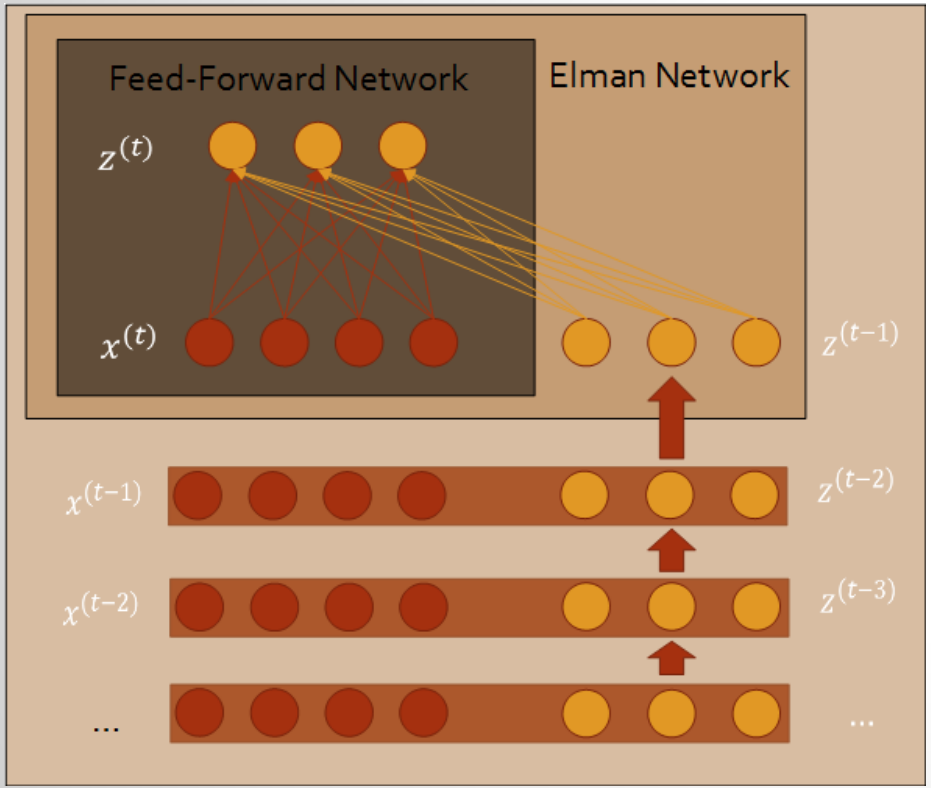
Deep Learning with Theano

Presented by Chuong Ngo

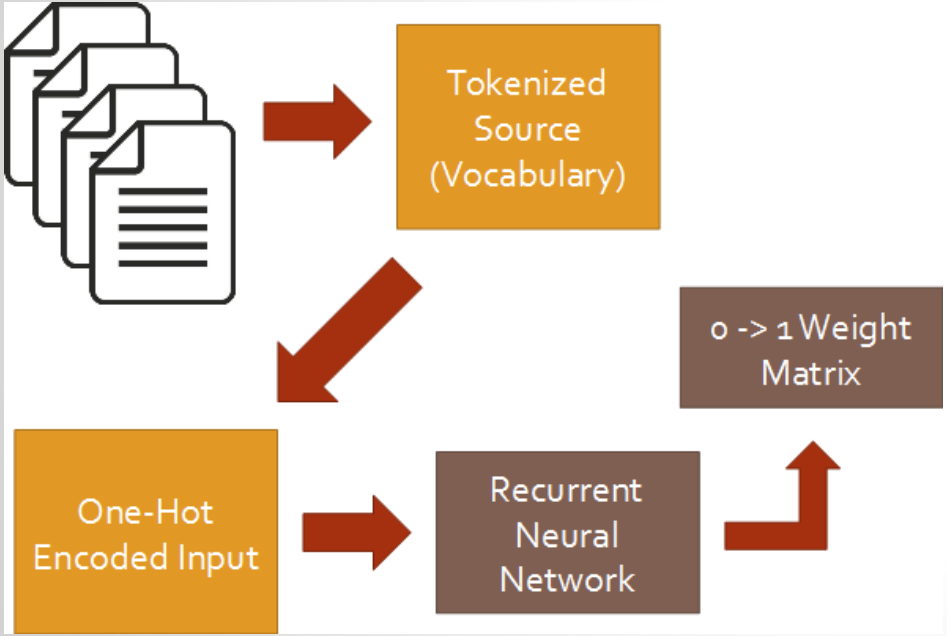
Some slides and/or images provided by Kyle Wallace
Based on the work by White et. al

Recurrent Neural Networks (RNN)



- Augmented with *Elman Networks*
  - Short-term memory
- Deep Learning Structure
  - Greater previous context
- Trained with *backpropagation through time*

