Introduction 2:

Computational ideas relevant to the study of how the brain (possibly) works

Emanuel Todorov
Cognitive Science Department
University of California San Diego

The need for top-down modeling

data ≠ information ≠ knowledge ≠ understanding

what you collect in the lab why we do science

• The need to interpret/model the details of experimental data is obvious.
• The need to make an educated guess about the high-level explanation and then see if it can be made consistent with data may be less obvious.
• But when studying something as complex as the brain, we have to pursue both bottom-up and top-down approaches simultaneously, and use one to inform the other.
Marr’s levels of modeling

<table>
<thead>
<tr>
<th>Computational theory</th>
<th>Representation and algorithm</th>
<th>Hardware implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>What is the goal of the computation, why is it appropriate, and what is the logic of the strategy by which it can be carried out?</td>
<td>How can this computational theory be implemented? In particular, what is the representation for the input and output, and what is the algorithm for the transformation?</td>
<td>How can the representation and algorithm be realized physically?</td>
</tr>
</tbody>
</table>

“Trying to understand perception [neural information processing] by studying only neurons is like trying to understand bird flight by studying only feathers: it just cannot be done. In order to understand bird flight, we have to understand aerodynamics; only then do the structure of the feathers and the different shapes of birds’ wings make sense.”

Tools of the trade 1: Information theory

**Entropy:** generalization of variance; measures how spread the probability distribution is

\[
H(x) = -\int p(x) \ln p(x) dx = -E_p[\ln p(x)]
\]

**Relative entropy (Kullback-Leibler divergence):** measures “distance” between two probability distributions

\[
KL(p \| q) = \int p(x) \log \frac{p(x)}{q(x)} dx = E_p[\log \frac{p(x)}{q(x)}]
\]

\[
KL(p \| q) \geq 0; \quad KL(p \| q) = 0 \iff p(x) = q(x)
\]

**Mutual information:** generalization of R²; measures how much one variable tells us about the other

\[
I(x,y) = \iint p(x,y) \log \frac{p(x,y)}{p(x)p(y)} dxdy = KL(p(x,y) \| p(x)p(y))
\]

\[
I(x,y) = H(x) - H(x \mid y) = H(y) - H(y \mid x) = I(y,x)
\]

\[
x, y \text{ independent} \iff p(x,y) = p(x)p(y) \iff I(x,y) = 0
\]

Other related concepts: channel capacity; data compression; minimal description length.

*Note:* information and computation are very different things!

If you take a set of images and scramble the pixels, the images still contain exactly the same information, but it is now much more difficult to compute anything from them.
Tools of the trade 2: Probabilistic inference

Given a set of data, and a parameterized model which we believe generated the data, find the parameters of the model that are most consistent with the observed data.

This is what statisticians, and scientists in general, do for a living.

But in some sense it is also what the brain does (so statistics/science should be easy to learn!)

Generative models: parameterized probabilistic descriptions of how we believe data is being generated

Graphical models: a way to represent conditional dependencies in complex probabilistic models

Bayes rule:

\[
p(x, y) = p(x)p(y|x) = p(y)p(x|y) \iff p(x|y) = \frac{p(y|x)p(x)}{p(y)}
\]

Why is Bayes rule so important? Normally we can form a generative model \( p(x|w) \), which tells us the probability of data \( x \) given some parameters \( w \). But in order to infer the most likely parameters \( w \) given the data \( x \), we actually need the opposite: \( p(w|x) \). Bayes rule allows us to do the inversion.

Example of probabilistic inference: computing the mean

Generative model: \( x_i = w + \varepsilon_i \); \( \varepsilon_i \sim N(0,1) \)

Observed data: \( x_1 \cdots x_N \)

Probability of data given model:

\[
p(x_1, \cdots, x_N | w) = \prod_n p(\varepsilon_n = x_n - w) = \frac{1}{\sqrt{2\pi}} \prod_n \exp\left(-\frac{1}{2}(x_n - w)^2\right)
\]

\[
= \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2} \sum_n (x_n - w)^2\right)
\]

Non-informative prior: \( p(w) \) constant \( \Rightarrow p(x | w) - p(w | x) \)

Maximize log-probability: \( -\frac{1}{2} \sum_n (x_n - w)^2 \) Optimal \( w \): \( w^* = \frac{1}{N} \sum_n x_n \)
Tools of the trade 3: Control/utility/game theory

How to choose the actions that are most beneficial to the organism

Optimal control theory: continuous systems
Utility theory: discrete actions (mostly)
Game theory: assumes an evil opponent instead of random noise (robust control is an analog in the continuous case)

Bellman’s optimality principle:

Underlying observation: the optimal choice of action depends on the current state, but not on how we got to that state.

Bellman’s equations (dynamic programming):

\[ v^*(x_t) = \max_u \left\{ r(x_t, u) + E_{x_{t+1}} v^*(x_{t+1}) \right\} \]
\[ u^*(x_t) = \arg \max_u \left\{ r(x_t, u) + E_{x_{t+1}} v^*(x_{t+1}) \right\} \]

Example: minimum-cost path in a graph

Classes of machine learning problems and algorithms

<table>
<thead>
<tr>
<th>Supervised Learning?</th>
<th>Unsupervised Learning **</th>
<th>Reinforcement Learning **</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>compute desired outputs from given inputs</td>
<td>find structure in given inputs</td>
<td>select actions so as to maximize rewards</td>
</tr>
<tr>
<td>Computation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>function approximation</td>
<td>density estimation</td>
<td>control of dynamical systems</td>
</tr>
<tr>
<td>Data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>input vectors along with the desired output vectors</td>
<td>input vectors only</td>
<td>the response of the system to previous control signals</td>
</tr>
<tr>
<td>Criterion</td>
<td></td>
<td></td>
</tr>
<tr>
<td>similarity between desired and computed outputs</td>
<td>ability to predict where new inputs are likely to be</td>
<td>reward accumulated over time</td>
</tr>
<tr>
<td>Applications</td>
<td></td>
<td></td>
</tr>
<tr>
<td>classification; regression; prediction</td>
<td>clustering; dim. reduction; feature extraction</td>
<td>control of actions; playing games; making plans</td>
</tr>
</tbody>
</table>