Research projects

Li Wenliang (Kevin)

Gatsby Computational Neuroscience Unit, University College London
kevin-w-li.github.io

April 4, 2018

Topics

- General interest
  - How the brain works in perception and cognition.
  - Use machine learning as a tool to study neuroscience.
  - Inspire machine learning algorithms from studying neuroscience.
- Theoretical neuroscience
  - Perceptual learning: Wenliang and Seitz, in press
  - Neural representation of uncertainty:
  - Wenliang and Sahani, 2018
- Machine learning
  - Kernel methods (Arthur Gretton)
    - density estimation
    - goodness-of-fit test, on-going
  - Approximate inference (Maneesh Sahani), on-going

Perceptual learning

Simple (Bayesian) networks

- Data: Dosher and Lu (1998)
- Method: learning as inference
- Unexpected insight into where learning occurs?
  - How do the brain learns to classify patterns of stimulus representation (high areas)
  - How do the representations themself adapt to the task? (low areas)

Deep neural networks

- Factors that modulate transfer
- Distribution of learning over a deeper hierarchy
  - How does this distribution change w.r.t. task?
  - How much does the layers conribute to performance?
- Similarities to physiological studies (e.g. Schoups et al. 2001)

Unsupervised perceptual learning (on going)

Bayesian network: ignoring subcortical

Stimulus: Gabors at$+s/2$ and $-s/2$ generate neural representations $\mathbf{x}_{-1}$and $\mathbf{x}_{+1}$

$$
\begin{aligned}
y \in \{-1, +1\} &\sim \mathit{Bernoulli}(0.5) \\
\mathbf{x}|y &\sim \mathcal{N}\left(c\bar{\mathbf{x}}_y, \mathbf{\Sigma}(c, \bar{\mathbf{x}}_y)\right) \\
\bar{x}_{y,a} &= \exp\left[-\frac{(a + ys/2)^2}{2}\right]
\end{aligned}
$$

Discrimination: noisy linear classification using probabilistic weights

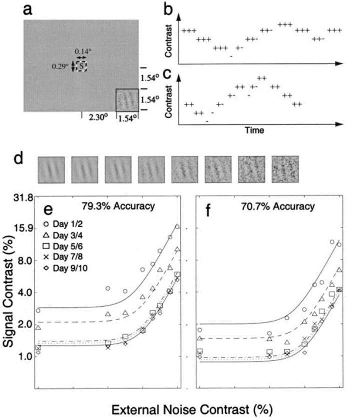
$$
\begin{aligned}
\mathbf{w} &\sim \mathcal{N}\left(\mathbf{0}, \sigma_w^2 \mathbf{I}\right) \\
b &= \phi(\mathbf{w} \cdot \mathbf{x}) \\
\hat{y} &= \mathit{Bernoulli}(b)
\end{aligned}
$$

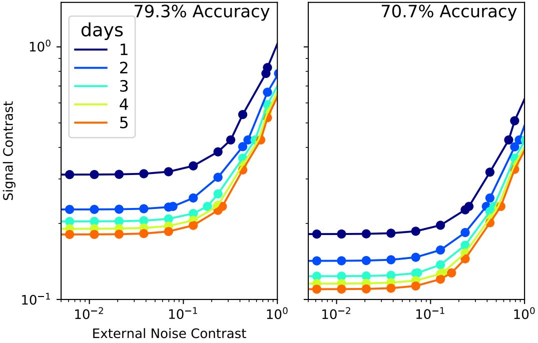
Learning: update $\mathbf{w}$ given training data $\{\hat{y}_i, \mathbf{x}_i\}$

$$
p\left(\mathbf{w} \middle| \{\hat{y}_i, \mathbf{x}_i\}_{i=1}^k\right) \propto p(\mathbf{w}) \left[p\left(\{\hat{y}_i = y_i\}_{i=1}^k \middle| \mathbf{w}\right)\right]
$$

