Optimizing Deep Architectures

Yoshua Bengio

December 6th, 2007

NIPS SATELLITE MEETING ON DEEP LEARNING

Thanks to: James Bergstra, Aaron Courville, Olivier Delalleau, Dumitru Erhan, Pascal Lamblin, Hugo Larochelle, Jerome Louradour, Nicolas Le Roux, Pierre-Antoine Manzagol, Dan Popovici, Clarence Simard, Joseph Turian, Pascal Vincent

PAPER AVAILABLE ON MY PAGE: Learning Deep Architectures for AI
Motivations:

- **understanding intelligence**, building AI, scaling to large scale learning of complex functions
- Decomposing problems into multiple levels
- Insufficient Depth $\Rightarrow$ poor generalization

Challenge: training deep architectures

- Greedy layerwise learning of *multiple levels of abstractions*
- Estimating the Log.Lik. gradient of RBMs
- Continuation methods for optimizing deep architectures
Machine Vision Example

- **MAN** abstraction corresponds to convoluted set of images (some very far in pixel distance)

- Biological & engineering solutions:
  - Multiple levels of representation
  - Multiple levels of computation

- Not clear which low & intermediate-level abstractions are good

- Want to learn representations at all levels

very high level representation:

\[
\begin{array}{c}
\text{MAN} \\
\text{SITTING}
\end{array}
\]

... etc ...

Slightly higher level representation

Raw input vector representation:

\[
\mathbf{V} = \begin{bmatrix}
23 & 19 & 20 \\
\mathbf{x}_1 & \mathbf{x}_2 & \mathbf{x}_3
\end{bmatrix}^{\mathbf{18}}
\]
When a function can be compactly represented by a deep architecture, it may need a very large architecture to be represented by an insufficiently deep one.
Insufficient Depth

- FFT time $O(n^2)$ with depth 1, $O(n \log n)$ with depth $\log n$
- Two-layer logic gates circuits:
  - can represent any function.
  - most functions require exponential number of gates
  - $\exists$ functions computable with a compact depth $k$ circuit, requiring exponential size with depth $k - 1$.
- Similar result holds for circuit of threshold neurons with non-negative weights.
- $\exists$ functions which require exponential size architecture for Gaussian kernel machine (but can otherwise be represented compactly).
- Most current machine learning algorithms: shallow

(Hastad, 1986; Yao, 1985; Wegener, 1987; Hastad and Goldmann, 1991; Bengio, 2007)
Until 2006, we knew no way to train a deep neural net to obtain better results than a shallow one (1 or 2 hidden layer) except for convolutional neural nets (Bengio and Le Cun, 2007).

Training seemed to get stuck in sub-optimal solutions, local minima or plateaus.

We still do not know why gradient descent works for deep convolutional nets.

We still do not fully understand why it is so difficult to optimize deep architectures by gradient-based techniques.

But DEPTH seems a necessary condition for statistical efficiency!
What happened in 2006?

Geoff Hinton, Simon Osindero and Yee-Wye Teh published a Neural Computation paper on “A fast learning algorithm for Deep Belief Nets” (2006), that introduces these ideas:

- A deep unsupervised network could be trained greedily, layer by layer.
- Each layer is an RBM modeling its inputs.
- Each layer outputs a representation of its input.
- This unsupervised net is a good initialization for a supervised net.

Presumably easier to learn within each layer than having to coordinate all the layers in a deep network.
RBMs are Universal Approximators

With enough hidden units any distribution can be represented exactly: (paper to appear by Le Roux & Bengio)

**Theorem**

Any distribution over \( \{0,1\}^n \) can be approximated arbitrary well with a RBM with \( k + 1 \) hidden units where \( k \) is the number of input vectors whose probability is not 0.

Adding one hidden unit (with proper parameters) increases log-likelihood:

**Theorem**

Let \( u \) be an arbitrary distribution over \( \{0,1\}^n \) and let \( P \) be a RBM with marginal distribution \( p \) over the visible units such that \( KL(u||p) > 0 \). Then there exists a RBM \( Q \) composed of \( P \) and an additional hidden unit, with marginal distribution \( q \) over the visible units such that \( KL(u||q) < KL(u||p) \).
Training RBMs: Contrastive Divergence

Exact computation of gradient $\frac{\partial \log P(x)}{\partial \theta}$ in RBMs is intractable, but a stochastic and biased approximation works well: Contrastive Divergence, requiring running $t$ steps of Gibbs chain.

It corresponds to truncation of a converging series:

**Theorem**

Consider converging Gibbs chain $x_1 \Rightarrow h_1 \Rightarrow x_2 \Rightarrow h_2 \ldots$ The log-likelihood gradient can be expanded in a converging series

$$\frac{\partial \log P(x_1)}{\partial \theta} = \sum_{s=1}^{t-1} \left( E \left[ \frac{\partial \log P(x_s|h_s)}{\partial \theta} \right] + E \left[ \frac{\partial \log P(h_s|x_{s+1})}{\partial \theta} \mid x_1 \right] \right)$$

$$+ E \left[ \frac{\partial \log P(x_t)}{\partial \theta} \mid x_1 \right]$$

with the terms in $s$ converging to 0 as $s \to \infty$, and the final term (in $t$) also converges to 0, as $t \to \infty$. 

Yoshua Bengio
In above expansion

- CD-k = keep leading $2k$ terms in expansion + sampling approx.
- Reconstruction error = keep FIRST term + mean-field approx.

