classification: We have used a way to classify neurons with fixed tunings using a very stringent criterion that is biased against higher order classification. All the stimuli above 50% of the maximum response of a neuron has to be long to one of the higher order class (corner, cross, curvature) in order for that neuron to be classified into one of those HO class. If the neuron responds to only one bar stimulus above its 0.5 maximum, it will automatically disqualified as a higher order (HO) neuron, and classified as an OT (orientation tuned) neuron. By discovering the most important features that a neuron likes, we can classify the neurons based on their most preferred fitted feature, removing the bias. We found classification based on this simple and intuitive measure is roughly 75% consistent with our earlier more stringent method, and discrepancy mostly lies in those neurons prefer mostly HO features, but fall under OT class due previous classification method as shown in Fig. 4's right neuron.