Coding (and computing with) Uncertainty

Maneesh Sahani

Gatsby Computational Neuroscience Unit
University College London

March 2021

Is uncertainty special?

- Current approach: relate neural activity to physical variables and then "bolt on" codes for uncertainty. Arguably, this gets things backwards.
- Selective pressure on neural processing is computational rather than representational: act effectively in response to dense sensory input over range of timescales.
- In principle, input-to-action transformations could be implemented by black-box function approximation
- . . . but, in fact, recognisable representations seem to emerge:
  - orientation, object type, phoneme, word, concept, affordance, action catagory, reward, . . .
  - activity varies systematically with small set of externally defined quantities (selectivity and invariance)
  - perhaps partitioning computation in terms of essential (causal) physical variables aides accuracy, efficiency and flexibility; and provides the substrate for unsupervised learning to accelerate learning of behaviour.

Decomposing computation

A simplified view of the the brain's computational task is to compute the values of possible actions (or control policies) given sensory history

- Each value may depend on a limited set of causal physical variables
- But: physical quantities accessed only indirectly through (integrated) sensory inputs
  - $\Rightarrow$ partial information, sensory noise, environmental stochasticity
  - $\Rightarrow$ internal representations must reflect estimates, sufficient statistics or *beliefs*.
- Accurate computation requires beliefs consistent with the calculus of probabilities.
- Thus, ultimately belief-like representation is arguably *more* natural than univalued representation.

Decomposing computation

Let X_t be a set of sensory variables at time t, Z_t a (set of) latent(s) and V_t a set of values associated with different choices of action or control policy.

We want

$$p(V_t|X_{1:t}) = \int dL_t \, p(V_t|Z_t)p(Z_t|X_{1:t})$$

where we assume Z separates V from X. (Drop time indices from here)

- Causal structure in the world suggests that Z can be broken into component variables $Z = \{z_1 \dots z_l\}$, such that each $v_k \in V$ depends on only a subset of the z_i.
- Thus, the crucial computations are to find (or approximate)

$$q(z_i|X) \approx p(z_i|X)$$

  (or joints over small subsets).
- Causal structure will also induce conditional independence is likely to extend amongst the z_i and x_j, so this computation may itself be achievable by local message passing.

Deterministic computation

- The core computations needed are:
  - Conditional marginalisation (prediction, message passing):

$$q(z_2) = \int dz_1\, p(z_2|z_1)q(z_1) = \mathbb{E}_{q(z_1)}\left[p(z_2|z_1)\right]$$

  - Action evaluation (Bayesian decision theory)

$$Q(a, b) = \mathbb{E}_{q(z)}\left[Q(a, z)\right]$$

  - Variational (EM) learning in latent variable models:

$$\theta^{\text{new}} = \text{argmax}\, \mathbb{E}_{q(z)}\left[\log p(x, z|\theta)\right]$$

These require:

- multiplication of densities (in message passing)
- computation of expectations

Generally, there is *no* reason for neural circuits to decode the density $q(z)$ from representation (although some targets of expectation may include indicator functions).

Stochastic computation

Probabilistic computation can be achieved using univalued representations and stochastic (sampling) algorithms.

Sometimes called the "sampling hypothesis". Avoids need for explicit probability representation.

- Stochastic computation may indeed be valuable in some settings:
  - accessing correlations
  - exploration
- Possibly linked to neural and behavioural variability.
- But no theoretical drive to univalued representations over distributional beliefs for latent quantities; so stochastic algorithms may equally operate on beliefs. (E.g. Sahani 2003).

Stochastic vs. deterministic algorithm is somewhat orthogonal to belief representation. Stochastic approaches only obviate the need for distributional codes in a special case.

Coding uncertainty

- Even parametrised beliefs almost always higher-dimensional than underlying variables.
- Thus, focus on population codes and firing rates:

  Population rates $\mathbf{r}_i$ (computed from S) represent belief $q(z_i)$.

  We will sometimes write $q(z_i; \mathbf{r}_i)$.
- Manipulating $q(z_i)$ experimentally is extremely difficult:
  - guess physical variable that corresponds to z_i
  - assume knowledge of learnt relationship between z_i and S – may not correspond to the narrow relationship established in an experiment
  - may not observe invariance beyond true z_i: belief likely to be affected by other physical values

  Thus, despite some attempts, the definitive experiment remains open.

Bayesian decoding?

