Data Science Tools for Neural Processing

Tools and tips for analyzing neural data

1. *Data compression*
    a. Let's say you have 1000 fMRI voxels. Brain state can be mapped to a 1000-dimension feature space at each time point. Data compression can map this data to a much smaller feature space such as 25 voxels and perserve most of the information. Lower dimensional feature spaces are easier to manage.
    b. Let's say you do connectivity analysis and have 1000 graph edge values. Compress this to only 25 values
    c. Principal Component Analysis (PCA)
        i. Map fMRI data to principal components and select components with most explained variance
        ii. https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html
        iii. https://en.wikipedia.org/wiki/Principal_component_analysis
    d. Extract outputs from bottleneck layer from feedforward autoencoder neural network
        i. Autoencoders are artificial neural networks that try to learn their own input. The input and output later may be 1000 nodes each corresponding to the 1000 voxel input data. However, the hidden "bottleneck" layer may only be 25 nodes
        ii. After training autoencoder, get output from bottleneck layer
        iii. https://blog.keras.io/building-autoencoders-in-keras.html
        iv. https://en.wikipedia.org/wiki/Autoencoder
    e. Manual feature selection based on knowledge of neural mechanism
        i. Pick 25 voxels associated with ROI you want to analyze
    f. Feature selection based on genetic algorithm
        i. Uses a supervised approach to search for more important set of features that predict a tag
        ii. https://medium.com/analytics-vidhya/feature-selection-using-genetic-algorithm-20078be41d16

2. *Clustering*
    a. Brain states can be clustered in an unsupervised manner
    b. K-means
        i. Pick how many clusters you want and cluster your brain state
        ii. Compress data before clustering
        iii. https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

3. *Decoding*
    a. Let’s say you want to understand which brain area or connection corresponds to color and your experiment has a red and green condition. Train a decoder to predict color representation.
    b. Experiment with inputting different features into to the predictor. Features may correspond to voxels, brain connectivity measures, or even clusters. Sets of features that yield highest decoding accuracy represent color information the most
    c. Support Vector machine
        i. https://towardsdatascience.com/support-vector-machines-explained-with-python-examples-cb65e8172c85
        ii. https://scikit-learn.org/stable/modules/svm.html
    d. Random Forest
        i. https://towardsdatascience.com/understanding-random-forest-58381e0602d2
        ii. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
    e. Linear Regression
        i. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html
    f. Artificial Neural Networks
        i. Only use if you have a lot of robust data.
        ii. https://towardsdatascience.com/a-beginners-guide-to-artificial-neural-network-using-tensor-flow-keras-41ccd575a876
    g. Avoid overfitting!
        i. Make sure your model does not have too many parameters compared to the amount of input data
        ii. Split training and testing with n fold cross validation
            1. https://scikit-learn.org/stable/modules/cross_validation.html