Learning Deep Architectures: a Stochastic Optimization Challenge?

Yoshua Bengio, U. Montreal

University of Waterloo, May 6th, 2009
Intelligence

- Intelligence = good decisions in new contexts
  = operational knowledge

- Mostly implicit knowledge

- How and where to take the knowledge?
  - Adaptation: evolution and learning
Adaptation = Evolution + Learning

- Each example/experience contributes some information
- Combine innate & learned knowledge
- Are there general principles at work?
- Learning tasks for which evolution did not prepare
What is learning?

- Extract underlying and previously unknown statistical structure, from examples

- generalize

- CAPTURE THE VARIATIONS

- DISENTANGLE THE EXPLANATORY FACTORS OF VARIATIONS
Locally capture the variations

\[ y \]
\[ f(x) \]

\[ \text{true function: unknown} \]

\[ \text{predicted} \]

\[ \text{learnt = interpolated} \]

\[ \text{test point} x \]

\[ *= \text{training example} \]
Curse of Dimensionality

To generalize locally, need examples representative of each possible variation.
Limits of local generalization: Theoretical results

- **Theorem**: Gaussian kernel machines need at least $k$ examples to learn a function that has $2k$ zero-crossings along some line.

- **Theorem**: For a Gaussian kernel machine to learn some maximally varying functions over $d$ inputs require $O(2^d)$ examples.

(Bengio & Delalleau 2007)
Distributed Representations

- Many neurons active simultaneously.
- Input represented by the activation of a set of features that are not mutually exclusive.
- Can be \textit{exponentially more efficient} than local representations
Local vs Distributed
## Nearby Words in Semantic Space

<table>
<thead>
<tr>
<th>Country</th>
<th>Word 1</th>
<th>Word 2</th>
<th>Word 3</th>
<th>Word 4</th>
<th>Word 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>France</td>
<td>Jesus</td>
<td>XBOX</td>
<td>Reddish</td>
<td>Scratched</td>
<td></td>
</tr>
<tr>
<td>Spain</td>
<td>Christ</td>
<td>Playstation</td>
<td>Yellowish</td>
<td>Smashed</td>
<td></td>
</tr>
<tr>
<td>Italy</td>
<td>God</td>
<td>Dreamcast</td>
<td>Greenish</td>
<td>Ripped</td>
<td></td>
</tr>
<tr>
<td>Russia</td>
<td>Resurrection</td>
<td>PS###</td>
<td>Brownish</td>
<td>Brushed</td>
<td></td>
</tr>
<tr>
<td>Poland</td>
<td>Prayer</td>
<td>SNES</td>
<td>Bluish</td>
<td>Hurled</td>
<td></td>
</tr>
<tr>
<td>England</td>
<td>Yahweh</td>
<td>WH</td>
<td>Creamy</td>
<td>Grabbed</td>
<td></td>
</tr>
<tr>
<td>Denmark</td>
<td>Josephus</td>
<td>NES</td>
<td>Whitish</td>
<td>Tossed</td>
<td></td>
</tr>
<tr>
<td>Germany</td>
<td>Moses</td>
<td>Nintendo</td>
<td>Blackish</td>
<td>Squeezed</td>
<td></td>
</tr>
<tr>
<td>Portugal</td>
<td>Sin</td>
<td>Gamecube</td>
<td>Silvery</td>
<td>Blasted</td>
<td></td>
</tr>
<tr>
<td>Sweden</td>
<td>Heaven</td>
<td>PSP</td>
<td>Greyish</td>
<td>Tangled</td>
<td></td>
</tr>
<tr>
<td>Austria</td>
<td>Salvation</td>
<td>Amiga</td>
<td>Paler</td>
<td>Slashed</td>
<td></td>
</tr>
</tbody>
</table>

Collobert & Weston, ICML’2008
Local vs Distributed Representations

- Debate since early 80’s (connectionist models)

- Local representations:
  - still common in neuroscience
  - kernel machines, graphical models
  - easier to interpret

- Distributed representations:
  - ≈ 1% active neurons in brains
  - exponentially more efficient
  - difficult optimization
Deep architecture in the brain

Sequence of transformations / abstraction levels
Computation performed by learned function can be decomposed into a graph of simpler operations.
Insufficient Depth

Insufficient depth
May require exponential size architecture

Sufficient depth
Compact representation
Expressive power

Good news

Universal approximator

- Logic gates (Hastad et al 86)
- Formal neurons (Hastad et al 91)

Bad news

May have exponential size

- RBF units (Bengio et al 2007)

Functions representable compactly with $k$ layers may require exponential size with $k-1$ layers

\[
\begin{array}{c}
1 \ 2 \ 3 \\
\vdots \\
1 \ 2 \ 3 \ n
\end{array}
\]
Neuro-cognitive inspiration

- Brains use a distributed representation
- Brains use a deep architecture
- Brains heavily use unsupervised learning
- Brains learn simpler tasks first
- Human brains developed with society / culture / education
Breakthrough!

