Towards Biologically Plausible Deep Learning

Yoshua Bengio
February 20, 2015
MILA, U. Montreal

Yoshua Bengio, Dong-Hyun Lee, Jorg Bornschein, and Zhouhan Lin,
ICML 2015 Submission, arXiv 1502.04156
Neuroscience 101

- Neurons
- Axons
- Dendrites
- Synapses
- No clock
- Real time
What is the brain’s learning algorithm?
Cue: Spike-Timing Dependent Plasticity

- Observed throughout the nervous system, especially in cortex
- STDP: weight increases if post-spike just after pre-spike, decreases if just before.
Machine Learning Interpretation of Spike-Timing Dependent Plasticity

- First suggested by Hinton 2007: this corresponds to temporal derivative filter applied to post-spike, around pre-spike.
- We argue
- (1) this corresponds to
\[ \Delta W_{ij} \propto S_i \Delta V_j \]

\[ \Delta V_j \approx \frac{\partial J}{\partial V_j} \]

- (2) which would be SGD on objective $J$ if
- (3) which corresponds to neural dynamics implementing a form of inference wrt $J$ seen as a function of parameters and latent vars
**STDP and Variational EM**

- Neural dynamics moving towards “improved” objective $J$ and parameter updates towards the same $J$ corresponds to a variational EM learning algorithm,

$$
\log p(x) \geq E_{q^*}(H|x) [\log p(x, H)]
$$

Approximate inference

- where $J =$ regularized joint likelihood of observed $x$ and latent $h$

$$
J = \log p(x, h) + \alpha \log q(h|x)
$$

Generative model
All interactions between neurons

Inference initial guess (forward pass)

- Generalizes PSD (Predictive Sparse Decomposition) from (Kavukcuoglu & LeCun 2008)
Inference Decouples Deep Net Layers

- After inference, no need for back-prop because the joint over layers decouples the updates of the parameters from the different layers:

  \[ p(x, h) = p(x|h^{(1)}) \left( \prod_{k=1}^{M-1} p(h^{(k)}|h^{(k+1)}) \right) p(h^{(M)}) \]

  \[ q(h|x) = q(h^{(1)}|x) \prod_{k=1}^{M-1} q(h^{(k+1)}|h^{(k)}) \]

- So \( J \) is of the form

  \[ J = \sum_k \log p(h^{(k)}|h^{(k+1)}) + \log q(h^{(k+1)}|h^{(k)}) \]
Iterative inference, e.g. MAP

Initialize $h \sim q(h|x)$

for $t = 1$ to $T$ do

$h \leftarrow h + \delta \frac{\partial J}{\partial h}$

Involves $\frac{\partial J}{\partial h}$ which has terms of the form

$$\frac{\partial \log p(h^{(k-1)}|h^{(k)})}{\partial h^{(k)}}$$

to change upper layer to make lower layer value more probable (or the equivalent for $q$)
But Inference Seems to Need Backprop

How to back-prop through one layer without explicit derivatives?

DIFFERENCE TARGET-PROP

Result: iterative inference climbs $J$ even though no gradients were ever computed and no animal was harmed!
Difference Target-Prop Estimator

- If the encoder is $f(x) + \text{noise}$ and the decoder is $g(h) + \text{noise}$, then

$$\frac{\partial \log p(x|h)}{\partial h} \approx \frac{f(x) - f(g(h))}{\sigma_h^2}$$

- which is demonstrated by exploiting

$$\log p(x|h) = \log p(x, h) - \log p(h)$$

- and the DAE score estimator theorem

$$\frac{r(x) - x}{\sigma^2} \rightarrow \frac{\partial \log p(x)}{\partial x}$$

- Considering two DAEs, one with $h$ as “visible” and one with $(x,h)$
Decomposition of the gradient into reconstructions

- We want
  \[
  \frac{\partial \log p(x|h)}{\partial h} = \frac{\partial \log p(x, h)}{\partial h} - \frac{\partial \log p(h)}{\partial h}
  \]

- which we get from two auto-encoders:
  1. The \((x,h)\) to \((h,x)\) AE: \(r(x, h) = (g(h), f(x))\)
     \[
     \rightarrow \frac{f(x) - h}{\sigma^2} \approx \frac{\partial \log p(x, h)}{\partial h}
     \]
  2. The AE with \(h\) as « visible » and \(x\) as « representation »
     \[
     \rightarrow \frac{f(g(h)) - h}{\sigma^2} \approx \frac{\partial \log p(h)}{\partial h}
     \]
- Result:
  \[
  \frac{\partial \log p(x|h)}{\partial h} \approx \frac{f(x) - f(g(h))}{\sigma^2_h}
  \]
Same Formula justifies Backprop-free Auto-Encoder based on Target-Prop

- If \( r(x) = f(g(h)) \) is smooth and makes a small move away from \( x \), then applying \( r \) from

\[
\tilde{x} = x - \Delta x = x - (g(f(x)) - x) = 2x - g(f(x))
\]

- should approximately give \( x \), so \( g(\tilde{h}) \approx x \)

- where

\[
\tilde{h} = f(\tilde{x}) = f(2x - g(f(x)))
\]

- And the encoder should be trained on the pair \((\tilde{x}, \tilde{h})\)
Difference Target-Prop for Inexact Inverse

- Make a correction that guarantees to first order that the projection estimated target is closer to the correct target than the original value

\[ \hat{h}_{i-1} = h_{i-1} - g_i(h_i) + g_i(\hat{h}_i) \]

- Special case: feedback alignment, if \( g_i(h) = B h \)

\[ \left\| \hat{h}_i - f_i(\hat{h}_{i-1}) \right\|^2 < \left\| \hat{h}_i - h_i \right\|^2 \]

if \( 1 > \text{max eigen value} \left[ (I - f'_i(h_{i-1})g'_i(h_i))^T (I - f'_i(h_{i-1})g'_i(h_i)) \right] \)
Obligatory MNIST Results (supervised target-prop)

Experimental Result
• We used hyper-parameters for the best valid error respectively
• Test error: 1.73%: target prop with high regression, 1.62%: difference target prop, 1.44%: back-prop, respective learning rates $\alpha=0.99$.

Experimental Result
• Left graph: Hyper-parameters for the best valid error
• Right graph: Hyper-parameters for the best training cost at 100 epoch
• Target prop is sometimes faster than back-prop though it is usually overfitting, but it can solve under-fitting problem (ex - very deep net, highly non-linear net and discrete net).

Hyper-optimizing for validation error

Hyper-optimizing for validation error

Hyper-optimizing for training error

Hyper-optimizing for training error
Targetprop can work for discrete and/or stochastic activations

Experimental Result

• We used hyper-parameters for the best valid error
  • Test error: ~2.5% (discrete networks with 3 hidden layers), ~2.5% (discrete networks with 2 hidden layers), 5~6% (just training top classifier with 2 hidden: back-prop)

Work in progress
Iterated Target-Prop Generative Deep Learning Experiments on MNIST

Generated model samples

Original examples

Inpainting starting point

Inpainted missing values (starting from noise)
What’s Next?

• Experiments only involved $p$ terms in $J$, but if there is going to be multiple modalities, we need correction signals (target prop) from above as well as from below.

• Using true gradients instead of diff targetprop yielded better final values of $J$ after each inference iteration but a worse final value of $J$ after training. Why?

• Proposed theory suggests that using only a few inference iterations should give a sufficient signal to update weights, but experiments required 10-15.

• Updates in paper did not follow the STDP framework but used final inference values as targets.