Artificial Intelligence and the Singularity

piero scaruffi

www.scaruffi.com

October 2014 - Revised 2016

"The person who says it cannot be done should not interrupt the person doing it" (Chinese proverb)
Piero Scaruffi

- piero scaruffi
  p@scaruffi.com
  scaruffi@stanford.edu

Intelligence is not Artificial

Why the Singularity is not coming any time soon and other Meditations on the Peri-Human Condition and the Future of Intelligence

Thinking about Thought
the structure of life and the meaning of matter

BRAIN

Volume 1
in the "Thinking about Thought" series

Piero Scaruffi

A History of Silicon Valley
1900-2015

Almost a third edition 2015 update

Olivetti AI Center, 1987
Piero Scaruffi

- Cultural Historian
- Cognitive Scientist
- Blogger
- Poet
- www.scaruffi.com
This is Part 2

- See http://www.scaruffi.com/singular for the index of this Powerpoint presentation and links to the other parts

1. Classic A.I. - The Age of Expert Systems
2. The A.I. Winter and the Return of Connectionism
3. Theory: Knowledge-based Systems and Neural Networks
4. Robots
5. Bionics
6. Singularity
7. Critique
8. The Future
9. Applications
10. Machine Art
11. The Age of Deep Learning
12. Natural Language Processing
The A.I. Winter and the Return of Connectionism
Neural Networks

COMPUTER SCIENCE

ARTIFICIAL INTELLIGENCE

MACHINE LEARNING

NEURAL NETWORKS

DEEP LEARNING
Neural Networks

3 LAYERS

MANY LAYERS ("DEEP")
1979: Kunihiko Fukushima’s “Neocognitron”: the birth of convolutional neural networks

Based on the cat’s visual system:

RECEPTIVE FIELDS, BINOCULAR INTERACTION AND FUNCTIONAL ARCHITECTURE IN THE CAT’S VISUAL CORTEX

BY D. H. HUBEL AND T. N. WIESEL
From the Neurophysiology Laboratory, Department of Pharmacology
Harvard Medical School, Boston, Massachusetts, U.S.A.

(Received 31 July 1961)
Neural Networks

1979: Kunihiko Fukushima’s “Neocognitron”

\[ u_{Sl}(k_l, n) = r_l \cdot \varphi \left[ 1 + \frac{\sum_{k_{l-1} = 1}^{K_{l-1}} \sum_{v \in S_l} a_l(k_{l-1}, v, k_l) \cdot u_{Cl-1}(k_{l-1}, n + v)}{1 + \frac{2r_l}{1 + r_l} \cdot b_l(k_l) \cdot v_{Cl-1}(n)} - 1 \right] \]

where

\[ \varphi[x] = \begin{cases} x & x \geq 0 \\ 0 & x < 0 \end{cases} \]

\[ v_{Cl-1}(n) = \sqrt{\sum_{k_{l-1} = 1}^{K_{l-1}} \sum_{v \in S_l} c_{l-1}(v) \cdot u_{Cl-1}^2(k_{l-1}, n + v)} \]

\[ \Delta a_l(k_{l-1}, v, \hat{k}_l) = q_l \cdot c_{l-1}(v) \cdot u_{Cl-1}(k_{l-1}, \hat{n} + v) \]

Convolution

Divisive normalization

Relu

Hebbian learning
Neural Networks

1981: Andrew Barto's & Richard Sutton's reinforcement learning

states are represented with $V^t$, a state at some time interval $t$ to be $s_t$, the estimate at a given time of our states to be $V(s_t)$.

$$V(s_t) = V(s_t) + \alpha \delta$$

where $\delta_t = R_t - V(s_t)$

Dopamine
Neural Networks

1982: John Hopfield’s recurrent neural network

\[ E(v) = -\frac{1}{2} \sum_{i=j} \sum_{i,j} w_{ij} v_i v_j - \sum_i I_i v_i + \sum_i \frac{1}{R_i} \int_0^{v_i} f_i^{-1}(z) dz \]

\[ \nabla E(v) = Wv + I - u / R \]
Neural Networks

1982-84: John Hopfield’s recurrent neural network

**Discrete Hopfield Network**

Update Equations:

\[ V_i(t+1) = \sigma(\sum_{j=1}^{N} w_{ij} V_j(t)) \]

Where \( \sigma() \) is the sign function, \( \text{sign()} \).