Let activity $\mathbf{r}_i$ (computed from X) represent $q(z_i)$

- treat $\mathbf{r}_i$ as a random variable (even if deterministically derived from X)
- provided the computed belief $q(z_i) = p(z_i|X)$ it must be that $\mathbf{r}$ is a sufficient stat and $q(z_i) = p(z_i|\mathbf{r})$.
- can we then just use Bayes' rule to find the encoding?

$$q(z_i; \mathbf{r}_i) = p(z_i|\mathbf{r}_i) \propto p(\mathbf{r}_i|z_i)p(z_i)$$

  (c.f. Ma et al. discussion of "PPC")

Three problems:

- Exact inference is generally impossible, and approximation breaks the correspondence.
  - (Downstream processing cannot learn 'correct' posterior without access to z_i)
- Even if exact,

$$p(\mathbf{r}_i|z_i) = \int dX\, p(\mathbf{r}_i|X)p(X|z_i)$$

  and while we can measure $p(\mathbf{r}_i|X)$, $p(X|z_i)$ – the distribution of *all* natural stimuli compatible with a particular value of z_i – is inaccessible.
  - In particular, $p(X|z_i)$ is *not* experimentally defined distribution (unless the entire neural computation has adapted to the experimental environment perfectly).
- Cannot distinguish information content with encoding: does retinal activity "encode" everything about a visual scene?

Plausible coding schemes

Simple:

- Firing rate encoding of binary probabilities (Rao, Deneve)
- Explicit mean/variance encoding

Distributed:

- Linear density codes (NEF, Anderson Eliassmith)
- (Noisy) convolved density functions (DPC, Zemel Dayan Pouget)
- Expected value codes (DDPC/DDC, Sahani Dayan, current work)
  - exponential family mean parameters
  - current variant uses difference in expectation from prior
- Log-linear codes (Rao; also most common form of PPC, Ma Beck Latham Pouget)
  - exponential family natural parameters
- ...

- Codes are defined by mapping $q \to \mathbf{r}$ ("encoding") or $\mathbf{r} \to q$ ("decoding").
- In distributed forms, both operations depend on functions analagous to tuning curves.
- Actual form driven by learning useful $\mathbf{r}$, with implicit correspondence to q.

Linear density codes

$$q(z; \mathbf{r}) \propto \Big[\sum_a \psi_a(z) r_a\Big]_+$$

- Discussed by Anderson (90s); recent work by EliasSmith and others.
- Useful for "neural engineering framework" where operations defined on density can be mapped to basis function computations by hand.
- Computations linear in probability / density become easy.
- Encoding may be difficult.
- Basis functions ψ_a set a bound on possible precision.
- Noise in $\mathbf{r}$ enters decoder directly – suppressed if uncorrelated.

Convolved density functions

$$r_a = \Big[\int dx\, \psi_a(z) q(z)\Big]_+ = [\langle \psi_a(z) \rangle]_+$$

- "Distributional Population Code" (DPC) – Zemel, Dayan, Pouget.
- Decoding to histogram from noisy rates by maximum likelihood.
- Historically confused uncertainty and multiplicity. challenging – MaxEnt or EM-like algorithm if rates are noisy.
- Encoding can be learnt (delta rule) with access to z.
- Computations not discussed (but see DDC).

Log-linear codes

$$q(z; \mathbf{r}) \propto \exp\Big(\sum_a \psi_a(z) r_a\Big)$$

- Natural parameters of an exponential family.
- Message multiplication (e.g. cue combination) easy.
- Encoding may be difficult to learn.
- Uncorrelated noise in activities may average away.
- Basis functions set maximum log-precision.

Probabilistic Population Codes

Defined by Bayesian decoding:

$$q(z;\mathbf{r}) = p(z|\mathbf{r}) \propto p(\mathbf{r}|z)p(z)$$

but see previous discussion.

In practice, commonly assumes “Poisson-like” $p(\mathbf{r}|z)$ (expfam with linear sufficient statistic):

$$p(\mathbf{r}|z) = e^{\psi(z)^\mathsf{T}\mathbf{r} - A(z)}\nu(\mathbf{r})$$

$$\Rightarrow q(z;\mathbf{r}) \propto e^{\mathbf{r}^\mathsf{T}\psi(z)}\nu(z)$$

so gives log-linear/natural parameter encoding

- Poisson-like intuition derives from measure neural variability, but this is conditioned on S not z, and so neglects realistic $P(S|z)$.
- Neural variability conditioned on stimulus cannot sensibly be part of deterministic coding (though could reflect stochastic computation).
- Gain modulation of tuned population appears to encode changes in confidence without change in width of activity.
  - If true, consistent with observations of contrast-invariant orientation tuning in V1.
  - However, for uncertainty to be non-negligible, noise must be strongly correlated
    $\Rightarrow$ stochastic shifts in bump of activity
    $\Rightarrow$ predicted widening at greater uncertainty.

