Assignment 2

CS 375/Psych 249 (Stanford University, Autumn 2018)

1 Assignment Questions

The purpose of this assignment is to explore multiple methods for comparing models and brain responses. These methods include linear regressions from model features to neural responses and Representational Similarity Analysis Kriegeskorte et al. [2008a] between models and brain responses.

1.1 Representational Similarity Analysis

Background: The Representational Dissimilarity Matrix (RDM) of a feature representation F with n images and m feature dimensions is the $n \times n$ matrix M defined by

$$M(i,j) = 1 - corr(F[i], F[j]),$$

where $F[i]$ is the length-m feature vector response to image i, and $corr$ is the pearson product-moment correlation[1]. RDMs describe how the m-dimensional feature representation lays out images relative to each other. To compare RDMs between two different representations F_1 and F_2, each $n \times n$ symmetric matrix M_1, M_2 is flattened into a length $n \cdot (n-1)/2$-dimensional vector, and then the Spearman rank-order correlation[2] between the two flattened vectors is computed.[3] This type of analysis comparing two RDMs is sometimes called "Representational Similarity Analysis", and it measures how similar the two representations treat the images. For more about RSA, see Kriegeskorte et al. [2008b], which introduced its use in neuroscience.

As for this assignment, RDMs computed from image-level responses are a bit hard to work with, as there are around 6,000 images. For this reason, you can reduce the RDM on a per-object basis by taking the mean of the features for all images of a given object. Since there are 64 objects in the dataset, this leads to a 64 x 64 RDM, which is much easier to work with. In this case, the 64x64 is also nicely visualizable, if the objects are ordered by category, with category block-diagonals emerging.

To be more specific, we expect you to do the following:

1. Use images in variation level V3 and V6, compute object-level RDMs for V4 and IT neuronal features, order the objects by category and visualize the RDMs.
2. Equally choose at least 6 layers from final ResNet-18 checkpoint, then similarly compute and visualize RDMs for these layers. You can use "test_from_params" in tfutils to do this.
3. For each RDM computed for network features, compute its correlation with V4 and IT RDMs. Plot two plots individually for V4 and IT for the correlations v.s. layers. What is the most similar layer to V4 and what is the most similar layer to IT in ResNet-18?

1.2 Linear regressions

For each neuron n_j, use linear regression to identify weights w_{ij} and bias b_j on model features f_i from various network layers, such that

$$n_j \sim b_j + \sum_{i=0}^{n} w_i f_i.$$

However, directly fitting the weights needed here for all model features can lead to serious overfitting. Taking ResNet-18 as an example, the highest layer of this model before pooling and fully connected layer is in the shape of $[7, 7, 512]$, which means there will be $7 \times 7 \times 512 + 1 = 25089$ parameters for one neuron. As there are only 5760 images in total, this

[1] `https://en.wikipedia.org/wiki/Pearson_correlation_coefficient`

[2] `https://en.wikipedia.org/wiki/Spearman's_rank_correlation_coefficient`

[3] If starting with a square-form version of the correlation matrix, be careful to only use the upper or lower triangular portion of the flattened matrix when computing the spearman. Otherwise you'll be double-counting all but the diagonal, which is trivially 0 anyhow. Have a look at `https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.pdist.html`, `https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.squareform.html`, and `https://docs.scipy.org/doc/numpy-1.13.0/reference/generated/numpy.triu_indices.html`.

number of parameters for each neuron is apparently too many. In this assignment, we will explore two different ways to solve this problem.

Features subsampling One easy way is to subsample units from model features. One implementation of this method is already shown in this tutorial[4]. As also shown in that tutorial, PLS regression can then be applied to solve the regression problem from subsampled model features to neuronal features.