Energy function:

\[ E = -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} w_{ij} V_i V_j \]

**Continuous Hopfield Network**

\( V(t+1) \mapsto V(t) \) - discrete model

\[ \frac{dV}{dt} = f(V) \] - continuous model

Equations:

\[ \frac{du_i}{dt} = -u_i + \sum_{j=1}^{N} w_{ij} V_j + I_i \]

\[ V_i = g(\lambda u_i) = \tanh(\lambda u_i) \]

Lyapunov or “Energy” Function:

\[ E = -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} w_{ij} V_i V_j + \sum_{i=1}^{N} \int_{V_i}^{V_i} g^{-1}(V) dV - \sum_{i=1}^{N} I_i V_i \]
Neural Networks

1983: Terry Sejnowski’s and Geoffrey Hinton's Boltzmann machine

\[ P(x) = \frac{\exp(-E(x))}{Z} \]

- \( E(x) \): Energy function
- \( Z \): partition function where \( \sum_x P(x) = 1 \)

1986: Paul Smolensky's Restricted Boltzmann machine

Hopfield networks
Multidirectional data flow
Total integration between input and output data
All neurons are linked between themselves
Trained with or without supervision

Restricted Boltzmann Machines

Boltzmann distribution:

\[ P(v = v, h = h) = \frac{1}{Z} \exp(-E(v, h)) \]

\[ Z(\theta) = \sum_{h,v} \exp(-E(v, h; \theta)) \]

No interaction between hidden variables
Neural Networks

1985: Judea Pearl's "Bayesian Networks" (‘belief networks’)

\[
P(C,S,R,W,F) = P(C) P(S|C) P(R|C) P(W|R,S) P(F|R)
\]

\[
P(C,F) = \sum_S \sum_R \sum_W P(C,S,R,W,F)
\]

\[
P(F|C) = P(C,F) / P(C)
\]

E.g.:

Stanford’s Quick Medical Reference-Decision Theoretic project or QMR-DT (1991)

NASA Vista project (1992)

Vista Goes Online: Decision-Analytic Systems for Real-Time Decision-Making in Mission Control

Matthew Barry
Propulsion Systems Section
NASA/Johnson Space Center DF63
Houston, TX 77058

Eric Horvitz (PI), Corinne Ruokangas, Sampath Srinivas
Information and Decision Sciences
Rockwell International Science Center
444 High Street
Palo Alto, CA 94301

An Empirical Analysis of Likelihood-Weighting Simulation on a Large, Multiply Connected Medical Belief Network


We are developing a probabilistic reformulation of the Quick Medical Reference (QMR) system.
Neural Networks

Backpropagation

1982: David Parker (Stanford): learning-logic

1985: Yann LeCun (France)
Neural Networks

Backpropagation

1986: David Rumelhart (San Diego)

• Rummelhart network

• Neurons arranged in layers, each neuron linked to neurons of the neighboring layers, but no links within the same layer

• Requires training with supervision

equations of backpropagation

$$\delta^L = \nabla_a C \odot \sigma'(z^L)$$

$$\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l)$$

$$\frac{\partial C}{\partial b^l_j} = \delta^l_j$$

$$\frac{\partial C}{\partial w^l_{jk}} = a_k^{l-1} \delta^l_j$$
Neural Networks

Gradient Descent Methods

Real Time Recurrent Learning (1987, Tony Robinson & Frank Fallside)

Back Propagation Through Time (1988, Paul Werbos)

Fast Forward Propagation (1991, CalTech)

Green Function (1992, University of Maryland)

Block Update (1989, Ronald Williams)

Real-time recurrent learning (RTRL)

\[
\frac{\partial x_i[t]}{\partial W_{kl}^{xx}} = g'(\Xi_i[t]) \left( \delta_{ik} x_i[t] + \sum_{j=1}^{n_X} W_{ij}^{xx} \frac{\partial x_j[t-1]}{\partial W_{kl}^{xx}} \right)
\]
## Neural Networks