Bayesian network: learning in high areas

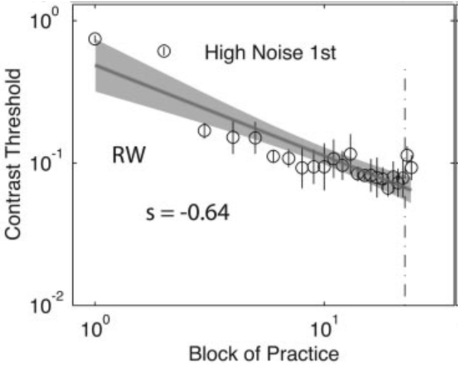
Dosher and Lu 1998

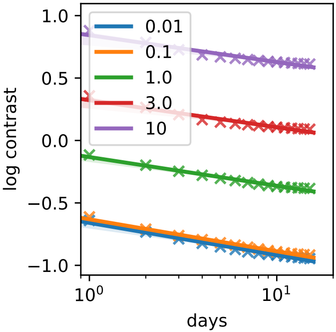




Bayesian network: results

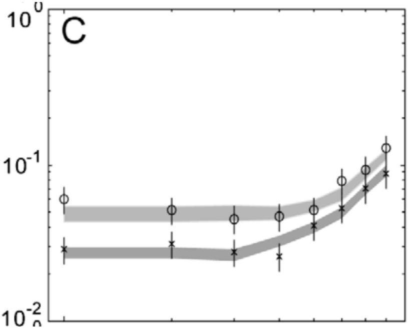
Dosher and Lu 2005

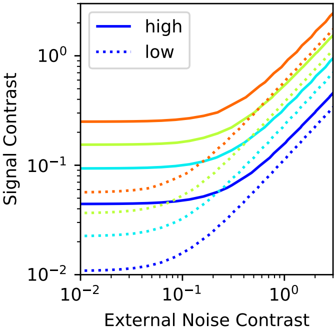




Bayesian network: asymmetric transfer

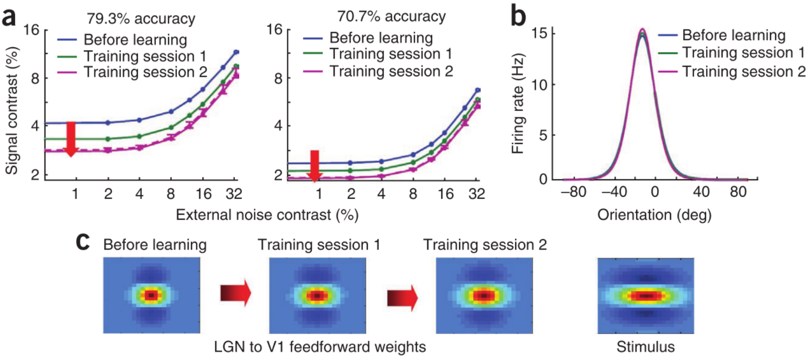
Dosher and Lu 2005





However, something is missing

- This model so far is able to qualitative reproduce the results modelled by Dosher and Lu (2010)
- Howevever, Bejjanki et al. (2011) offered another perspective: the TVC curves can be achieved by changes in subcortical connections



- **Which is correct?**

Taking into account sub-cortical pathways

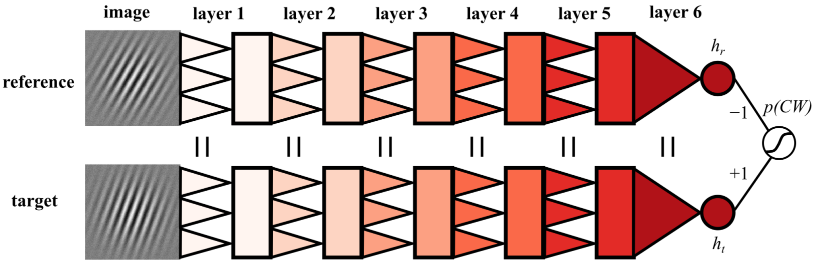
- Now we add noise directly to the images.

$$\mathbf{I_y} \sim N(c\bar{\mathbf{I}}_y, \sigma_n^2 \mathbb{I})$$
$$x_a | y \sim Poisson(\mathbf{v}_a^\intercal \mathbf{I_y})$$
$$b = \phi(\mathbf{w} \cdot \mathbf{x})$$

- The representations $\mathbf{x}$ are correlated, and changing only $\mathbf{w}$ no longer reproduce the same effect on TVC.
- However, by re-allignining $\mathbf{v}$ towards $\mathbf{I}$, we can reproduce the effect shown by Bejjanki et al. (2011).
- Note that, in Dosher and Lu (2010), independent noise are added at many intermediate stages, effectively reducing the noise correlations of the activities used for classification.
- **Hypothesis:**
  - **if neuronal noise makes the representations independent, then only higher level changes can explain data;**
  - **if not, then the representations have to change.**