Disentangle spatial and channel dimension Another method is introduced Klindt et al. [2017], which smartly uses the property of convolutional neural networks. Specifically, this method decomposes the fitting parameters into a spatial mask and a channel mask. For example, assume the model features f are of shape $[X, Y, C]$. Then for each neuron n_j, this method defines a spatial parameter mask $w_{s,j}$ of shape $[X, Y]$ and a channel mask $w_{c,j}$ of shape $[C]$. The final prediction for this neuron computed from model features is:

$$\hat{n}_j = b_j + \sum_{x=0}^{X} \sum_{y=0}^{Y} \sum_{c=0}^{C} w_{s,j}[x, y] * w_{c,j}[c] * f[x, y, c].$$

In other words, the linear weights have been factored into a separable product of the spatial receptive field and the channel weights.

To optimize for these parameters, the loss function should be of the form:

$$L_j = ||n_j - \hat{n}_j||^2 + \lambda_s ||w_{s,j}||^2 + \lambda_c ||w_{c,j}||^2$$

where λ_c is L2 regularization parameter for channel mask and λ_s is for spatial mask.

This method greatly reduces the number of parameters for one neuron from $XYC + 1$ to $XY + C + 1$.

In this assignment, we expect you to do the following:

1. For the random feature subsampling method, use images in variation level V3 and V6, fit to both V4 and IT neurons from the same network features used in section 1.1 through subsampling 1250 units if needed and using parameters for PLS regression similar to those in the tutorial. Just as in the tutorial, you should use "test_from_params" in tfutils for this problem. For each pair of model layer and neuronal area (V4 or IT), compute the median of Pearson correlations between model predictions and neural ground truth on test images across all neurons within that area. Then, for both V4 and IT, plot the medians of correlations v.s. layers. Please use multiple train-test data splits to get multiple values so that you can have an error-bar plot. What is the best layer for V4 and IT?

2. Now do the same again, but this time using the Klindt method. With images in variation level V3 and V6, fit to both V4 and IT neurons from the same network features used in section 1.1. You can set both parameters λ_s and λ_c to be $1e - 3$. For implementation, you need to use "train_from_params" in tfutils, while keeping the ResNet-18 network weights fixed and only training the spatial and channel masks for neurons. Plot the same error-bar plot for results from this method. What is the best layer for V4 and IT?

3. As mentioned in the class, the spatial mask from the Klindt method has a biological meaning, which is the receptive field for one neuron. Looking at trained spatial masks and comparing them between V4 and IT neurons can tell us the difference between receptive fields of V4 and IT neurons. To do this, please first choose the network layer that best explains V4, then randomly choose 25 neurons in V4 area and visualize the trained spatial masks for these V4 neurons. For the same network layer, randomly choose 25 neurons in IT area and visualize the trained spatial masks for these IT neurons. Compare these two groups of spatial masks, what do you find? Do the same thing for the network layer that best explains IT, what do you find?

4. (Optional and bonus) We have suggested the values for several parameters including number of units to subsample, including `n_components`$= 25$, and $\lambda_s = \lambda_c = 1e - 3$. However, these regularization parameter suggestions may not be optimal. You can vary these parameters to explore the optimal values, trying different values and then use the median of correlations across neurons to decide which regularization values are best.

References

D. Klindt, A. S. Ecker, T. Euler, and M. Bethge. Neural system identification for large populations separating "what" and "where". In *Advances in Neural Information Processing Systems*, pages 3506–3516, 2017.

[4]`https://github.com/neuroailab/cs375/blob/master/2018/tutorials/Tutorial%20--%20Mapping%20Models%20to%20Neural%20Data%20(Lecture%205).ipynb`

N. Kriegeskorte, M. Mur, and P. Bandettini. Representational similarity analysis - connecting the branches of systems neuroscience. *Front Syst Neurosci*, 2:4, 2008a.

N. Kriegeskorte, M. Mur, D. A. Ruff, R. Kiani, J. Bodurka, H. Esteky, K. Tanaka, and P. A. Bandettini. Matching categorical object representations in inferior temporal cortex of man and monkey. *Neuron*, 60(6):1126–41, 2008b.