DDC computation

- Many computations depend on finding expectations wrt q.
- If the $\psi_a(z)$ form an adequate basis for the required functions of z, then these expectations can be computed as linear combinations of r_a:

$$f(z) = \sum_a \alpha_a \psi_a(z)$$

$$\Rightarrow \mathbb{E}\left[f(z)\right] = \sum_a \alpha_a \mathbb{E}\left[\psi_a(z)\right] = \sum_a \alpha_a r_a$$

- Marginalisation, value computation and some forms of learning reduce to linear operations.
- Message combination may require mapping to natural parameters.

Distributed distributional codes

$$\mathbf{r} = \langle \psi(z) \rangle_{q(z)} = \int dz\, \psi(z) q(z)$$

- Encoding essentially the same as DPC, but intepretation differs.
- Doubly-DPC (Sahani, Dayan) proposed expectation form, based on (thresholded) DPC encoding of *multiplicities*. Let $z = z(x)$ be a feature map (e.g. motion strength as function of angle).

$$\mathbf{r} = \Big\langle \underbrace{\Big[\int dx\, z(x)\phi(x)\Big]^+}_{\psi(z)} \Big\rangle_{q[z(x)]}$$

- Maxent intepretation: q maximally uncertain given constraints $\Rightarrow$

$$q(z;\mathbf{r}) \propto e^{\eta^\mathsf{T}\psi(z)}$$

  exponential family, with $\mathbf{r}$ representing the *mean* parameters.
  (DDPC paper also discussed decoding to a mixture by ML)
- Maxent interpetation may be important for unsupervised learning; but supervised learning and computation can be formulated without it.
- [Related to belief states, predictive state representations and RKHS mean embeddings]

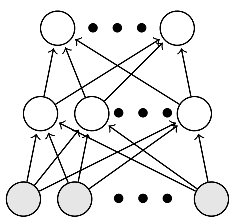
Supervised learning

- Expectations are easily learned from samples:

$$\{x^{(s)}, z^{(s)}\} \sim p(x,z)$$

- Consider a network that learns a parameter(ρ)-dependent function $R(x;\rho)$

target: $\psi(z^{(s)})$



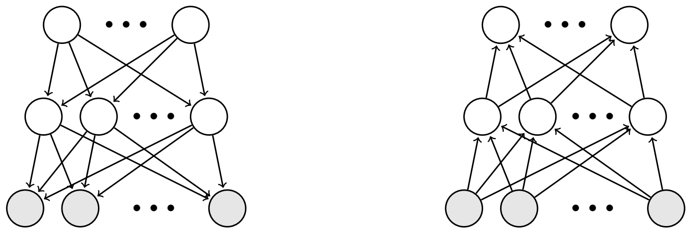
input: $x^{(s)}$

$$\Delta\rho \propto \left(\psi(z^{(s)}) - R(x^{(s)};\rho)\right) \cdot \nabla_\rho R(x^{(s)};\rho)$$

$\Rightarrow$ $R(x;\rho) \rightsquigarrow \langle \psi(z) \rangle_{p(z|x)}$.

Unsupervised learning: the Helmholtz machines

The Helmholtz Machine (Dayan et al. 1995). Approximate inference by recognition network.



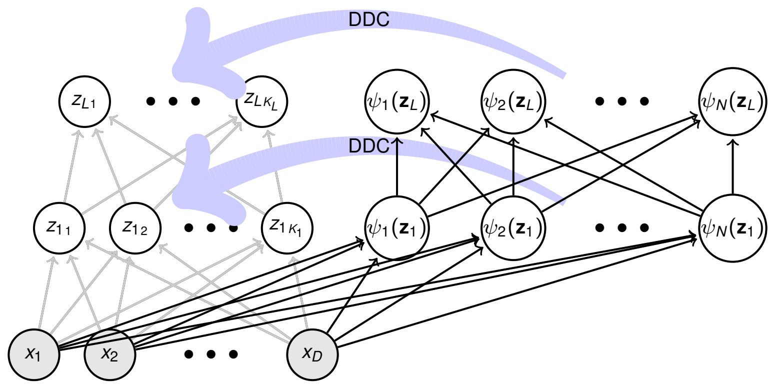
Generative or causal network
A model of the data

Recognition or inference network
Reasons about causes of a datum

Learning:

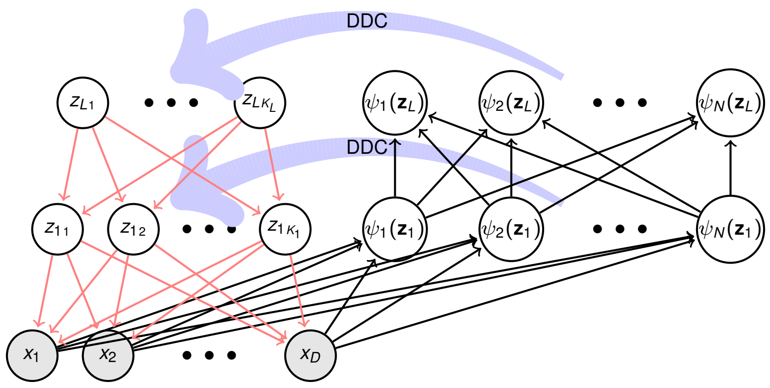
- Wake phase: estimate mean-field representation $\hat{z} = q(z) = R(x; \rho)$. Update generative parameters θ to increase a likelihood-related function F.
- Sleep phase: sample from generative model. Update recognition parameters ρ.