- **Reconstruction error vs CD-1**: more bias, less variance.
- Makes sense to monitor RBM progress with reconstruction error.
- Combining the two can improve results (H. Larochelle).
It is easy to construct a DBN that is equivalent to an RBM.

Consider RBM1 $P(x, h_1)$ with weights $W$

Build transpose RBM2 $P(g_1, g_2)$ with weights $W'$

By symmetry $P(h_1) = P(g_1)$.

In RBM1 $P(x, h_1) = P(x|h_1)P(h_1) = P(x|h_1)P(g_1) = \text{DBN}$
Greedy Procedure Increases Likelihood

\[ P(\cdot) = \text{DBN probabilities.} \]
\[ Q(\cdot) = \text{lower-level RBM prob.} \]

Previous justification for greedy procedure:

\[
\log P(x) \geq \sum_h Q(h|x) \left( \log P(h) + \log P(x|h) \right)
\]

Keeping \( P(x|h) = Q(x|h) \) and \( Q(h|x) \) fixed (first RBM), train \( P(h) \) with samples from

\[
P^*(h) = \frac{1}{n} \sum_{t=1}^{n} Q(h|x = x_t)
\]

i.e. draw \( h \sim Q(h|x) \) as training example for 2nd RBM.
Variational bound does not guarantee that adding 3rd level helps!

However

$$\log P(x) =$$

$$KL(Q(h|x)\|P(h|x)) + H_{Q(h|x)} + \sum_h Q(h|x) \left( \log P(h) + \log P(x|h) \right)$$

Training higher levels moves $P(h)$ from $Q(h)$ towards $P^*(h)$ which moves $Q(h|x)$ away from $P(h|x)$

$\Rightarrow$ increase KL

$\Rightarrow$ increase Log.Lik.(DBN)

$\Rightarrow$ CONJECTURE: ADDING LEVELS KEEPS IMPROVING THE LOG.LIK.
Might need **global random search** to find **good local minima**.

Culture and language make human species = *optimization machine for space of abstractions*:

- Abstractions that have worked taught through **language**.
- Each human explores different solution in belief space
- Language = *low-capacity discrete channel* to hint at the good abstractions (high-level & discrete)
- Evolution & selection of ideas: **Dawkins’ memes**

Simulations show it helps training (Hutchins and Hazlehurst, 1995)
Learning high-level abstractions fundamentally difficult

First teach the lower-level concepts and once they are mastered show more advanced concepts based on previously learned ones.

Humans use this trick.

Why should we expect computers to need less help?

Future AI research may require to design an appropriate sequence of learning tasks for AIs.
Continuation Methods: optimize a sequence of gradually less smooth cost functions leading to target cost function.
The greedy layer-wise approach to training DBNs is a discrete continuation method.

Hypothesis: training 1 RBM is easier than training the DBN

Adding each layer *removes a constraint*. Continuation parameter: (number of levels, amount of tying)

- Current greedy layerwise algorithm = discrete sequence
- Can be made continuous: introduce decreasing penalty on tying
Stochastic gradient descent from small parameters is nearly a continuation method.

It would be one exactly if tracking regularization path

Continuation parameter: $||\theta||$ or regularization coefficient

- multi-layer net: obviously convex when $\theta \to 0$
- RBM: non-obvious but also true ($||\theta|| \leftrightarrow$ temperature) used successfully by R. Memisevic (PhD thesis, 2007)
A curriculum can be framed as a continuation method

Continuation parameter: $\lambda$

$P_0 = \text{very easy examples}$

$P_1 = \text{complete data distribution}$

Imagine a continuous family of distributions $P_\lambda$ over training examples s.t.

$$H(P_\lambda) < H(P_{\lambda + \epsilon}) \quad \forall \epsilon > 0$$

and $P_\lambda(x)$ is **monotone** in $\lambda$
Conclusions

- AI ⇒ learn **high level abstractions** efficiently
  ⇒ **deep architectures** (statistical efficiency)

- Optimizing deep architectures? (computational efficiency)
  - What goes wrong with deep MLPs?
  - RBMs as building blocks of DBNs.
    Contrastive Divergence approximation
  - greedy layer-wise unsupervised, more generally **continuation methods**.
These slides and review/tutorial paper on deep architectures and in particular Deep Belief Nets:

*Learning Deep Architectures for AI*

available on my web page.
Learning deep architectures for AI.
Technical Report 1312, Dept. IRO, Université de Montréal.

Scaling learning algorithms towards AI.
In Bottou, L., Chapelle, O., DeCoste, D., and Weston, J., editors, Large Scale Kernel Machines. MIT Press.

Hastad, J. (1986).
Almost optimal lower bounds for small depth circuits.

On the power of small-depth threshold circuits.
Computational Complexity, 1:113–129.

Training products of experts by minimizing contrastive divergence.
Neural Computation, 14:1771–1800.

Generative models for discovering sparse distributed representations.

A fast learning algorithm for deep belief nets.

Unsupervised discovery of non-linear structure using contrastive backpropagation.
Cognitive Science, 30(4).

How to invent a lexicon: the development of shared symbols in interaction.
Extracting distributed representations of concepts and relations from positive and negative propositions.

*The Complexity of Boolean Functions*.
John Wiley & Sons.

Separating the polynomial-time hierarchy by oracles.