Neural Networks

dependence on time

Topic 5
Note: lecture notes by Bob Keller (Harvey Mudd College, CA) are used

Main idea: include time dynamics into NN models

- So far NN have been a combinational input pattern presented at once
- How to include into the models time?
- We want to consider cases where network inputs and learned behaviour can include functions of time

Example: Time series or problems of future prediction

Applications

Signal processing
Sun-spot prediction
Predict the degradation of the ozone layer
Market analysis

Learning to mimic
Recall the Adaline training rule:

\[
\Delta W = \eta (\text{desired} - \text{actual output}) \cdot \text{input}
\]

Here input vector is the current input, along with all the delayed inputs (one per weight).

An Adaline is trained to mimic a specific input-output behavior.

The output is an attenuated version of the input.

When subsequently presented with the input, the output is observed and the error computed.

Applications in training: Matlab

```matlab
% DEFINE THE NETWORK
% NEWLIN generates a linear network.
% We will use a learning rate of 0.5, and two
% delays in the input. The resulting network will predict the next value of
% the target signal using the last two values of the input.
% lr = 0.5;
% delays = [0 1];
% net = newlin(minmax(cat(2,P{:})),1,delays,lr);
% ADAPTING THE LINEAR NEURON
% ADAPT simulates adaptive linear neurons. It takes the initial network,
% an input signal, and a target signal, and filters the signal adaptively. The
% output signal and the error signal are returned, along with new network.
% Adapting begins...please wait...
% [net,y,e]=adapt(net,P,T);
```

On-line training

- The Adaline Predictor can be trained during operation.
- At each time step, one set of weight modifications can be made.
- After a transient, the network learns to mimic the desired behavior.

How to learn to predict?
% DEFINING A WAVE FORM
% TIME1 and TIME2 define two segments of time.
% time1 = 0:0.05:4; % from 0 to 4 seconds, steps of .05
% time2 = 4.05:0.024:6; % from 4 to 6 seconds, steps of .05
% TIME defines all the time steps of this simulation.
% time = [time1 time2]; % from 0 to 6 seconds
% T defines a signal which changes frequency once:
% T = con2seq([sin(time1*4*pi) sin(time2*8*pi)]);
% The input P to the network is the same as the target.
% The network will use the last five values of the target to predict the next value.

% NEWLIN generates a linear network.
% We will use a learning rate of 0.1, and five delays in the input. The resulting network will predict the next value of the target signal using the last five values of the target.
% lr = 0.1;
% delays = [1 2 3 4 5];
% net = newlin(minmax(cat2(P{:})),1,delays,lr);
Once the network has been trained

- It can use its own output as the next input.
- That is, it can “run free”, predicting the full output sequence.
- Since the output was only an approximation, the accuracy of the predicted output will deteriorate with time.

Free – running mode

Noise-Reduction Scenario
Filter Learns to Predict the Noise

ANC Audio Demo from Ariz. State Univ.
http://www.eas.asu.edu/~dpgrad/anand/Java/ANC/ANC.html

Filters
- Classical filters don’t adapt
  (Lowpass / Highpass / Bandpass) filters
- Adaptive filters adapt
- LMS filter (least-mean-squared)
- RLS filter (recursive least squares, based on pseudo-inverse, not as stable)
- Kalman filter (based on a stochastic state space model)
- Filter form is also called a FIR (Finite Impulse Response) filter.
- In statistics, it is called an MA (Moving Average) filter.

Santa Fe Institute time series prediction

- Available at http://www.psych.stanford.edu/~andreas/TimeSeries/SantaFe.html
- Contains descriptions of the 6 data sets (A, B, C, D, E, and F) that were used in the Santa Fe Competition, directed by Neil Gershenfeld (now at MIT’s Media Lab) and Andreas Weigend (now at NYU’s Stern School of Business).
Data Set A: Laser generated data
- Reason for choice:
  - A good example of the complicated behavior that can be seen in a clean, stationary, low-dimensional non-trivial physical system for which the underlying governing equations dynamics are well understood.

Data Set B: Physiological data
- Reason for choice:
  - Heart rate variability
    - There is growing (but still controversial) evidence that the observed variations in the heart rate might be related to a low-dimensional governing mechanism, underscoring the necessity of understanding this mechanism is obviously very important in order to understand its failures (i.e., heart attacks).

Data Set C: Currency exchange rate data
- Reason for choice:
  1. Financial data
     - Predicting currency exchange rates is a classic problem in time series analysis, and is of both academic and financial interest.
     - These particular data were chosen because they were available on a tick-basis, and were representative of financial prediction tasks, but still were obscure enough that they would not be easily recognized.
  2. Multiple prediction data sets
     - We collected the predictions for 10 data sets in order to build up better statistics about how algorithms compare, and to check to see if the predictability of the exchange rate varies over time.

Data Set F: J. S. Bach's last (unfinished) fugue
- Reason for choice:
  - Fan.

Other models
- Time-Lagged Feed-Forward Networks
- Time-Delay Neural Networks (TLFF, TDNN)
- FIR-Multi-layer networks (FIRNET)
- Backpropagation through time (BPTT)
- Real-Time Recurrent Learning (RTRL)
- Elman nets, Jordan nets
- Temporal difference method (TD(λ))
An extension of the “Adaline” adaptive filter model
Use an arbitrary feed-forward net (MLP) in place of the Adaline
Train using ordinary backpropagation, analogous to LMS

Unlike TLFF input samples are not kept in explicit delay
Input fed sequentially into network
Training is as if the network were unrolled to accommodate the entire sequence of input samples.
Only one set of weights is actually used in operation; the weight changes are averaged across stages to get the actual weight change

Problem: Back up a truck so that (xtrailer,ytrailer) = (xdock, ydock), given initial values for (xtrailer, ytrailer, xcab, ycab, θtrailer, θcab)

20,000 trials required to train
16 lessons of 1000-2000 each
Initially truck positioned very close to dock and in a nearly-correct position, so controller could learn easy tasks first.
Final MSE was 3% of truck length, angle 7 degrees

The truck moves in small time increments Δ
A neural net is first trained to mimic the truck backing using real truck dynamics.
Given the current state at a time t (which includes the steering angle), the network learns to determine the next state (at time t + Δ)
This is done by starting the truck in a random state, observing the error between what the network does and the dynamic model, and adjusting the weights.
An error value is produced by comparing the desired final state with the goal.
The error value is backpropagated through the controller-truck combination to adjust the controller’s weights, using BPTT.

20,000 trials required to train
16 lessons of 1000-2000 each
Initially truck positioned very close to dock and in a nearly-correct position, so controller could learn easy tasks first.
Final MSE was 3% of truck length, angle 7 degrees

The truck moves in small time increments Δ
A neural net is first trained to mimic the truck backing using real truck dynamics.
Given the current state at a time t (which includes the steering angle), the network learns to determine the next state (at time t + Δ)
This is done by starting the truck in a random state, observing the error between what the network does and the dynamic model, and adjusting the weights.
An error value is produced by comparing the desired final state with the goal.
The error value is backpropagated through the controller-truck combination to adjust the controller’s weights, using BPTT.

20,000 trials required to train
16 lessons of 1000-2000 each
Initially truck positioned very close to dock and in a nearly-correct position, so controller could learn easy tasks first.
Final MSE was 3% of truck length, angle 7 degrees