Reservoir Computing for Neural Networks

Felix Grezes

CUNY Graduate Center

fgrezes@gc.cuny.edu

September 4, 2014

Introduction

- The artificial neural network paradigm is a major area of research within A.I., with feedforward networks having the most recent success.
- Recurrent networks offer more biological plausibility and theoretical computing power, but exacerbate the flaws of feedforward nets.
- Reservoir computing emerges as a solution, offering a generic paradigm for fast and efficient training of RNNs.
- Used in a variety of fields: NLP, computational neuroscience, robotics, machine learning and more.

Overview

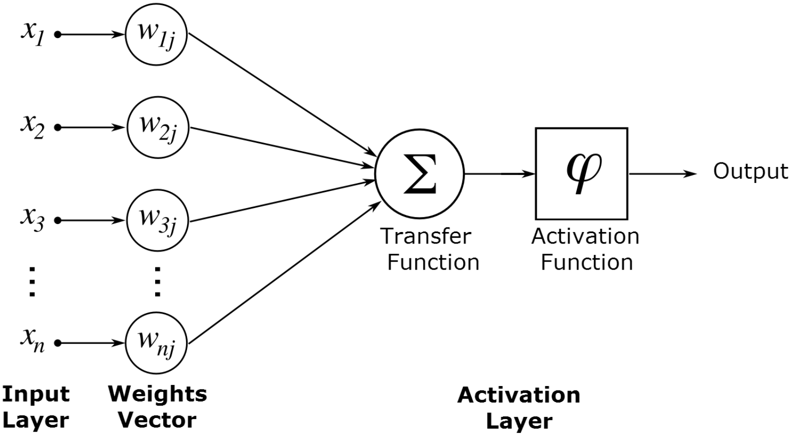
1. Introduction
2. Neural Networks
   - Short History
   - Feedforward Networks
   - Recurrent Neural Networks
3. The Reservoir Computing Paradigm
   - Models
   - Reservoir Computing Theory
4. How the Reservoir Computing Paradigm is used
   - Other Randomized Networks
5. Conclusion
   - Future Work using Reservoirs

Short History of Neural Networks

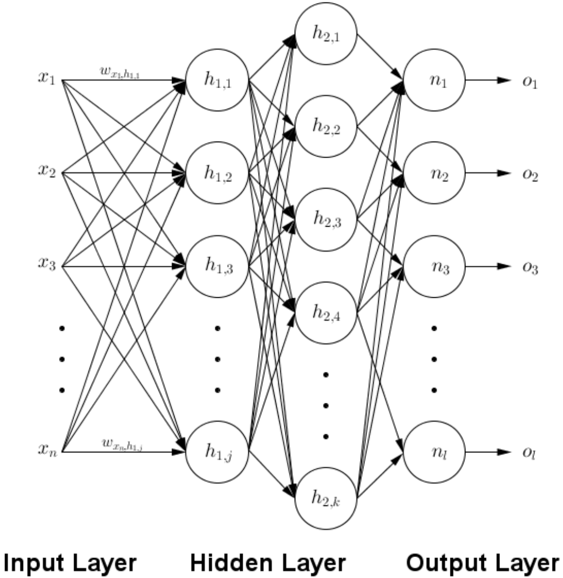
Some Important Dates

- 1890 Discovery of the biological neuron by Golgi and Ramón y Cajal.
- 1957 Rosenblatt's Perceptron:
  $f(x) = 1$ if $w \cdot x + b > 0$, and 0 otherwise.
- 1969 Minsky and Papert show that perceptron cannot learn XOR
- 1975 Werbos proposes the backpropagation algorithm, training over multiple layers of perceptrons.
- 1989/91 Cybenko/Hornik proves that multi-layer feedforward networks are universal function approximators.
- 1990 Werbos proposes the backpropagation through time algorithm for RNNs.
- 2001/02 Jaeger/Maass propose the reservoir computing paradigm, under the names of Echo State Networks/Liquid State Machines.

Feedforward Networks - The Artificial Neuron



Feedforward Networks Architecture



Feedforward Networks - Universal Approximators

In landmark results, Cybenko (1989) and Hornik(1991) proved that feedforward networks can approximate any continuous function from $\mathbb{R} \to \mathbb{R}$ under certain conditions:

- The activation function needs to be continuous, non-constant, bounded, and monotonically-increasing.
- The network contains one or more hidden layers.

These results show that it is the layered architecture that gives the networks the power to approximate any function, legitimizing the search for the most efficient learning algorithms.

