A Neurocognitive Graph Theoretical Approach to Understanding the Relationship Between Minds and Brains

Joshua T. Vogelstein, R. Jacob Vogelstein, Carey Priebe
Johns Hopkins University

January 9, 2010

Abstract

1 Paradigms of research

1.1 Current Paradigm

The dominant paradigm of quantitative neuroscience in the 20^{th} century has been the "signal processing" framework []. Essentially, the brain is a box, that filters some stimulus, to produce some response (see Figure 1). This framework leads to the following goal:

Goal 1. *Learn the filter that the brain performs on stimuli to result in the actualized responses.*

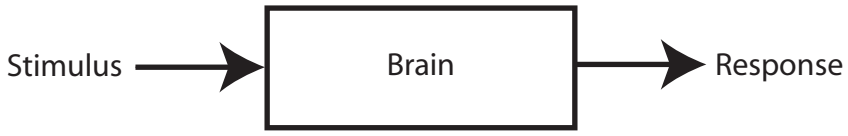

Figure 1: The signal-processing paradigm of quantitative neuroscience. The brain is a box that essentially *filters* the stimulus, outputting some response, which is often take to be multivariate time series (such as populations of spike trains or fMRI activity).

This paradigm has been appropriate, given the kind of data available to people investigating the brain. More specifically, the kind of data that has been most available has been time-series of signals related to neural activity []. Given that electrical engineers were largely the individuals obtaining and analyzing the data, it was natural to take a signal-processing approach. Often, the dynamic time-series data were used to estimate static parameters. In the previous decades, these communities have been more and more sophisticated models and algorithms to estimate these parameters []. Recently, issues such as parameter identifiability, consistency, bias, and model selection have been gaining traction as important desiderata for our models [].

1.2 Alternate Paradigm

Here, we define a different goal, which suggests a complementary research paradigm to the current dogma:

Goal 2. *Construct a family of brain-models, $\mathcal{B}$, that is sufficient to provide* causal *explanations relating properties of minds with properties of brains.*

Note how the above goal is distinct in certain respects from the "filtering" goal. First, neither stimulus nor response is explicitly incorporated into this goal. While stimuli and response may be used as tools to obtain the parameters of $\mathcal{B}$, that is their only merit in this paradigm. Second, minds are explicitly incorporated into this goal. While spike trains

or fMRI signal may indicate mental processes, they are likely not what is meant by "mind"; rather, they may be used as tools to infer mental states. Third, the inclusion of a class of models $\mathcal{B}$ suggests a statistical *static* framework, rather than a dynamics perspective. In addition to the above stated goal, we would like the family of models $\mathcal{B}$ to satisfy a number of desiderata:

1. $\mathcal{B}$ should be sufficiently general to account for whatever properties of the brain are casually related to the properties of cognition under investigation.
2. The properties of any particular brain, $b \in \mathcal{B}$, should be either measurable or estimatable, such that experimental observation may be used to obtain them.
3. $\mathcal{B}$ should admit algorithms that (are guaranteed to be able to) capture the relationship of interest.
4. $\mathcal{B}$ should also admit *causal* studies, which entail modifying a particular $b \in \mathcal{B}$, to modify the corresponding mental property, $m \in \mathcal{M}$.

2 NeuroCognitive Graph Theory

2.1 Brain-graphs

Here, we propose the notion of a *brain-graph*. Specifically, we say that the brain may be well characterized as a labeled, attributed multigraph (which is a generalized notion of a or network). Formally, we define a brain-graph, $b \in \mathcal{B}$ as a 4-tuple, $\mathcal{B} = (\mathcal{V}, \mathcal{E}, \mathcal{X}_V, \mathcal{X}_E)$, defined by the following:

- The set of vertices (nodes), $V = \{V_i\}_{i \in [n_v]}$, where $V_i \in \mathcal{V} \subseteq \mathbb{Z}$ for $i \in [n_v] = \{1, 2, \ldots, n_v\}$.
- The set of edges, $E = \{E_{ij}\}_{i,j \in [n_v]}$, where $E_{ij} \in \mathcal{E} \subseteq \mathbb{Z}$ for $(i, j) \in [n_v] \times [n_v]$. If edges are undirected, then $E_{ij} = E_{ji}$. If edges are binary, then $\mathcal{E} = \{0, 1\}$. If we have multi-edges corresponding to categorically different edges (such as electrical and chemical synapses), each edge may also be indexed by $k \in \mathbb{Z}$. If multi-edges correspond to integer weighted edges, then $\mathcal{E}$ is the set of allowable integers (which may be countably infinite).
- If vertices have features (or labels/attributes), then let X_i correspond to the feature vector for vertex i, where $X_i \in \mathcal{X}_V \subseteq \mathbb{R}^{d_V}$ for $i \in [n_v]$, and d_V is the dimensionality of the feature vectors. It may be the case that certain features are measurable/observable, and others are hidden. If so, let $X_i = X_i^o \cup X_o^h$, and $X_i^o \cap X_i^h = \emptyset$.
- If edges also have features, then let X_{ij} correspond to the feature vector for edge (i, j), where $X_{ij} \in \mathcal{X}_E \subseteq \mathbb{R}^{d_E}$ for $(i, j) \in [n_v] \times [n_v]$, and d_E is the dimensionality of the edge features. In the scenario where categorically different edges exist, let an additional index k indicates edge features for each category k.