Deep neural network model

- Recent similarities found in the literature (Yamins, Kriegeskorte, etc) motivates the use of DNN to investigate VPL.
- We set up a deep neural network to simulate learning of Gabor orientation discrimination under 2AFC.
- The weights are initialized from the first five layers of pre-trained AlexNet

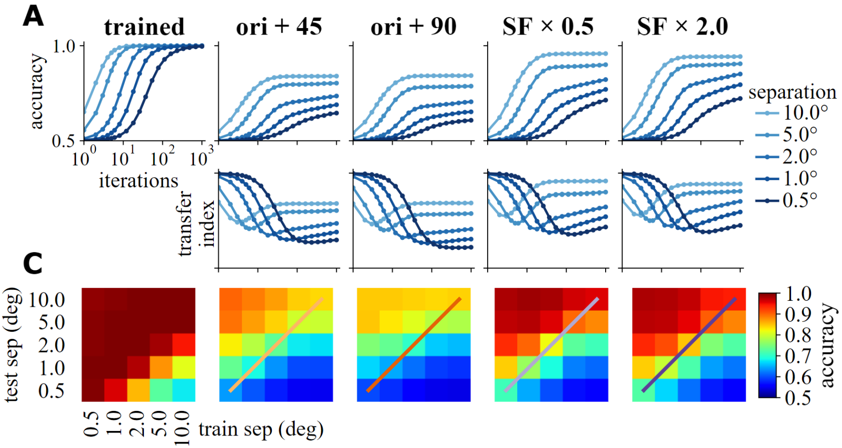


- **Questions: can this model reproduce findings of VPL?**

DNN model: Behavioral level

Behavioral performance measured as percentage correct

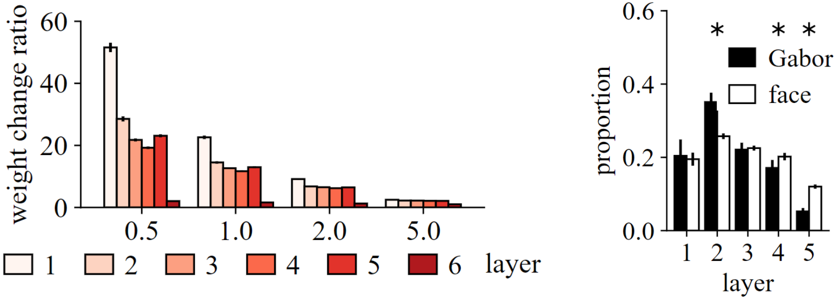
- Specificity prevails for more precise discrimination (Ahissar and Hochstein 1997)
- Test precision has a major effect while training precision may help transfer for coarser discrimination



DNN model: System level

Distribution of weight change moves towards lower layers for

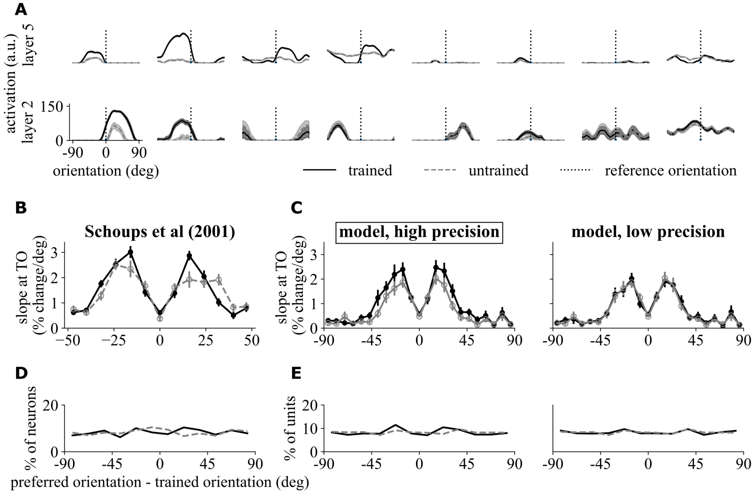
- more precise discriminations
- lower-level task (orientation vs gender)