### Gradient Descent Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Advantages</th>
<th>Disadvantages</th>
<th>( O(\cdot) )</th>
</tr>
</thead>
</table>
| RTRL   | Computing error gradient after obtaining gradients of the network states w.r.t weights at time \( t \) in terms of those at time \( t - 1 \). | - online updating of weights  
- suitable for online adaption property applications | - large computational complexity | \( O(N^4) \) |
| BPTT   | Unfolding time iterations into layers with identical weights converts the recurrent network into an equivalent feedforward network, suitable for training with back-propagation method. | - computationally efficient  
- suitable for offline training | - not practical for online training | \( O(N^2) \) |
| FFP    | Recursive computing of boundary conditions of back-propagated gradients at time \( t = 1 \). | - on-line technique  
- solving the gradient recursion forward in time, rather than backwards. | - more computational complexity than BPTT method | \( O(N^3) \) |
| GF     | Computing the solution using the sought error gradient based on the recursive equations for the output gradients and a dot product. | - improving RTRL computational complexity  
- online method | - more computational complexity than BPTT method | \( O(N^3) \) |
| BU     | Updating the weights every \( O(N) \) data points using some aspects of the RTRL and BPTT methods. | - online method | - more computational complexity than BPTT method | \( O(N^3) \) |

Hojjat Salehinejad et al
Neural Networks

Gradient Descent Methods

Aiya-Parlos (2000, Amir Atiya & Alexander Parlos) unifies all of them

\[
\gamma(1) = -D^{-1}(0)e^T(1)
\]

(60a)

\[
\gamma(2) = -D^{-1}(1)e^T(2) + We^T(1)
\]

(60b)

\[
\vdots
\]

\[
\gamma(K) = -D^{-1}(K-1)e^T(K) + We^T(K-1).
\]

(60c)

\[
V(K) = \epsilon I + \sum_{k=0}^{K-1} x(k)x^T(k).
\]

(64)

\[
\Delta W = \eta \left[ \sum_{k=1}^{K} \gamma(k)x^T(k-1) \right] V^{-1}(K).
\]

(65)
Neural Networks

1986: Terry Sejnowski's NETtalk (based on backpropagation)
Neural Networks

1986: Hinton and Sejnowski organize the first "Connectionist Summer School" at CMU
Neural Networks

1987: Bernardo Huberman

Artificial Intelligence
Volume 33, Issue 2, October 1987, Pages 155-171

Phase transitions in artificial intelligence systems

Bernardo A. Huberman, Tad Hogg
Neural Networks

Unsupervised Learning

1986: David Zipser’s "autoencoder", an unsupervised neural network that is trained by backpropagation to output the input, or a very close approximation of it: autoencoders are actually powerful models for capturing characteristics of data.
Neural Networks

Unsupervised Learning
1986: Jeanny Herault’s and Christian Jutten’s independent component analysis
1987: Dana Ballard uses unsupervised learning to build representations layer by layer (deep learning ante-litteram)
1988: Ralph Linsker’s infomax method (information theory + neural networks)
1988: Mark Plumbey (information theory + neural networks)
Neural Networks

Reinforcement Learning

1989: Chris Watkins' Q-learning
Neural Networks

Reinforcement Learning

1992: Long-ji Lin’s “experience replay”

1992: Ron Williams’ REINFORCE, a "policy gradient method"

REINFORCE algorithm:

1. sample \( \{ \tau^i \} \) from \( \pi_\theta(a_t|s_t) \) (run the policy)
2. \( \nabla_\theta J(\theta) \approx \sum_i (\sum_t \nabla_\theta \log \pi_\theta(a_t^i|s_t^i)) (\sum_t r(s_t^i, a_t^i)) \)
3. \( \theta \leftarrow \theta + \alpha \nabla_\theta J(\theta) \)

Source: Sergey Levine
Neural Networks

1989: George Cybenko proves that neural networks can approximate continuous functions


Approximation by Superpositions of a Sigmoidal Function*

G. Cybenko†
Classifiers

1985: Ross Quinlan’s ID3
1993: Ross Quinlan’s C4.5
Classifiers

• The support-vector machine
  – 1963: Vladimir Vapnik and Alexey Chervonenkis
  – 1975: Tomaso Poggio's polynomial kernel
  – 1987: Marc Mezard and Werner Krauth's minover
  – 1991: Isabelle Guyon pattern classification with SVMs
  – 1991: Corinna Cortes's soft-margin classifier
  – 1998: Thorsten Joachims text classifier
Classifiers

- The support-vector machine

Number of papers on SVMs 1998-2014
Classifiers

1990: Robert Schapire's boosting
1993: Ross Quinlan's C4.5
1995: Tin-Kam Ho's random forests