Assume, for the moment, that we take the fundamental computational unit of the brain to be a point neuron. Then, each vertex is a neuron, and each edge is a synapse. Categorically different edges may correspond to chemical and electrical synapses. Vertex attributes could include neurotransmitter released, proteins expressed, morphological properties, receptive fields, etc. Edge attributes could include probability of release, post-synaptic potential shape, etc. This level of description, however, is not necessary. For instance, $\mathcal{V}$ might instead correspond to neuroanatomical regions, which admit very different notions for edges and attributes. This multi-scale aspect of $\mathcal{B}$ is an important advantage over other frameworks.

2.2 What to do with these brain-graphs

Given $\mathcal{B}$, what can we do with it? As stated above, our goal is to relate these models to properties of cognition. More specifically, let $\mathcal{M}$ characterize the space of the mental (cognitive) property, and $g \in \mathcal{G}$ be some function to learn.

- If $\mathcal{M} = \{0, 1\}$, then g is a two-way classifier: $g : \mathcal{B} \to \{0, 1\}$.
- If $\mathcal{M} = \{0, 1, \ldots, n_m\}$, then g is an n_m-way classifier: $g : \mathcal{B} \to \{0, 1, \ldots, n_m\}$.
- If $\mathcal{M} = \mathbb{R}^a$, then g is a (multivariate-) regressor: $g : \mathcal{B} \to \mathbb{R}^a$.

In general, solving the above problems—which means finding g—will depend on the joint distribution of brains and minds, $F = F_{BM}$. In practice, however, F is typically unknown, and therefore g must be estimated from the data. Assume we have a corpus of training data, $\mathcal{D}_n = \{(B_1, M_1), \ldots, (B_n, M_n)\}$, where n is the number of training samples. Our goal then is to compute $g_n : \mathcal{B} \times (\mathcal{B}, \mathcal{M})^n \to \mathcal{M}$, which takes as input an observed brain-graph b and n training paris $\{(b_1, m_1), \ldots, (b_n, m_n)\}$ and produces a prediction $\widehat{m} = g_n(b; \mathcal{D}_n)$. The particular g_n should be the one that minimizes some loss function, $L_F(g_n)$, over the space of all possible g_n's, $\mathcal{G}$. For instance, when $|\mathcal{M}| = 2$, $\mathcal{G}$ is all possible two-way classifiers, and a potentially reasonable loss function is $L_F(g_n) = \mathbb{E}[P_F[g_n(B; \mathcal{D}_n) \neq M | \mathcal{D}_n]]$.

Importantly, in addition to finding a g_n that minimizes some loss-function, g_n should admit a way to *morph* any brain's mental property m, by modifying b. This will be discussed at greater length in the sequel.

2.3 Finding a good g_n

Thus, given a mental property, a decision about how to represent it, $\mathcal{M}$, and a loss function, L, our task is to find a good algorithm, g_n. Two complementary strategies are possible: model-free and model-based. Model-free algorithms have the advantage that no model need be specified. Thus, in theory, model-free algorithms have the advantage of having little or no bias. Unfortunately, this freedom comes with the cost of relatively high variance. On the other hand, model-based algorithms can significantly reduce variance, but (almost) necessarily increase bias. Importantly, many standard algorithms, including linear, quadratic, and support vector based classifiers/regressors *implicitly* define a model, and are therefore not strictly "model-free".

2.3.1 Model-free algorithms

Model-free algorithms often operate on interpoint distance space, as opposed to the explicit data space [**?**]. More formally, given any two brain-graphs, b_1 and b_2, first define an interpoint (pseudo-) distance metric: $\rho : \mathcal{B} \times \mathcal{B} \mapsto [0, \infty)$. This reduces the problem from operating in $\mathcal{B}$ to operating in $\mathbb{R}$. Because the data collected is often corrupted by noise, it is typical to also introduce a smoothing function: $s : \mathcal{B} \mapsto \mathcal{B}$. For the brain-graph scenario, this may correspond to inferring unobserved edges. Thus, the smoothing-derived (pseudo-) distance metric, ρ' is defined as: $\rho' = \rho(s(b_1), s(b_2))$.