Distributed Distributional Recognition for a Helmholtz Machine



Wake phase – learning the model

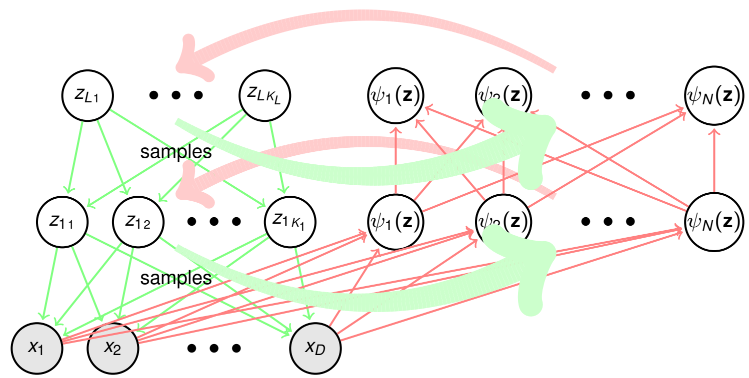
Learning requires expected gradients of joint likelihood.



$$\nabla_\theta F(\mathbf{z}_l, \theta) \approx \sum_i \gamma_l^i \psi_i(\mathbf{z}_l) \;\Rightarrow\; \langle \nabla_\theta F(\mathbf{z}_l, \theta) \rangle_q \approx \sum_i \gamma_l^i \langle \psi_i(\mathbf{z}_l) \rangle$$

Sleep phase – learning to recognise and to learn

Sleep phase simulation are used to learn the recognition model and the gradients needed for learning.



- Train ρ to map x to $\langle \psi(\mathbf{z}) \rangle_{|\mathbf{x}}$
- Train weights γ to map $\psi(\mathbf{z})$ to $\nabla_\theta F$

Wake phase

We use the recognition model and expectation transformations to update parameters from observations $x^{(n)}$:

$$
\begin{aligned}
\Delta\boldsymbol{\theta}_1 \propto \frac{\partial}{\partial\boldsymbol{\theta}_1}\langle \log p(\mathbf{x},\mathbf{z}_2,\mathbf{z}_1)\rangle_q &= \frac{\partial}{\partial\boldsymbol{\theta}_1}\left[\boldsymbol{\theta}_1\langle\mathbf{T}_1\rangle_q - \Phi_1 y(\boldsymbol{\theta}_1)\right] \\
&= \sum_i \alpha_1^i R_1(\mathbf{x}^{obs},\boldsymbol{\rho}) - \Phi_1'(\boldsymbol{\theta}_1) \\
\Delta\boldsymbol{\theta}_2 \propto \frac{\partial}{\partial\boldsymbol{\theta}_2}\langle \log p(\mathbf{x},\mathbf{z}_2,\mathbf{z}_1)\rangle_q &= \frac{\partial}{\partial\boldsymbol{\theta}_2}\left[\langle\mathbf{g}(\mathbf{z}_1,\boldsymbol{\theta}_2)\mathbf{T}_2(\mathbf{z}_2)\rangle_q - \langle\Phi_2\rangle_q\right] \\
&= \sum_i \beta_1^i[R_1]_i \sum_i \alpha_2^i[R_2]_i - \sum_i \gamma_1^i[R_1]_i \\
\Delta\boldsymbol{\theta}_x \propto \frac{\partial}{\partial\boldsymbol{\theta}_x}\langle \log p(\mathbf{x},\mathbf{z}_2,\mathbf{z}_1)\rangle_q &= \frac{\partial}{\partial\boldsymbol{\theta}_x}\left[\langle\mathbf{g}(\mathbf{z}_2,\boldsymbol{\theta}_x)\rangle_q\mathbf{T}_x(\mathbf{x}) - \langle\Phi_x\rangle_q\right] \\
&= \sum_i \beta_2^i[R_2]_i\mathbf{T}_x(\mathbf{x}) - \sum_i \gamma_2^i[R_2]_i
\end{aligned}
$$

- The added variational factorisation assumption in the second step is needed to avoid degenerate gradients.