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X$, $y_i \in Y = \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution $D_t$.
- Get weak hypothesis $h_t : X \rightarrow \{-1, +1\}$ with error
  \[ \epsilon_t = \Pr_{i \sim D_t} [h_t(x_i) \neq y_i]. \]
- Choose $\alpha_t = \frac{1}{2} \ln \left( \frac{1 - \epsilon_t}{\epsilon_t} \right)$.
- Update:
  \[ D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \left\{ \begin{array}{ll}
  e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\
  e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i
  \end{array} \right. \]
  \[ = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t} \]
  where $Z_t$ is a normalization factor (chosen so that $D_{t+1}$ will be a distribution).

Output the final hypothesis:

\[ H(x) = \text{sign} \left( \sum_{t=1}^{T} \alpha_t h_t(x) \right). \]

AdaBoost
Classifiers

1995: No Free Lunch theorems (David Wolpert)
No learning algorithm can excel at learning everything

The Lack of A Priori Distinctions Between Learning Algorithms

David H. Wolpert
The Santa Fe Institute, 1399 Hyde Park Rd.,
Santa Fe, NM, 87501, USA

March 1996

Neural Computation 8, 1341–1390 (1996) © 1996 Massachusetts Institute of Technology

No Free Lunch Theorems for Optimization

David H. Wolpert
IBM Almaden Research Center
William G. Macready
Santa Fe Institute

December 31, 1996
Classifiers

No Free Lunch in continuous domains?
Auger/Teytaud: Yes
Vose: No

Continuous lunches are free!

Submitted on 19 Sep 2007

Anne Auger
TAO Team - INRIA Futurs
LRI - Paris-Sud University

Olivier Teytaud
TAO Team - INRIA Futurs
LRI - Paris-Sud University

Reinterpreting No Free Lunch

Jon E. Rowe
School of Computer Science, The University of Birmingham, Edgbaston, Birmingham, B15 2TT, United Kingdom
J.E.Rowe@cs.bham.ac.uk

M. D. Vose
Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville, TN, 37996-3450, USA
vose@eecs.utk.edu

Alden H. Wright
Department of Computer Science, The University of Montana-Missoula Missoula, MT 59812, USA
aldenwright@fastmail.us

Abstract
The result is a general set-theoretic version of NFL which speaks to symmetries when arbitrary domains and co-domains are involved.
An Artificial Scientist?

- Can an Artificial Intelligence win a Nobel Prize?
- or file a patent for an invention?
An Artificial Scientist?

• Lawrence Hunter (2003 AAAI talk): the publication of a paper in a peer-reviewed scientific journal is a better test for human-level intelligence than the Turing Test.
An Artificial Scientist?

• Hiroaki Kitano (October 2016): "Artificial Intelligence to Win the Nobel Prize and Beyond"
• Grand challenge for AI: to develop a system capable of scientific research in the biomedical sciences that can discover something worthy of the Nobel Prize
An Artificial Scientist?

Diagram:
- Entire Hypothetical Body of Scientific Knowledge
  - Hypotheses Generated
    - False
    - Verified
  - Experiments
    - Some hypotheses require experimental verification
    - Are newly verified hypotheses consistent with current knowledge, or do they generate inconsistencies?
  - Scientific Knowledge in AI System
  - Data added to database
    - Experiments may include errors and noise
  - Knowledge Extraction
  - Papers and Databases
    - Portions of knowledge believed to be correct may in fact be false
    - Papers and databases contain errors, inconsistencies, and even fabrications

Hiroaki Kitano
Help!

• 2016: more than 1.2 million papers were published in life science journals alone, on top of the 25 million already in print
• A new article is being published every 30 seconds
• On average a scientist reads about 264 papers per year
• More than 70,000 papers have been published on the tumor suppressor p53

Piero Scaruffi is one of the victims of information overload
What is Scientific Discovery?

1962: Saul Amarel’s "An Approach to Automatic Theory Formation"
1962: Thomas Kuhn’s “The Structure of Scientific Revolutions”
1965: Karl Popper’s "The Logic of Scientific Discovery"
1966: Herbert Simon’s "Scientific Discovery and the Psychology of Problem Solving"
What is Scientific Discovery?

• The cognitive process that leads to novel, creative ideas:
  • the discovery of natural laws from experimental data
  • … i.e. the formation of theories that explain those data
  • … and then the design of experiments to confirm those theories.
What is Scientific Discovery?