Perhaps the prototypical model-free algorithm is the k_n nearest neighbor (kNN) algorithm. Vogelstein et al. (2010) showed that a kNN classifier is a universally consistent classifier (meaning, achieves the Bayes optimal performance), for any F_{BM}, under a Frobenius norm distance metric. In other words, they let defined $\rho(\cdot) = \|\cdot\|_F$, and s was simply the identity (that is, no smoothing). Simulations showed that for only a few hundred simulated sample data points, this kNN achieved misclassification error rate below 10%. In practice, it is often the case that other model-free algorithms outperform (in accuracy) any particular kNN, including class cover catch digraphs, decision trees, and various ensemble approaches such as random forests []. Unfortunately, scant theoretical works is available to provide proofs of universal consistency for these alternate model-free algorithms.

2.3.2 Model-based algorithm

The other possible strategy is to propose a class of models, $\mathcal{P} = \{P_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \boldsymbol{\Theta} \subseteq \mathbb{R}^d\}$, that describe the data (note that the choice of how to determine the dimensionality d of the model specifies whether the model is parametric, semiparametric, or nonparametric). Ideally, the class of models is sufficiently large to include models very close to the "truth", and be able to find a Minimally Sufficient Model (MSM; by analogy with minimally sufficient statistics) within the class, that sufficiently explains the data, with the "smallest" $\boldsymbol{\Theta}$ (a potentially useful measure of size is the set cardinality). Classification/regression with g_n is then performed on the estimate of $\boldsymbol{\theta}$, which lives in some $\boldsymbol{\Theta}$ space smaller than $\mathcal{B}$, thereby reducing the variance, without increasing bias too much (hopefully). The model-based approach also (potentially) offers the advantage of *interpretability*, if the parameters, $\boldsymbol{\theta}$, correspond to interpretable features of the brain.

Let b_i correspond to brain-graph i. The model-based approach then typically assumes that each b_i is sampled identically and independently from some parametric distribution, $b_i \overset{iid}{\sim} P(b_i|\boldsymbol{\theta}) \, \forall i \in [n]$. Because $\boldsymbol{\theta}$ is typically unknown, an estimate, $\widehat{\boldsymbol{\theta}}$, must be found. Given such an estimate, g_n operates directly on the estimated parameters, as opposed to the brain-graphs [1]

The maximum likelihood approach then specifies to find The task is then to find an estimate $\widehat{\boldsymbol{\theta}}$, such that

[1] Is it interesting to note that neither paradigm actually operates directly on the data? The model-free approach operates

$$\widehat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \prod_{i \in [n]} P(b_i|\boldsymbol{\theta}) \tag{1}$$

If a prior over the parameters is specified

Generative Model Consider the following generative model for attributed multigraphs:

- Let c be a *class identity*, where $c \in \mathcal{C} = \{0, 1, \ldots, C\}$. The distribution of class identity, which is a random variable, is given by $f_c = MN(\boldsymbol{\pi})$, where MN indicates a multinomial, and $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_)$
- Let $\boldsymbol{\theta}_c$ be the *parameters* for class c, where $\boldsymbol{\theta}_c \in \boldsymbol{\Theta} \subseteq \mathbb{R}^d$, for some $d = d_V + d_E \in \mathbb{Z}$, where the dimensionality of the parameters is implicitly a function of the data, $\mathcal{D}_n$.
- Let b be a *brain-graph*, where $b(\boldsymbol{\theta}_c) \in \mathcal{B} = (\mathcal{V}, \mathcal{E}, \mathcal{X}_V, \mathcal{X}_E)$, where for clarity, we restrict edges to be integer weights (i.e., only include a single category of edge attributes, but this may straightforwardly generalized)

To sample graphs from this generative model, assuming that C and V are given (the number of classes and vertices per graph, respectively), one can use the following procedure (generalizing to the unknown V case is straightforward and therefore omitted):

- sample $c \sim f_c$
- sample $\boldsymbol{\theta}_c \sim f_{\boldsymbol{\theta}}(\cdot|c)$
- for $i \in [n_v]$, sample $X_i \sim f_{X_V}(\cdot|\boldsymbol{\theta}_c^V)$
- for $(i, j) \in [n_v] \times [n_v]$
  - sample $X_{ij} \sim f_{X_E}(\cdot|\boldsymbol{\theta}_c^E)$
  - sample $E_{ij} \sim f_E(\cdot|X_i, X_j, X_{ij})$