Before 2006

- Failure of deep architectures

After 2006

- Train one level after the other, unsupervised, extracting abstractions of gradually higher level

Deep Belief Networks (Hinton et al 2006)
Success of deep neural networks

Since 2006

- Records broken on MNIST handwritten character recognition benchmark (Ranzato et al 2007, 2008)
- State-of-the-art beaten in language modeling (Collobert & Weston 2008)
- NSF et DARPA are interested…
- Similarities between V1 & V2 neurons and representations learned with deep nets
V1 and V2-like filters learned

- Slow features 1st layer
- RBM 1st layer
- DBN 2nd layer
- Denoising auto-encoder 1st layer
Unsupervised layer-wise pre-training

Easy

More difficult
RBM and Auto-Encoders

- Building blocks of current learning algorithms for deep architectures
- Mathematically similar
- Feedback connections for learning
- Injection of noise

![Diagram of RBM and Auto-Encoder](image)
What neuron model?

- Amount of noise / randomness in individual neuron behavior?
- Linear or higher-order computations in the dendritic tree?
- Exponentially or polynomially saturating non-linearity?
- Temporal constancy? multiple time scales?
Quadratic interactions? Sigmoid?
Back-prop in the Brain?

- The best-performing models require weight adaptation driven by gradient wrt prediction error.
- Insufficient to adapt only one layer: need to adapt many layers wrt predictive goal.
- Back-propagation of errors mostly believed to be biologically implausible.
Brain Back-prop? Hinton’s way

Hinton proposed a solution at NIPS’2007:

- requires roughly symmetric connections
- Slow time scale for predictions
- Fast time scale (temp.deriv.) for error signals
Why is unsupervised pre-training working?

- Learning mostly layer-local with unsupervised learning
- Generalizing better when many factors of variation (Larochelle et al ICML’2007)
- Deep neural nets iterative training: stuck in poor local minima (AISTATS 2009)
- Pre-training moves into improbable region with better basins of attraction, adds prior on $p(\text{input})$
- Training one layer after the other $\approx$ continuation method (Foundations & Trends in ML 2009)
Deep Training Trajectories

(Erhan et al. AISTATS 09)

Random initialization

Unsupervised guidance
Really an Optimization Problem

\[ E \left[ \frac{\partial C(x)}{\partial \theta} \right] = \frac{\partial}{\partial \theta} \int C(x)p(x)dx \]

Online learning:
Generalization = training objective

If unsupervised pre-training purely a regularizer, its effect would disappear as # examples increases

Above hypothesis contradicted by experiment
Non-convex optimization

- Humans somehow find a good solution to an intractable non-convex optimization problem.

  How?

- Guiding the optimization near good solutions
- Guiding / giving hints to intermediate layers
Continuation Methods

- Track local minima
- Final solution
- Easy to find minimum
- Heavily smoothed objective
- Target objective
Even with the correct gradient, lower layers (far from the prediction, close to input) are the most difficult to train.

Lower layers benefit most from unsupervised pre-training:
- Local unsupervised signal = extract / disentangle factors
- Temporal constancy
- Mutual information between multiple modalities

Credit assignment / error information not flowing easily?
Guiding the Stochastic Optimization of Representations

- Train lower levels first (DBNs)
- Start with more noise / larger learning rate (babies vs adults)
- Slow features / multiple time scales
- Cross-modal mutual information
- Curriculum / shaping
- Parallel search / culture, education & research
Curriculum Learning

Guided learning helps training humans and animals

Start from simpler examples / easier tasks (Piaget 1952, Skinner 1958)
Curriculum Learning

- Sequence of training distributions
- Initially peaking on easier / simpler ones
- Gradually give more weight to more difficult ones until reach target distribution

See ICML’2009 paper
Shape Recognition

First: easier, basic shapes

Second = target: more varied geometric shapes
Shape Recognition Experiment

- 3-hidden layers deep net known to involve local minima (unsupervised pre-training finds much better solutions)

- 10 000 training / 5 000 validation / 5 000 test examples

Procedure:
1. Train for k epochs on the easier shapes
2. Switch to target training set (more variations)
Shape Recognition Results
Language Modeling Experiment

- **Objective**: compute the score of the next word given the previous ones (ranking criterion)
- **Architecture of the deep neural network** (Bengio et al. 2001, Collobert & Weston 2008)
Language Modeling Results

- Gradually increase the vocabulary size (dips)
- Train on Wikipedia with sentences containing only words in vocabulary
Parallelized exploration in brain space

- Each brain explores a potential solution
- Instead of exchanging synaptic configurations, exchange ideas through language
Memes

<table>
<thead>
<tr>
<th>Genetic Algorithms</th>
<th>Evolution of ideas</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population of candidate solutions</td>
<td>Brains</td>
</tr>
<tr>
<td>Recombination mechanism</td>
<td>Culture and language</td>
</tr>
</tbody>
</table>
Conclusions

- Shallow architectures and local generalization are insufficient to represent complex functions efficiently.
- Deep distributed architectures could not be trained before 2006.
- Now understand it is a non convex optimization problem connected to credit assignment for deeper layers.
- Many successful algorithms proposed:
  - Optimizing easier proxys (continuation methods)
  - Guiding the learning of intermediate representations