• Kuhn: The history of science is a history of "paradigm shifts", of sudden realizations that old theories were wrong and a whole new way of thinking is required, a conceptual restructuring.
The two schools of A.I.

Artificial Intelligence (1956)

- Knowledge-based approach uses mathematical logic to simulate the human mind

- Neural-net approach simulates the structure of the brain

Knowledge Base
(facts, heuristics)

Inference Engine
(reasoning mechanism)

User Interface

User
Programs for Theory Formation

1970: Ed Feigenbaum and Bruce Buchanan’s Meta-Dendral
1976: Douglas Lenat's AM
1977: Pat Langley’s Bacon
1978: Peter Friedland and Mark Stefik’s Molgen
1986: Jan Zytkow’s Fahrenheit
1986: Ryszard Michalski's Abacus
1988: Deepak Kulkarni’s Kekada
1990: Raul Valdes-Perez’s Mechem
1990: Michael Sims’ IL
Programs for Theory Formation

1986: Don Swanson’s literature-based knowledge discovery
- 1,000 papers on fish oil
- 2,000 papers on Raynaud’s syndrome
Conclusion: fish oil may prevent Raynaud’s syndrome

1998: Swanson’s Arrowsmith
Programs for Theory Formation

1997: Faye Mitchell’s Daviccand
1998: Simon Colton’s HR
Programs for Theory Formation

2004: Ross King’s "robot scientist" Adam

The Automation of Science
Programs for Theory Formation

2007: Michael Schmidt’s Eureqa
Programs for Theory Formation

2009: Andrey Rzhetsky‘s “machine science”

Looking at Cerebellar Malformations through Interactomes of Mice and Humans

Science 23 Jul 2010:
Vol. 329, Issue 5990, pp. 399-400

Machine Science

James Evans and Andrey Rzhetsky
University of Chicago
Programs for Theory Formation

2009: Joshua Tenenbaum, Thomas Griffiths, Charles Kemp

Probabilistic model of theory formation
Programs for Theory Formation

Joshua Tenenbaum

Human-level concept learning through probabilistic program induction

Brenden M. Lake, Ruslan Salakhutdinov, Joshua B. Tenenbaum
Programs for Theory Formation

2014: IBM and Baylor College: KnIT
2015: Michael Levin's program
Programs for Theory Formation

The search for relevance

Sparrho
Meta

Find the science that matters to you

Discover the latest and most relevant research from more than 60,000,000 articles in over 45,000 journals
Neural Networks

1989: Alex Waibel’s time-delay networks
Neural Networks

1989: Yann LeCun's convolutional neural network LeNet-1 (backpropagation applied to convolutional networks)

Figure 1: Architecture of LeNet 1

Given functions $x(t)$ and $w(t)$, their convolution is a function $s(t)$

$$s(t) = \int x(a)w(t - a)da$$

Written as

$s = (x * w) \quad \text{or} \quad s(t) = (x * w)(t)$

Landwritten Digit Recognition with a Back-Propagation Network

AT&T Bell Laboratories, Holmdel, N. J. 07733
Neural Networks

1998: Yann LeCun's LeNet-5

Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition.

Convolutional Network:

- Input data
- Feature Extraction: F+R+N
- Feature Pooling: P
- Linear Classifier
- Object Categories

Architecture of a typical convolutional network for object recognition

From a paper by Yann LeCun

Figure 2: error rate on the test set (%).

<table>
<thead>
<tr>
<th>Model</th>
<th>Error Rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LeNet 1</td>
<td>0.9</td>
</tr>
<tr>
<td>LeNet 4</td>
<td>0.7</td>
</tr>
<tr>
<td>LeNet 4 / Local</td>
<td>1.1</td>
</tr>
<tr>
<td>LeNet 4 / K-NN</td>
<td>1.1</td>
</tr>
<tr>
<td>LeNet 5</td>
<td>1.1</td>
</tr>
<tr>
<td>Boosted LeNet 4</td>
<td>1.7</td>
</tr>
</tbody>
</table>

