Recurrent neural networks
Modelling sequential data
Recurrent Neural Networks 1: Modelling sequential data

Steve Renals

Machine Learning Practical — MLP Lecture 9
13 November 2018
We often wish to model data that is a sequence or trajectory through time, for instance audio signals, text (sequences of characters/words), currency exchange rates, motion of an animal.
Sequential Data

We often wish to model data that is a sequence or trajectory through time, for instance audio signals, text (sequences of characters/words), currency exchange rates, motion of an animal.

What a good model of sequential data include?

- Use of the past to predict the future
- A notion of state – the current state depends on the past, compresses the history
- The ability to share data across time

Convolutional networks model invariances across space – can we do something similar to model invariances across time?

Yes - time-delay neural networks

Can we use units to act as memories?

Yes - recurrent networks
Sequential Data

- We often wish to model data that is a sequence or trajectory through time, for instance audio signals, text (sequences of characters/words), currency exchange rates, motion of an animal.

What a good model of sequential data include?

- Use of the past to predict the future
- A notion of state – the current state depends on the past, compresses the history
- The ability to share data across time
Sequential Data

- We often wish to model data that is a sequence or trajectory through time, for instance audio signals, text (sequences of characters/words), currency exchange rates, motion of an animal
- What a good model of sequential data include?
  - Use of the past to predict the future
  - A notion of state – the current state depends on the past, compresses the history
  - The ability to share data across time
- Convolutional networks model invariances across space – can we do something similar to model invariances across time?
  - Yes - time-delay neural networks
Sequential Data

We often wish to model data that is a sequence or trajectory through time, for instance audio signals, text (sequences of characters/words), currency exchange rates, motion of an animal.

What a good model of sequential data include?
- Use of the past to predict the future
- A notion of state – the current state depends on the past, compresses the history
- The ability to share data across time

Convolutional networks model invariances across space – can we do something similar to model invariances across time?
- Yes - time-delay neural networks

Can we use units to act as memories?
- Yes - recurrent networks
Imagine modelling a time sequence of 3D vectors.
Imagine modelling a time sequence of 3D vectors

Can model fixed context with a feed-forward network with previous time input vectors added to the network input

MLP Lecture 9 / 13 November 2018
Recurrent Neural Networks 1: Modelling sequential data
Imagine modelling a time sequence of 3D vectors.

Can model fixed context with a feed-forward network with previous time input vectors added to the network input.

Model using 1-dimension convolutions in time - time-delay neural network (TDNN).
Imagine modelling a time sequence of 3D vectors.

Can model fixed context with a feed-forward network with previous time input vectors added to the network input.

Model using 1-dimension convolutions in time - time-delay neural network (TDNN).

Network takes into account a finite context.
TDNNs in action – Speech-to-text

Networks with state

- Feed-forward = finite context: feed-forward networks (even fancy ones like Wavenet) compute the output based on a finite input history. Sometimes the required context is known, but often it is not.

- State units: we would like a network with state across time – if an event happens, we can potentially know about that event many time steps in the future.
  - State units as memory – remember things for (potentially) an infinite time.
  - State units as information compression – compress the history (sequence observed up until now) into a state representation.

- Recurrent networks with state units
Recurrent networks

output (t) □ □ □ □ □

recurrent hidden

input (t) □ □ □ □

x1  x2  x3

t
Graphical model of a recurrent network
Graphical model of a recurrent network

Unfold a recurrent network in time
Simple recurrent network

\[ y_k(t) = \text{softmax} \left( \sum_{r=0}^{H} w_{kr}^{(2)} h_r(t) + b_k \right) \]

\[ h_j(t) = \text{sigmoid} \left( \sum_{s=0}^{d} w_{js}^{(1)} x_s(t) + \sum_{r=0}^{H} w_{jr}^{(R)} h_r(t-1) + b_j \right) \]
View an RNN for a sequence of $T$ inputs as a $T$-layer network with shared weights.
View an RNN for a sequence of $T$ inputs as a $T$-layer network with shared weights.
View an RNN for a sequence of $T$ inputs as a $T$-layer network with shared weights.

Train an RNN by doing backprop through this unfolded network.

Weight sharing:
- If two weights are constrained to be equal ($w_1 = w_2$) then they will stay equal if the weight changes are equal ($\partial E/\partial w_1 = \partial E/\partial w_2$).
- Achieve this by updating with $(\partial E/\partial w_1 + \partial E/\partial w_2)$ (cf Conv Nets).
Bidirectional RNN

- Output a prediction that depends on the whole input sequence
- Bidirectional RNN – combine an RNN moving forward in time, with one moving backwards in time
- State units provide a combined representation that depends on both the past and the future
We can train a network by unfolding and *back-propagating through time*, summing the derivatives for each weight as we go through the sequence.

More efficiently, run as a recurrent network:
- cache the unit outputs at each timestep
- cache the output errors at each timestep
- then backprop from the final timestep to zero, computing the derivatives at each step
- compute the weight updates by summing the derivatives across time

Expensive – backprop for a 1,000 item sequence equivalent to a 1,000-layer feed-forward network

Truncated BPTT – backprop through just a few time steps (e.g. 20)
Recurrent neural networks
examples from ancient history
(using “vanilla” RNNs and BPTT)
Example 1: speech recognition with recurrent networks

Example 2: recurrent network language models

T Mikolov et al (2010). “Recurrent Neural Network Based Language Model”, *Interspeech*  
Summary

- Model sequences using finite context using feed-forward networks with convolutions in time (TDNNs, Wavenet)
- Model sequences using infinite context using recurrent neural networks (RNNs)
- Unfolding an RNN gives a deep feed-forward network with shared weights
- Train using back-propagation through time
- Back-propagation through time
- (Historical) examples on speech recognition and language modelling
- Reading: Goodfellow et al, chapter 10 (sections 10.1, 10.2, 10.3) http://www.deeplearningbook.org/contents/rnn.html
- Next lecture: LSTM, sequence-sequence models