Reservoir Computing with Emphasis on Liquid State Machines

Alex Klibisz

University of Tennessee

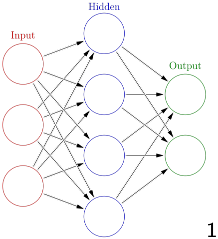
aklibisz@gmail.com

November 28, 2016

Context and Motivation

- Traditional ANNs are useful for non-linear problems, but struggle with temporal problems.
- Recurrent Neural Networks show promise for temporal problems, but the models are very complex and difficult, expensive to train.
- Reservoir computing provides a model of neural network/microcircuit for solving temporal problems with much simpler training.

Artificial Neural Networks



[1]

- Useful for learning non-linear $f(x_i) = y_i$.
- Input layers takes vectorized input.
- Hidden layers transform the input.
- Output layer indicates something meaningful (e.g. binary class, distribution over classes).
- Trained by feeding in many examples to minimize some objective function.

[1] source: wikimedia

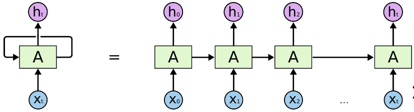
Feed-forward Neural Networks

- Information passed in one direction from input to output.
- Each neuron has a weight w for each of its inputs and a single bias b.
- Weights, bias, input used to compute the output:

$$\text{output} = \frac{1}{1+\exp(-\sum_j w_j x_j - b)}.$$

- Outputs evaluated by objective function (e.g. classification accuracy).
- Backpropagation algorithm adjusts w and b to minimize the objective function.

Recurrent Neural Networks



[2]

Figure: The network state and resulting output change with time.

- Some data have temporal dependencies across inputs (e.g. time series, video, text, speech, movement).
- FFNN assume inputs are independent and fail to capture this.
- Recurrent neural nets capture temporal dependencies by:
  1. Allowing cyclic connections in the hidden layer.
  2. Preserving internal state between inputs.
- Training is expensive; backpropagation-through-time is used to unroll all cycles and adjust neuron parameters.

[2] http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Continuous Activation vs. Spiking Neurons

How does a neuron produce its output?

- Continuous activation neurons
  1. Compute an activation function using inputs, weights, bias.
  2. Pass the result to all connected neurons.
- Spiking neurons
  1. Accumulate and store inputs.
  2. Only pass the results if a threshold is exceeded.
- Advantages
  - Proven that spiking neurons can compute any function computed by sigmoidal neurons with fewer neurons (Maass, 1997).

Conceptual Introduction

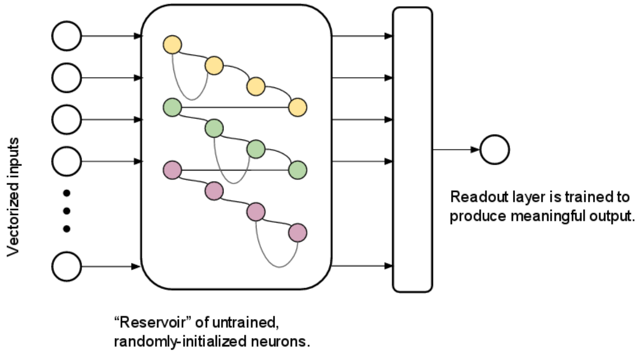


Figure: Reservoir Computing: construct a reservoir of random recurrent neurons and train a single readout layer.

History

Random networks with a trained readout layer

- Frank Rosenblatt, 1962; Geoffrey Hinton, 1981; Buonamano, Merzenich, 1995

Echo-State Networks

- Herbert Jaeger, 2001

Liquid State Machines

- Wolfgang Maass, 2002

Backpropagation Decorrelation[3]

- Jochen Steil, 2004

Unifying as *Reservoir Computing*

- Verstraeten, 2007

[3] Applying concepts from RC to train RNNs

Successful Applications

Broad Topics

- Robotics controls, object tracking, motion prediction, event detection, pattern classification, signal processing, noise modeling, time series prediction

Specific Examples

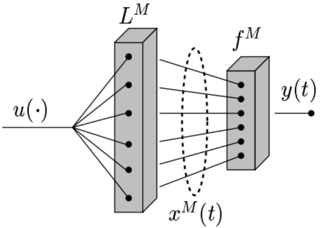
- Venayagamoorthy, 2007 used an ESN as a wide-area power system controller with on-line learning.
- Jaeger, 2004 improved noisy time series prediction accuracy 2400x over previous techniques.
- Salehi, 2016 simulated a nanophotonic reservoir computing system with 100% speech recognition on TIDIGITS dataset.