(LeCun, 1995)
Neural Networks

Activation functions: binary, sigmoid, tanh, ReLu, etc

<table>
<thead>
<tr>
<th>Function</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identity</td>
<td>$f(x) = x$</td>
</tr>
<tr>
<td>Binary step</td>
<td>$f(x) = \begin{cases} 0 &amp; \text{for } x &lt; 0 \ 1 &amp; \text{for } x \geq 0 \end{cases}$</td>
</tr>
<tr>
<td>Logistic (a.k.a Soft step)</td>
<td>$f(x) = \frac{1}{1 + e^{-x}}$</td>
</tr>
<tr>
<td>TanH</td>
<td>$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$</td>
</tr>
<tr>
<td>ArcTan</td>
<td>$f(x) = \tan^{-1}(x)$</td>
</tr>
<tr>
<td>Rectified Linear Unit (ReLU)</td>
<td>$f(x) = \begin{cases} 0 &amp; \text{for } x &lt; 0 \ x &amp; \text{for } x \geq 0 \end{cases}$</td>
</tr>
</tbody>
</table>
Neural Networks

Stateful networks: RNN

- 1991: Hava Siegelmann’s and Eduardo Sontag's theorem (RNN=Turing Machine)
Neural Networks

Stateful networks: RNN

• 1996: Mike Schuster and Kuldip Paliwal’s bidirectional recurrent neural networks

\[
y(t) = \frac{1}{10} \sum_{\Delta t = 10}^{t} x(t + \Delta t) \cdot \left(1 - \frac{\Delta t}{10}\right) \\
+ \frac{1}{20} \sum_{\Delta t = 0}^{19} x(t + \Delta t) \cdot \left(1 - \frac{\Delta t}{20}\right).
\]
Neural Networks

Stateful networks: LSTM

- 1997: Sepp Hochreiter's and Jeurgen Schmidhuber's Long Short Term Memory (LSTM) model

![Diagram of LSTM](image)

\[
y_{\text{out}}(t) = f_{\text{out}}(\text{net}_{\text{out}}(t)); y_{\text{in}}(t) = f_{\text{in}}(\text{net}_{\text{in}}(t));
\]

where

\[
\text{net}_{\text{out}}(t) = \sum_u w_{\text{out}_u} y^u(t-1),
\]

and

\[
\text{net}_{\text{in}}(t) = \sum_u w_{\text{in}_u} y^u(t-1).
\]

We also have

\[
\text{net}_f(t) = \sum_u w_{\text{f}_u} y^u(t-1).
\]
Neural Networks

Stateful networks: LSTM

- 2002: Music composition
- 2006: Alex Graves’ CTC for speech recognition

Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks

A First Look at Music Composition using LSTM Recurrent Neural Networks

2002

Douglas Eck
doug@idsia.ch

Jürgen Schmidhuber
juergen@idsia.ch

Alex Graves
Santiago Fernández
Faustino Gomez
Jürgen Schmidhuber

ALEX@IDSIA.CH
SANTIAGO@IDSIA.CH
TINO@IDSIA.CH
JUERGEN@IDSIA.CH
Neural Networks

• **CNN vs RNN**
  - CNN: a feed forward neural network with 4 kinds of layers: convolution layers (that extract features), ReLU layer, pooling (that reduces the dimensionality) and a fully-connected layer (that classifies)
  - RNN saves the output of a layer and feeds it back to the input, i.e. RNN considers the current input and also the previously received inputs
  - CNN takes a fixed-size inputs (typically a two-dimensional matrix) and generates fixed-size outputs (for example, a number)
  - RNN can handle arbitrary input/output lengths
  - CNNs are especially good at classification problems (e.g. object recognition)
  - RNNs are especially good at processing sequential data (e.g. speech recognition)
  - Example of a hybrid: generating the caption (description) of an image (the CNN classifies the image and the RNN generates the words)
Neural Networks

• Unsupervised learning
  – 1996: Bruno Olshausen’s sparse coding for autoencoders
  – 1992: Suzanna Becker’s imax
  – 1995: Geoffrey Hinton’s "wake-sleep" algorithm
  – 1995: Anthony Bell’s improved infomax
  – 1996: Shunichi Amari’s infomax with natural gradient
Neural Networks

• Semi-supervised learning
  – Avrim Blum’s and Tom Mitchell’s Co-training (1998)
  – Zhi-Hua Zhou’s Tri-training (2005)
Neural Networks

Reservoir Computing: deep learning made practical

- Herbert Jaeger’s echo state networks (2001)
- Wolfgang Maass and Henry Markram’s liquid state machines (2002)