Backpropagation - Werbos (1975)

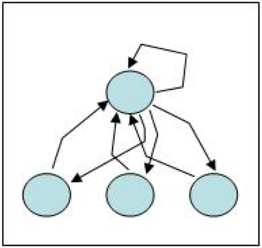
The most successful training algorithm for feedforward neural networks. It is a variant of gradient-descent and requires the activation function to be differentiable.

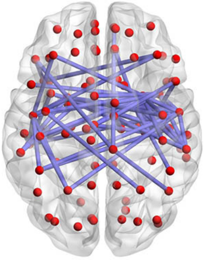
Steps:

1. Forward propagation of the input values to obtain the activation of each neuron.
2. Backwards propagation of the error for each node, in order to calculate the delta (weight update) for each weight. Computing the delta is done by using the calculus chain rule to calculate the partial derivative of the error with respect to a weight.
3. Update each weight according to the gradient and learning rate.
4. Repeat until the error over the data is below a threshold, or the gradient converges, or for a fixed number of iterations.

Recurrent Neural Networks

Recurrent Neural Networks contain cycles in the graph.





Why study recurrent neural networks?

- The human brain has a recurrent architecture. Computational neuroscientists need to understand how RNNs work.
- Because of the recurrent architecture, RNNs approximate not just functions but dynamical systems.

RNNs - Dynamical Systems

Definition

A discrete dynamical system is defined by a state $x \in X$ (state space) that evolves over discrete time-steps according to a fixed rule f.
The rule f is a function of time and the initial state x_0, and $f(x_0, t)$ it the state at time-step t.

Dynamical system are famous for their long term dependencies and sensitivity to small initial changes (Chaos Theory).

RNNs can model complex dynamical systems, allowing them to 'remember' input values over time by echoing them through the network nodes. This is particularly useful for tasks with temporal dependencies.

Feedforward networks have relied on hand-crafted data representation, e.g. n-grams in NLP tasks.

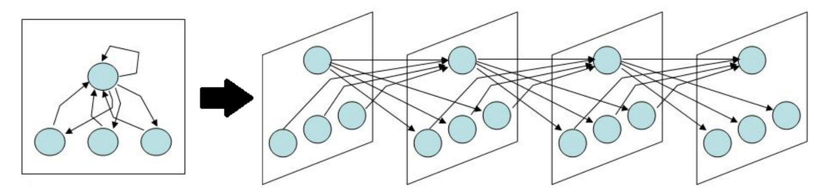
Training RNNs

Universal Approximator Theorem - Siegelmann and Sontag (1991)

Similar to feedforward networks, RNNs are universal approximators of dynamical systems, proved by emulating a universal Turing machine.

Backpropagation Through Time - Werbos (1990)

Adapts the backpropagation algorithm by 'unfolding' the RNN into a feedforward network at each time step. The related weights are tied by averaging the changes after each iteration.



The Vanishing Gradient Problem - Hochreiter (1991)

- In networks with many hidden layers, the error gradient weakens as it moves from the back of the network to the front.
- RNNs trained with BPTT exacerbate the problem since the number of layers grows over time.
- Schiller and Steil (2005) verified this effect experimentally, noticing that dominant changes during training only appeared on the output layer of the network.

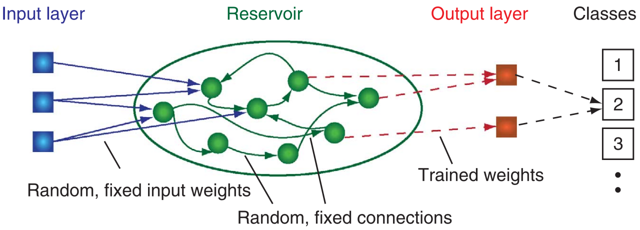
The reservoir computing paradigm emerges as a solution.

The Reservoir Computing Paradigm for RNNs

Central Idea

Since only the changes in the output layer weights are significant, then the treatment of the weights of the inner network can be *completely separated* from the treatment the output layer weights.

In many practical applications, the initial weights of the network are randomized and never changed, with only the weights of the output layer being trained, usually by a simple linear classifier such as logistic regression or the least squares.



Reservoir Models

The idea of reservoir computing was proposed independently by multiple teams in the early 2000s. Since 2008, they have been united under the reservoir term.

Echo State Networks - Jaeger (2001)

Pioneered by Jaeger and his team. Their work focused on the properties of reservoir networks that make them work, and applied them to signal processing tasks, from a machine learning angle.

Liquid State Machines - Maass (2002)

Proposed by Maass, LSMs are the other pioneer model of reservoir computing. Coming from a neuroscience background, this type of reservoir has been used to understand the computing power of real neural circuits.

Other Models

The reservoir computing paradigm can be extended beyond neural networks.