RNN and Software Engineering

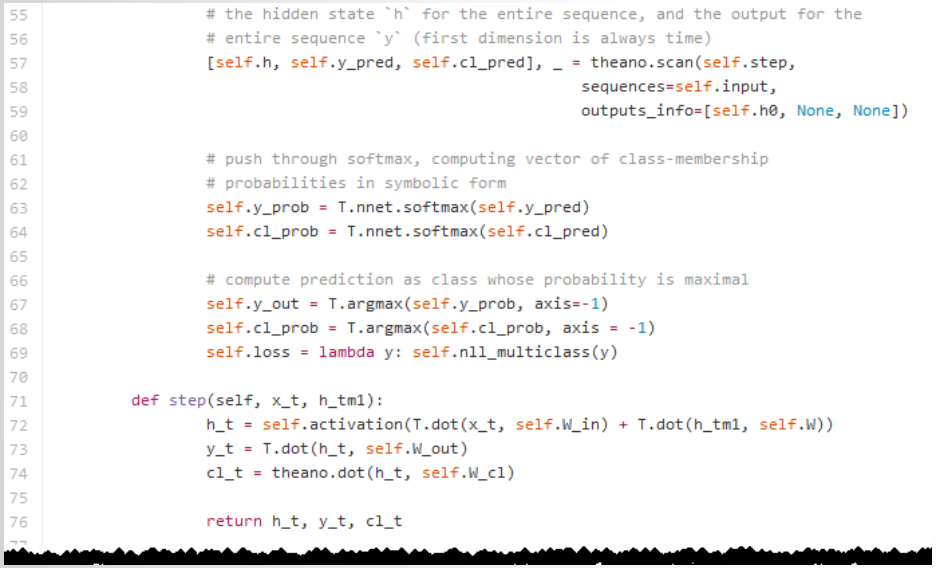


- Vocabulary set from tokenized source
  - One-hot encoding
- Stochastic gradient descent
- Softmax over classes and output
  - Decreases runtime.
- Sigmoid activation

Theano Overview

- Python library and optimizing compiler for ML
  - By University of Montreal
- Package of various technologies
- Compiled and function optimizations
- Runs on CPU/GPU

RNNs and Theano

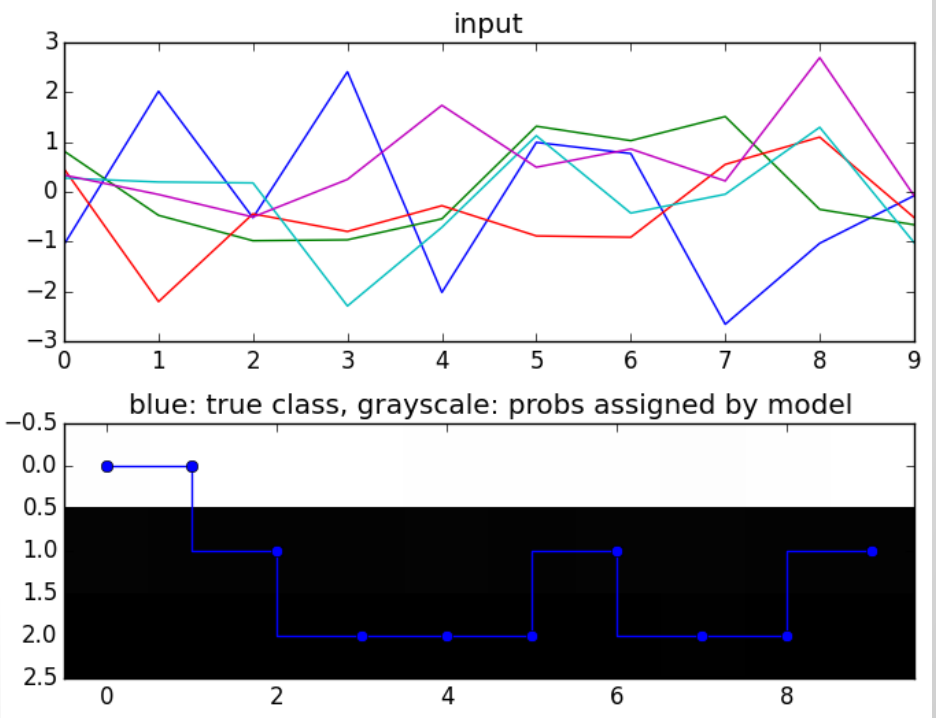


```
55                # the hidden state `h` for the entire sequence, and the output for the
56                # entire sequence `y` (first dimension is always time)
57                [self.h, self.y_pred, self.cl_pred], _ = theano.scan(self.step,
58                                                        sequences=self.input,
59                                                        outputs_info=[self.h0, None, None])
60
61                # push through softmax, computing vector of class-membership
62                # probabilities in symbolic form
63                self.y_prob = T.nnet.softmax(self.y_pred)
64                self.cl_prob = T.nnet.softmax(self.cl_pred)
65
66                # compute prediction as class whose probability is maximal
67                self.y_out = T.argmax(self.y_prob, axis=-1)
68                self.cl_prob = T.argmax(self.cl_prob, axis = -1)
69                self.loss = lambda y: self.nll_multiclass(y)
70
71        def step(self, x_t, h_tm1):
72                h_t = self.activation(T.dot(x_t, self.W_in) + T.dot(h_tm1, self.W))
73                y_t = T.dot(h_t, self.W_out)
74                cl_t = theano.dot(h_t, self.W_cl)
75
76                return h_t, y_t, cl_t
```

- Shared variables
  - Lower memory footprint
- Scan
  - Less memory used
  - Slightly faster
  - Can see previous *n* time steps
- Taps - allows for multiple inputs

Way Ahead

- Run one-hot encodes through RNN
  - Type errors
- Compare performance (time)
  - Is Theano faster?
- CPU vs GPU
  - Is the GPU worth it?



Deep Learning Structures & Software Engineering Tasks

- Where else can Deep Learning be used?
- Deep Autoencoders
  - Compressed representation of data
  - Clone detection?
  - Error prediction?
- Deep Belief Networks
  - Error prediction?
  - API preconditions?

Questions?