Computational Intelligence Seminar B

Computational Aspects of Feedback in Neural Circuits

Wolfgang Maass, Prashant Joshi, Eduardo D. Sontag

presented by
Gregor Hörzer
gregor@igi.tugraz.at
Overview

- Introduction
- Theoretical Results
- Simulation Results using a Generic Cortical Microcircuit Model (GCMM)
- Conclusions
Introduction: The basic problem

- GCMMs like the liquid state machine (LSM) can only solve problems that only need rapidly fading memory.
- Basic problem: How to solve problems that need persistent (or at least very slowly fading) memory?
  - e.g. decision making on basis of accumulated evidence,
  - working memory tasks,
  - internal representation of expected rewards
Memory in GCMMs

- Usual GCMMs do not have persistent (non-fading) memory (max. 200-300 ms of fading memory when using biologically realistic parameters)
- Persistent memory can be achieved by highly increasing the weights of the synaptic connections in the model, but
  - it is almost impossible to control the information to be stored, and
  - the system tends to store unimportant details of the input stream from the distant past.
Persistent Memory using Feedback

- New Approach:
  - Train some readout neurons to extract information from the GCMM using “teacher forcing”
  - Use the output of the readout as an additional input to the GCMM (→ feedback)

- This approach creates persistent (or at least slowly fading) memory in GCMMs without the drawback of falling into the chaotic regime
Theoretical Results

- Common model of the dynamics of firing rates in recurrent circuits of neurons:

\[ x'_i(t) = -\lambda_i x_i(t) + \sigma \left( \sum_{j=1}^{n} a_{ij} x_j(t) \right) + b_i \sigma(v(t)), \quad i = 1, \ldots, n \]

- Generic form of the dynamical system:

\[ x'_i(t) = f_i(x_1(t), \ldots, x_n(t)) + g_i(x_1(t), \ldots, x_n(t))v(t), \quad i = 1, \ldots, n \]
First Theorem

For any n-th order differential equation of the form
\[ z^{(n)}(t) = G(z(t), z'(t), z''(t), \ldots, z^{(n-1)}(t)) + u(t) \]
(for arbitrary smooth functions \( G : \mathbb{R}^n \rightarrow \mathbb{R} \)) there exists

- a memoryless feedback function \( K(x(t), u(t)) : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R} \)
- and a memoryless readout function \( h(x(t)) : \mathbb{R}^n \rightarrow \mathbb{R} \) (both smooth, and continuous)
- such that for every external input \( u(t), t \geq 0 \), and each solution \( z(t) \) of the forced system (equation above), there is an input \( u_0(t) \) with \( u_0(t) = 0 \) for all \( t \geq 1 \), such that the solution \( x(t) = (x_1, \ldots, x_n) \) of the fixed system (cf. generic form of the dyn. system, previous slide)

\[ x'(t) = f(x(t)) + g(x(t))K(x(t), u(t) + u_0(t)), \quad x(0) = 0, \]
\[ f, g : \mathbb{R}^n \rightarrow \mathbb{R}^n, \text{ is such that } h(x(t)) = z(t) \text{ for all } t \geq 1. \]
Feedback Linearization/Equivalence

- Using feedback, a large class of nonlinear dynamical systems can be transformed into linear systems (→ feedback linearization).

- If two dynamical systems can be transformed into each other using a suitable feedback function and a change of basis in the state space (by applying the readout function \( h(x(t)) \)), they are denoted as “feedback equivalent”.

- The linear system
  \[ x'(t) = A_n x(t) + b_n v(t) \]
  has the universality property specified in Theorem 1, where
  \[
  A_n := \begin{pmatrix}
  0 & 1 & 0 & \cdots & 0 \\
  0 & 0 & 1 & \cdots & 0 \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  0 & 0 & 0 & \cdots & 1 \\
  0 & 0 & 0 & \cdots & 0 \\
  \end{pmatrix}, \quad b_n := \begin{pmatrix}
  0 \\
  0 \\
  \vdots \\
  0 \\
  1 \\
  \end{pmatrix}
  \]

- Feedback equivalence preserves the universality property, and one can show that every feedback linearizable system is also feedback equivalent to the system described by the equation above.
Notes on the first Theorem

● Using feedback, any dynamical system belonging to class $S_n$ has the computational power of a universal Turing machine, because any Turing machine can be simulated with a system of differential equations like the one in Theorem 1.