Figure 1: The basic schema of an ESN, illustrated with a tuneable frequency generator task. Solid arrows indicate fixed, random connections; dotted arrows trainable connections.

\[
\tilde{s}_n = (1 - \alpha)\tilde{s}_{n-1} + \alpha f_{\text{in}}(W_{\text{in}}[b_{\text{in}}; \bar{u}_n] + W\tilde{s}_{n-1})
\]

Herbert Jaeger (2007)
Image Recognition

Image Recognition

2005: Dileep George & Jeff Hawkins’ hierarchical probabilistic model of the visual cortex

A Hierarchical Bayesian Model of Invariant Pattern Recognition in the Visual Cortex

Dileep George
Department of Electrical Engineering
Stanford University and
Redwood Neuroscience Institute
Menlo Park, CA 94305

Jeff Hawkins
Redwood Neuroscience Institute
Menlo Park, CA 94305
E-mail: jhawkins@mi.org
No need for Neural Nets

1991: The Beier-Neely method to generate fake images (used in Michael Jackson’s video “Black and White”)

Computer Graphics, 26, 2, July 1992

Feature-Based Image Metamorphosis

Thaddeus Beier
Silicon Graphics Computer Systems
2011 Shoreline Blvd, Mountain View CA 94043

Shawn Neely
Pacific Data Images
1111 Karlstad Drive, Sunnyvale CA 94089

Figure 17
A sequence from Michael Jackson’s Black or White
No need for Neural Nets

1994: Ernst Dickmanns' self-driving car drives more than 1,000 kms near the airport Charles-de-Gaulle in Paris

1997: IBM's "Deep Blue" chess machine beats the world's chess champion, Garry Kasparov
No need for Neural Nets


The anatomy of a large-scale hypertextual Web search engine

Sergey Brin, Lawrence Page

Computer Science Department, Stanford University, Stanford, CA 94305, USA

In this paper, we present Google, a prototype of a large-scale search engine which makes heavy use of the structure present in hypertext. Google is designed to crawl and index the Web efficiently and produce much more satisfying search results than existing systems. The prototype with a full text and hyperlink database of at least 24 million pages is available at http://google.stanford.edu/
No need for Neural Nets

- Statistical machine learning
  - 1990: Robert Schapire's boosting
  - 1991: Vladimir Vapnik’s SVM
  - 1993: Ross Quinlan’s C4.5
  - 1995: Tin-Kam Ho's random forests

Number of papers on SVMs 1998-2014
No need for Neural Nets

• Statistical machine learning (2016 survey)
No need for Neural Nets
1999: David Lowe’s SIFT for computer vision
No need for Neural Nets

- Computer Vision
  - Viola-Jones (2001)
  - HOG (2005)
  - DPM (2008)
No need for Neural Nets

2005: Sebastian Thrun's driverless car Stanley wins DARPA's Grand Challenge

2011: IBM's Watson debuts on a tv show
No need for Neural Nets

2011: Apple Siri (2011)

2014: Vladimir Veselov's and Eugene Demchenko's program Eugene Goostman, which simulates a 13-year-old Ukrainian boy
Unsung Heroes: The datasets

1990: Switchboard-1 Telephone Speech Corpus (TI)
1991: Continuous Speech Recognition (CSR) Corpus
1993: FERET (Army Research Lab)
1994: ORL face dataset (Olivetti)
1996: Broadcast News corpus
1999: MNIST handwritten-digit dataset (NYU)
2006: PASCAL VOC
2007: Tiny Images Dataset (MIT)
2007: Labeled Faces in the Wild (University of Massachusetts)
2009: ImageNet
2013: dataset of Atari games (University of Alberta)
2014: COCO (Microsoft)
2016: SQuAD (Stanford)
2016: MARCO (Microsoft)
Unsung Heroes: The datasets

Human Affective Datasets

<table>
<thead>
<tr>
<th>Human Affective Datasets</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEMAINE (2008)</td>
</tr>
<tr>
<td>EAGER (2013)</td>
</tr>
<tr>
<td>Cam3D (2011)</td>
</tr>
<tr>
<td>MAHNOB-HCI (2011)</td>
</tr>
<tr>
<td>RECOLA (2013)</td>
</tr>
</tbody>
</table>
Next…

• See http://www.scaruffi.com/singular for the index of this Powerpoint presentation and links to the other parts