Liquid State Machines vs. Echo State Networks

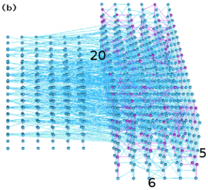
Primary difference: neuron implementation

- ESN: neurons do not hold charge, state is maintained using recurrent loops.
- LSM: neurons can hold charge and maintain internal state.
- LSM formulation is general enough to encompass ESNs.

LSM Formal Definition



[4]



[5]

- A *filter* maps between two functions of time $u(\cdot) \mapsto y(\cdot)$.
- A Liquid State Machine M defined $M = \langle L^M, f^M \rangle$.
  - Filter L^M and readout function f^M.
- State at time t defined $x^M(t) = (L^M u)(t)$.
  - *Read: filter L^M applied to input function $u(\cdot)$ at time t*
- Output at time t defined $y(t) = (Mu)(t) = f^M(x^M(t))$
  - *Read: the readout function f applied to the current state $x^M(t)$*

[4] Maass, 2002

[5] Joshi, Maass 2004

LSM Requirements Th. 3.1 Maass 2004

Filters in L^M satisfy the **point-wise separation property**

Class CB of filters has the PWSP with regard to input functions from U^n if for any two functions $u(\cdot), v(\cdot) \in U^n$ with $u(s) \neq v(s)$ for some $s \leq 0$, there exists some filter $B \in CB$ such that $(Bu)(0) \neq (Bv)(0)$.

Intuition: there exists a filter that can distinguish two input functions from one another at the same time step.

Readout f^M satisfies the **universal approximation property**

Class CF of functions has the UAP if for any $m \in N$, any set $X \subseteq \mathbf{R}^m$, any continuous function $h : X \mapsto \mathbf{R}$, and any given $\rho > 0$, there exists some f in CF such that $|h(x) - f(x)| \leq \rho$ for all $x \in X$.

Intuition: any continuous function on a compact domain can be uniformly approximated by functions from CF.

Examples of Filters and Readout Functions

Filters Satisfying Pointwise Separation Property

- Linear filters with impulse responses $h(t) = e^{-at}$, $a > 0$
- All delay filters $u(\cdot) \mapsto u^{t_0}(\cdot)$
- Leaky Integrate and Fire neurons
- Threshold logic gates

Readout functions satisfying Universal Approximation Property

- Simple linear regression
- Simple perceptrons
- Support vector machines

Building, Training LSMs

In General

- Take inspiration from known characteristics of the brain.
- Perform search/optimization to find a configuration.

Example: Simulated Robotic Arm Movement (Joshi, Maass 2004)

- Inputs: x,y target, 2 angles, 2 prior torque magnitudes.
- Output: 2 torque magnitudes to move the arm.
- 600 neurons in a 20 x 5 x 6 grid.
- Connections chosen from distribution favoring local conections.
- Neuron and connection parameters (e.g. firing threshold) chosen based on knowledge of rat brains.
- Readout trained to deliver torque values using linear regression.

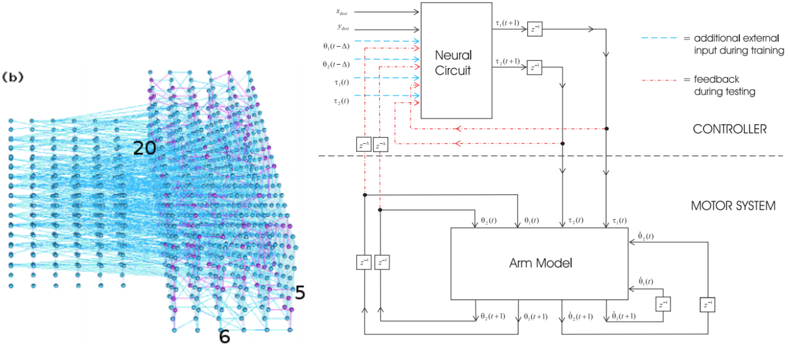


Figure: Reservoir architecture and control loop from Joshi, Maass 2004

Research Trends

1. Hardware implemenations: laser optics and other novel hardware to implement the reservoir.
2. Optimizing reservoirs: analytical insight and techniques to optimize a reservoirs for specific task. (Current standard is intuition and manual search.)
3. Interconnecting modular reservoirs for more complex tasks.

Summary

- Randomly initialized reservoir and a simple trained readout mapping.
- Neurons in the reservoir and the readout map should satisfy two properties for universal computation.
- Particularly useful for tasks with temporal data/signal.
- More work to be done for optimizing and hardware implementation.