DNN model: Neuronal level

Explore similarities between visual neurons in monkeys to the units in the DNN

- Qualitatively match without direct fitting to data



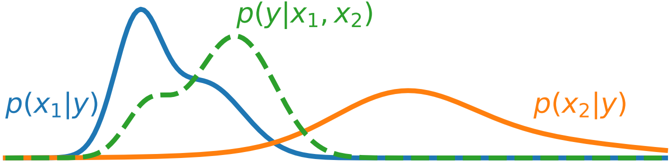
- Also reproduces double training (but with convolution)

Unsupervised perceptual learning

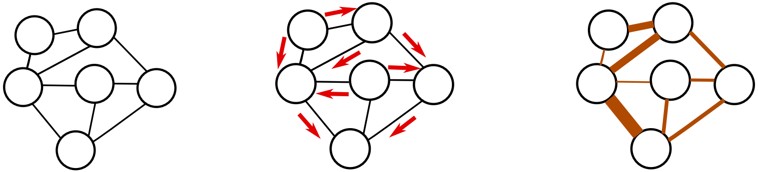
- Experiments show that perceptual learning can happen without supervision signal
  - By exposure along with a irrelevant task and stimulus (Watanabe et al. 2001)
  - By pairing with reward (Seitz et al. 2009)
  - By mental imagery (Tartaglia 2009)
  - By passive exposure in absence of attention (Amitay 2006)
- Conjectures and Approaches
  - Use unsupervised learning methods to obtain features/representations (ICA)
  - Explanation by long-term adaptation (Harris et al. 2012)

Neural representation of uncertainty

- Humans and animals are able to process uncertain information as if they are able to manipulate the underlying probability distributions



- How might the brain achieve this? Three problems need to be solved

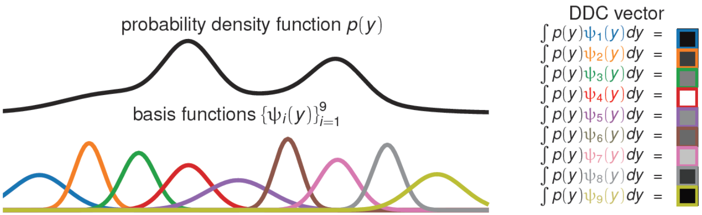


Representation Computation Learning

Distributed distributional representation (DDC)

We propose the DDC scheme for representing and computing with distributions

- The mean rate of a neuron encode the expectation of some nonlinear functions $\psi(\cdot)$ under the probability distribution encoded.



(cf. distributional population codes Sahani and Dayan 2003; Zemel et al. 1998)

- Expectations define the underlying distribution by maximum entropy,
  - e.g. for Gaussians, only need: $[r_1, r_2] = [\mathbb{E}_{p(y)}[y], \mathbb{E}_{p(y)}[y^2]]$
  - In general, these expectations define the **mean parameters** of an exponential family distributionvspace-0.5em

$$\mu_y := [r_1, r_2, \ldots, r_k] \overset{\text{max. ent.}}{\Longrightarrow} p(y) \propto \exp\left(\sum_i \theta_i \psi_i(y)\right)$$

Computation and learning with DDC

Usually, the quantity of interest is the **expectation** of some other function $g(y)$

- In prediction/regression, the quantity that minimises the squared loss is the **expectation** under the predictive distribution.
- In reinforcement learning, the value function $Q(s, a)$ is an **expectation** over the world

Using DDC, approximating an expectation is straightforward

$$g(y) \approx \sum_i \alpha_i \psi_i(y) \Rightarrow \mathbb{E}_{p(y)}\left[g(y)\right] \approx \sum_i \alpha_i \mathbb{E}_{p(y)}\left[\psi_i(y)\right] = \sum_i \alpha_i r_i$$

Bayes rule in densities corresponds to linear regression in DDC

$$p(y|x) \propto p(y)p(x|y) \Leftrightarrow \mathbb{E}_{p(y|x)}[\boldsymbol{\psi}(y)] = \mathcal{W}\boldsymbol{\psi}(x)$$

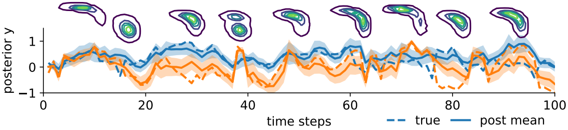
where $\mathcal{W} = \mathbb{E}[\boldsymbol{\psi}(y) \otimes \boldsymbol{\phi}(x)]\mathbb{E}[\boldsymbol{\phi}(x) \otimes \boldsymbol{\phi}(x)]^{-1}$ estimated from training data