Water Basin - Fernando (2003)

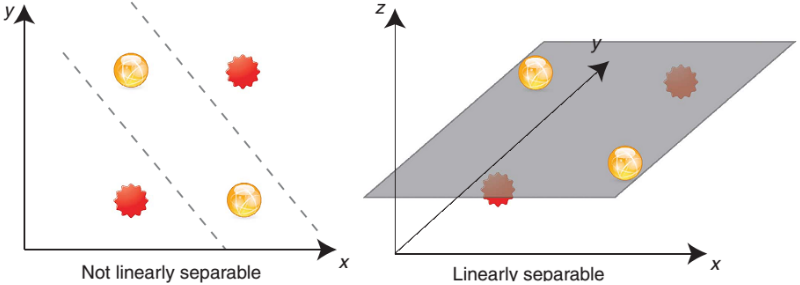
Taking the idea of a reservoir and echoes literally, an experiment was set up where the inputs were projected into a bucket of water, and by recording the waves bouncing around the liquid's surface, the authors were able to successfully train a pattern recognizer.

Bacteria Colony - Fernando(2007)

Another exotic idea for an untrained reservoir is an E.Coli. bacteria colony, with chemical stimuli as input and protein measures as output.

Why do untrained reservoirs work?

All these reservoir models rely on exploding the data dimensions in a random or untrained manner, where it can then be easily classified.



Reservoir Computing Theory - Echo State Property

Because of the network cycles, node activities can be self-reinforcing, leading to a saturation of the network. To avoid this, reservoir networks should possess the Echo State Property.

Definition: Echo State Property

The influence of past inputs and past states over the network activities should gradually vanish over time.

Heuristics

- Scale the weights of the network to a desired spectral radius of the weight matrix.
- A spectral radius smaller than 1 guarantees the echo state property.
- For practical tasks with long range time dependencies, the spectral radius should be close to 1 or slightly larger.

Reservoir Computing Theory - Separability

If two different input signals produce a similar output signal, then the output layer classifier will fail.
To produce reservoirs with a high separability power, the network should be:

- Large, allowing numerous activations.
- Sparse, making the activations decoupled.

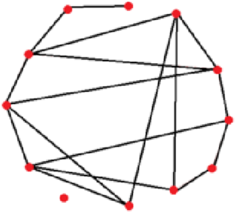
When producing random reservoirs, their separability power can be compared by measuring the distances between states of different inputs.

Reservoir Computing Theory - Topology

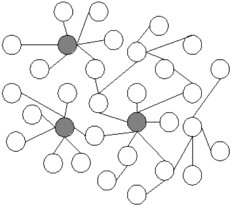
The best topology for a reservoir network remains an open question, and certainly task dependent.

A study by Liebald (2004) compared well known topologies on dynamical system prediction tasks, including small world, scale free, biologically inspired, and exhaustively searching very small networks.

No specific topology performed better than fully random networks.



Small World



Scale Free

How the Reservoir Computing Paradigm is used

Since the discovery of reservoir computing in the early 2000s, it has been applied in a number of tasks.

- Computational Neuroscience
- Machine Learning
- Speech Processing
- Physics: Hardware Implementation

Research using Reservoirs

The use of reservoir in scientific research can be broadly classified in two categories:

- Used as a generic and powerful machine learning tool.
- Used to explain and simulate realistic biological processes.

Reservoirs in Neuroscience - Dominey (1995)

- Dominey was the first to exhibit reservoir patterns in the brain, as early as 1995, before the reservoir computing was coined.
- He was the first to spell out that some parts of the brain seemed to be randomly connected and did not change over time, and that learning only happened on the output layer.
- In this early work, the activity of the pre-frontal cortex was simulated using leaky-integrator neurons and the least-mean squares output classifier.

Reservoirs in Neuroscience - Bernacchia (2011)

The authors found that monkey brains exhibit multiple timescales when processing expected rewards. Classical reinforcement learning only accounts for a fixed timescale.

What was observed

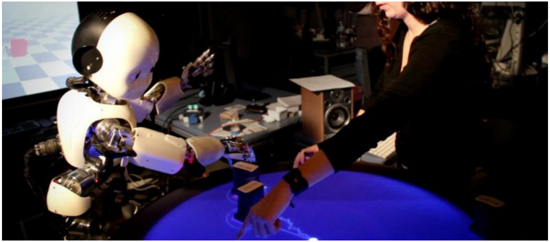
The activation of different blocks of neuron correlated with when the monkey expected a reward.

How this was reproduced

Single 1000 neuron reservoir, with sparse and random connections. The simulations showed a similar distribution of timescales in neural activation.

Reservoirs in Neuroscience - Hinaut (2014)

Combines research from neuroscience, robotics and linguistics to show that an iCub robot can learn complex grammar systems through purely associative mechanisms observed in its environment.