Note that we have partitioned $\boldsymbol{\theta}_c$ into $\boldsymbol{\theta}_c^V$ and $\boldsymbol{\theta}_c^E$. The probability of obtaining any graph, when using this procedure, is therefore given by:

$$P(G|C, V) = \left(\prod_{i,j \in [n_v]} f_E(E_{ij}|X_i, X_j, X_{ij}) f_{X_E}(X_{ij}|\boldsymbol{\theta}_c^E) \right) \left(\prod_{i \in [n_v]} f_{X_V}(X_i|\boldsymbol{\theta}_c^V) \right) f_{\boldsymbol{\theta}}(\boldsymbol{\theta}_c|c) f_c(c) \tag{2}$$

Evaluating Eq. 2 requires defining $f = \{f_c, f_{\boldsymbol{\theta}}, f_{X_v}, f_{X_E}, f_E\}$ (note that $f_{\boldsymbol{\theta}}$ implicitly depends on defining $\boldsymbol{\Theta} \subseteq \mathbb{R}^d$, which, in turn, requires a rule for deciding d based on the data). The most natural choice for f_c is a multinomial, that is, $f_c = MN(\vec{\pi})$, where $\boldsymbol{\pi} = \{\pi_1, \ldots, \pi_C\}$, $\pi_c > 0$, and $\sum_c \pi_c = 1$. Our task is then to choose the rest of f that yields: (i) models that display the properties of the data, and (ii) are statistically tractable, meaning that $\boldsymbol{\theta}$ may be estimated consistently from the data [2]. To proceed, we first write the posterior of interest. Importantly, it is often the case that some attributes and edges are hidden. Define $\boldsymbol{E} = \{E_{ij}\}_{i,j \in [n_v]}$, and let $\boldsymbol{E} = \boldsymbol{E}^h \cup \boldsymbol{E}^o$, where $\boldsymbol{E}^h$ corresponds to hidden edges, and $\boldsymbol{E}^o$ corresponds to observed edges. Similarly, let $X = X_V \cup X_E$, and then $X = X^h \cup X^o$. Finally, define $\boldsymbol{\theta} = \{\boldsymbol{\theta}_c\}_{c \in [C]}$. Thus, we are interested in estimating/maximizing:

$$P(\boldsymbol{\theta}, X^h, \boldsymbol{E}^h|C, V, X^o, \boldsymbol{E}^o) \propto ... \tag{3}$$

Several strategies for maximizing Eq. 3 are possible. First, one could use a Gibbs sampling strategy, iteratively sampling $\boldsymbol{\theta}$, X^h, and $\boldsymbol{E}^h$. Second, one could use an expectation maximization algorithm, recursively finding the expected values for X^h and $\boldsymbol{E}^h$, and then use them to estimate $\boldsymbol{\theta}$. The precise definition of f might necessitate approximating both these strategies. Alternately, greedy or variational approaches might be more efficient.

[2] The above generative framework generalizes and unifies previously proposed stochastic graph models, including Random Dot Product Graphs, stochastic block models, etc.

Classification problem Imagine we are in the classification setting. What does it mean that $P(B|M = 0) \neq P(B|M = 1)$? It must mean that, for some subset of edges, the distribution is different for brain-graphs in the two different classes. More formally, let $S = \{(i, j)|(i, j) \in S\}$. If class conditional posteriors are different, then it must be the case that $P(\{E_{ij}\}_{(i,j)\in S}|C = 0) \neq P(\{E_{ij}\}_{(i,j)\in S}|C = 1)$. Given the above class of models, we may write:

$$\begin{aligned} P(\{E_{ij}\}_{(i,j)\in S}|c) &= f(\cdot|X, \boldsymbol{\theta}_c) \\ &= f_E(\cdot|X) f_X(\cdot|\boldsymbol{\theta}_c) f_{\boldsymbol{\theta}}(\cdot|c) \\ &= f_E(\cdot|X) f_{X^h}(\cdot|X^o, \boldsymbol{\theta}_c) f_{\boldsymbol{\theta}}(\cdot|c) \end{aligned} \tag{4}$$

Thus, given a specification for f, the task is to find S, estimate the parameters $\boldsymbol{\theta}$, compute the likelihood under the two models, and compare. Finding S, however, is, in general, non-trivial.

Limitations/extensions attributed vertices, hyperedges

3 Simulated applications

4 Concluding thoughts