● A dynamical system from class $S_n$ is able to emulate any continuous dynamic response to an input stream $u(t)$ if it receives a suitable continuous feedback $K(x(t), u(t))$ and has an adequate readout function $h(x(t))$. 
Second Theorem

- Feedback allows linear and nonlinear fading memory systems with amplitude-bounded noise to process time-varying inputs in real time, having the computational capability of a Finite State Machine (FSM).

- Noise reduces the memory capacity of such circuits to a finite number of bits. One can imagine that such circuits are like a Turing machine with a tape of finite length (which is a special case of a FSM).
Notes on the second theorem

- Generic neural circuits employ persistent internal states for state-dependent processing of online input streams if a few (or only a single one for linear feedback functions \( K(x) = w x \)) neurons in the circuit are trained.

- Therefore, the dynamics of the system is not dominated by the persistent memory states.

- By using “teacher forcing” as a training method, so-called “High-Dimensional Attractors” are generated.
Training using “Teacher Forcing”

- train readouts that provide feedback first, then train the other readouts
- for feedback-providing readouts, the actual feedback was replaced by a noisy version of the target value.
- each readout was trained by linear regression
- due to teacher forcing, the high dimensional attractor properties emerge in the circuit
High-Dimensional Attractors

- Using HDAs, the given circuit does not end up in a fixed state for the whole system (like for low-dimensional attractors), but only a small part of the system is forced to follow a trajectory to a fixed state (therefore “attractor”).

- The rest of the system keeps sensible to the input (therefore “high-dimensional”).

- In other words, the system state is restricted to a submanifold \( \{ x : K(x) = c \} \), which is, for a linear \( K(x) \), only reduced by one dimension when compared to the unrestricted state space.
Applications to GCMMs

- Until this point, only models based on firing rates (mean field models) were considered, because they were more feasible for the theoretical analysis.

- Simulation parameters:
  - 600 Integrate & Fire (Exp. 1, 2) or Hodgkin-Huxley (Exp. 3) neurons
  - dynamical synapses
  - biologically realistic amounts of noise
Exp. 1: HDA with switching behavior

- 8 neurons were trained to fire at 40 Hz if a burst occurred most recently in $r1(t)$, and not to fire if it occurred in $r2(t)$ (non-fading state of a FSM).

- Other neurons were trained to combine the state of the HDA with the online input stream or to independently compute a function of the input stream.

feedback that creates a discrete attractor
Results of Experiment 1 (I)

A
Input spike trains

B
Firing rates of the 4 input streams

C
Circuit response (shown for 100 out of 600 neurons)

time (s)
Results of Experiment 1 (II)

Target behavior and performance of high-dimensional attractor

State dependent signal amplification

State dependent computational operation

Product of $r_3(t)$ and $r_4(t)$

Time (s)
Exp. 2: Parametric memory

- Hold and update an analog value (e.g. intended eye position)
- Two HDAs:
  - One times 400 ms time intervals
  - Other one times 600 ms time intervals
- Cue: spike burst indicating the beginning of an interval
Results of Experiment 2 (I)

A. External input consisting of a cue (red) and noise (black).

B. Circuit response (shown for 100 out of 600 neurons).

C. Time course of a continuous attractor with decay time of 400 msec.
Results of Experiment 2 (II)

D: Time course of a continuous attractor with decay time of 600 msec

E: Response of the same circuit with a different high-dimensional attractor

F: Time course of continuous attractor trained to reach plateau after 600 msec
Exp. 3: Analog Real-Time Computation

- continuous attractor trained to integrate the difference of the two inputs $r_1(t)$ and $r_2(t)$
- other readouts trained to compute functions of the inputs or to combine inputs with the attractor

Feedback that creates a continuous attractor $CA(t)$
(neural integrator with target value $\int_0^t (r_1(s) - r_2(s)) \, ds$)
Results of Experiment 3 (I)

A

Circuit inputs

B

Input rates

C

Circuit response (shown for 100 out of 600 conductance based HH neurons)

time (s)
Results of Experiment 3 (II)

- **D**: Activation CA(t) of continuous attractor
- **E**: Switch of second readout between $r_2(t)$ and 0, depending on CA(t)
- **F**: Computation of $r_1(t)$ – CA(t) by third readout
Conclusions

- Real-Time computation with persistent states can be performed by GCMMs with appropriate feedback without requiring
  - biologically unrealistic assumptions (e.g. symmetric weight distributions)
  - a handcrafted circuit structure for the task
  - biologically unrealistic small amounts of noise
- One single assumption needed: Adaptive procedures (synaptic plasticity) in generic neural circuits can approximate linear regression
Reference


Thank you for your attention!