The iCub robot learning from its environment

By using a reservoir network modeled after the human pre-frontal cortex, the robot learns that sentences like "John hit Mary" and "Mary was hit by John" have the same meaning and same grammatical structure.

What do Reservoirs bring to Neuroscience?

Some kind of learning must be going on in the recurrent networks that are animal brains. But from a purely mechanical viewpoint, is it realistic that errors can influence every single connection between neurons?

If efficient training of RNNs is possible using the reservoir computing paradigm, it may drastically simplify the chemical mechanisms needed to explain the learning capacities of the brain.
Neuroscientists could confirm or refute this hypothesis with precise observations of neural activities.

Reservoirs in Machine Learning - Oubbati (2012)

Multi-objective Problems

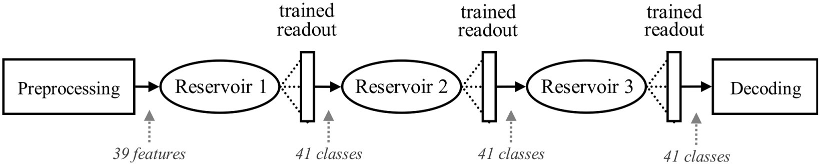
Multi-objective tasks seeks a solution that balances multiple goals at the same time: speed/quality, financial/social.

The authors used a technique called Adaptive Dynamic Programming developed to solve these multi-objective tasks, but needed a fast and powerful "critic" to compute multiple expected rewards at once.

Reservoir networks satisfy these criteria and proved capable of reaching different Pareto optimal solutions, depending on the preferences of the critic.

Reservoirs in Speech Processing - Triefenbach (2010)

Can reservoir networks compete with other state of the art techniques (HMM, Deepf Belief Nets) on the task of continuous phoneme recognition on the TIMIT dataset?



Results

The overall performance was comparable to more sophisticated approaches. The second layers improves the recognition rate by 3-5%, but subsequent layers only provide minimal gain.

What do Reservoirs bring to Machine Learning?

The reservoir computing paradigm provides scientists with a generic and powerful tool to tackle tasks that require temporal dependencies, without requiring hand-crafted data representations.

With the above early success, the path is clear for other fields to use the paradigm. Simple ideas: continuous hand-written character recognition, stock market predictions...

Hardware Implementation - Appeltant 2011

Hardware implementation are always much faster than computer simulation. However, it is both expensive to create random circuitry, and inefficient since not all random topologies possess the appropriate properties.

The authors test a tunable dynamical system, whose values are saved for a number of time-step. These delayed values act as virtual nodes of the reservoir, whose connections to the output layer a trained in classical reservoir fashion.

The Mackey-Glass oscillator

The state X evolves according to $\dot{X}(t) = -X(t) + \frac{\eta \cdot [X(t-\tau)+\gamma \cdot J(t)]}{[1+X(t-\tau)+\gamma \cdot J(t)]^p}$ with η, γ, p tunable parameters. $J(t)$ is the input vector.

Other Randomized Networks

The idea of separating the treatment of the output layer from the inner layers has also been applied to feedforward networks.

Extreme Learning Machines - Huang (2011)

- Randomize weights for the hidden layers.
- Train only weights of the output layer.

ELMs have also been proven to be universal approximators, and applied to both toy tasks and real world problems.

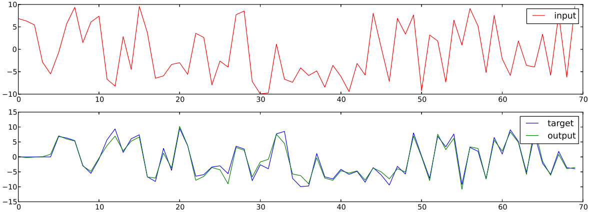
Conclusion

- The simple reservoir computing paradigm states that recurrent neural networks can be efficiently trained by optimizing only the weights connected to the output neurons, leaving the weights of internal connections unchanged.
- The reservoir computing paradigm offers a theoretically sound approach to modeling dynamical systems with temporal dependencies.
- The paradigm also promises to be much more computationally efficient than traditional RNN approaches that aim at optimizing every weight of the network.
- Reservoir computing techniques have successfully been applied to classical artificial intelligence problems, providing state-of-the-art performance on engineering problems, and offering explanations of biological brain processes.

Future Work using Reservoirs

- Speech Lab @ Queens College: speech prosody analysis.
- NSF IGERT: Multi-disciplinary work (bio-medical data)

I have already explored toy problems using reservoir networks, implemented using the OGER Python toolbox.



Results of a simple 50 neuron random reservoir on a k-delay replication task

The End

Thank You