Automatic appearance in neural network

A tracking task: estimate y_t given all available data $x_{1:t}$

$y_t \sim \text{Normal}(f(y_{t-1}), \sigma^2)$

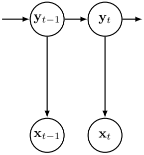
$x_t \sim \text{Poisson}(g(y))$



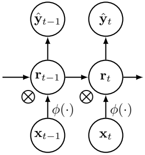
To minimise the mean squared error, the solution is $\mathbb{E}_{p(y_t|x_{\mathbf{1}:t})}[y_t]$
Inference requires belief updating (c.f. kernel Bayes rule):

$$p(y|x_{1:t}) \propto p(y|x_{1:t-1})p(y_t|y_{t-1})p(x_t|y_t) \Leftrightarrow \mu_y = \sum_{jk} W_{ijk}\mu_{y_{-\mathbf{1}},j}\phi_k(x)$$

This defines a Bilinear RNN. So far there has been no constructions of $\psi(y)$



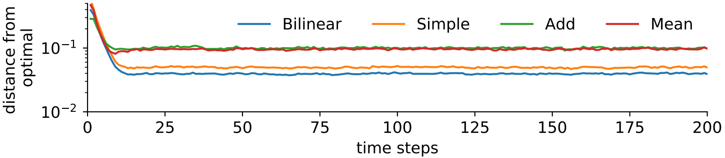
Generative



Bilinear

Automatic appearance in neural network

The DDC neural network should perform well on the task

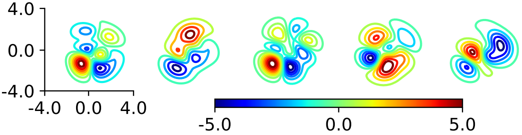


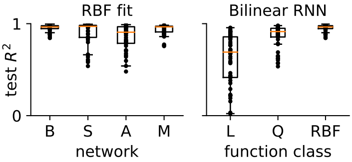
The neurons adopt DDC representation

- activation equals posterior expectation of some nonlinear function $\psi(y_t)$

$$r_t^{(i)} \stackrel{?}{=} \mathbb{E}_{p(y_t|x_{1:t})}\left[\psi_i\left(y_t\right)\right]$$

- $\psi(y_t)$ found by function approximation (RBF)





Machine learning

- Kernel methods for
  - density estimation: score matching with a neural network
  - goodness of fit: benchmarking implicit models

Density estimation with exponential family distributions

Fit a probability distribution for data $x \sim p(x)$

- log density function: infinite dimensional exponential family

$$q(x) = \frac{1}{Z} \exp\left[f(x)\right]$$

- using score matching (Hyvärinen 2005)

$$\min_f \mathbb{E}_{p(x)} \left[\|\partial_x \log p(x) - \partial_x \log q(x)\|_2^2 \right]$$

  - Regularization reduces overfitting and also avoids spurious modes in $q(x)$

We find $f(\cdot)$ in a reproducing kernel Hubert space (RKHS)

- Kernel Lite (Strathmann et al. 2015)

$$f(x) = \sum_m \alpha_m k(x_m, x)$$

- **Deep kernel lite**

$$f(x) = \sum_m \alpha_m k_\theta(x_m, x)$$

$$k_\theta(x_m, x) = k(g_\theta(x_m), g_\theta(x))$$

Goodness of fit test

- In benchmarking implicit models, it is difficult to evaluate performance using log likelihoods
- We would like to utilize an attractive property of the Stein operator $\mathcal{T}_p$ acting on a function f. Define Stein operator as

$$\mathcal{T}_p[f] = \partial_x \log p(x) \cdot f(x) + \partial_x f(x)$$

  then given another distribution q the following holds if and only if $q = p$

$$\mathbb{E}_q\left[\mathcal{T}_p[f]\right] = 0$$

- So given data from distribution q, e.g. from a generative model
  - Variational autoencoder (VAE)
  - Generative adversarial network (GAN)
- And if the data are from a differentiable process resulting in true density p, then we can try to learn an operator $\mathcal{T}_p$ to perform a statistical test. (Chwialkowski et al., 2016)
- Our approach
  - Construct a true model density p
  - Train generative models to obtain p
  - Evaluate $\mathbb{E}_q\left[\mathcal{T}_p[f]\right]$ where f can be learned to maximise statistical power