Recurrent Neural Networks in Theano

Philémon Brakel
Institut des algorithmes d'apprentissage de Montréal
Montreal Institute for Learning Algorithms
Université de Montréal

August 11th, Deep Learning Summer School 2015, Montréal

Recurrent Neural Networks

Neural networks that can process sequences of inputs

- Used to process speech, language, music,. . .
- Recurrent Neural Networks are very powerful:
  - Non-linear
  - Distributed representations
  - No Markov assumptions
- However:
  - Optimization can be challenging
  - Learning *long-term dependencies* is difficult
  - Computations are not as easy to *parallelize*

Standard Architecture

$$h_t = q(W_{xh}x_t + W_{hh}h_{t-1} + b_h)$$
$$y_t = r(W_{hy}h_t + b_y)$$

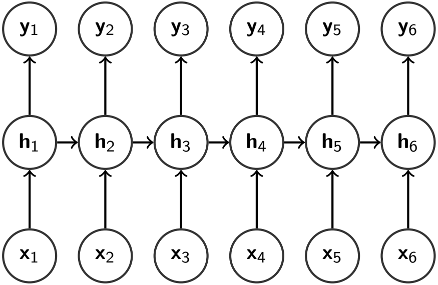


Figure : A simple Recurrent Neural Network

Long Short-Term Memory (LSTM)

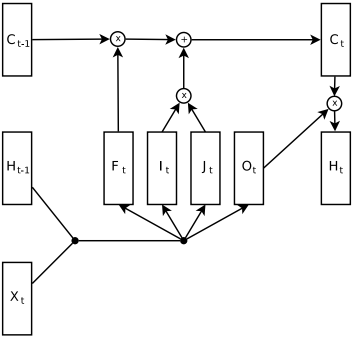


Figure : LSTM: Learn long term dependencies by asserting control over what goes in and out of *memory cells*.[2]

[2] Figure Taken from Jozefowicz et al. (2015)

Long Short-Term Memory (LSTM)

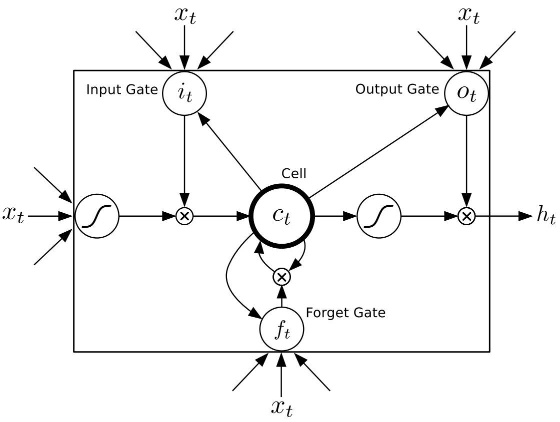


Figure : Another LSTM[3]

[3]Figure from Graves et al. (2013)

Update Equations

$$
\begin{aligned}
i_t &= \tanh(W_{xi}x_t + W_{hi}h_{t-1} + b_i)\\
j_t &= \sigma(W_{xj}x_t + W_{hj}h_{t-1} + b_j)\\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f)\\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o)\\
c_t &= f_t \otimes c_{t-1} + i_t \otimes j_t\\
h_t &= \tanh(c_t) \otimes o_t
\end{aligned}
$$

RNNs can be Stacked

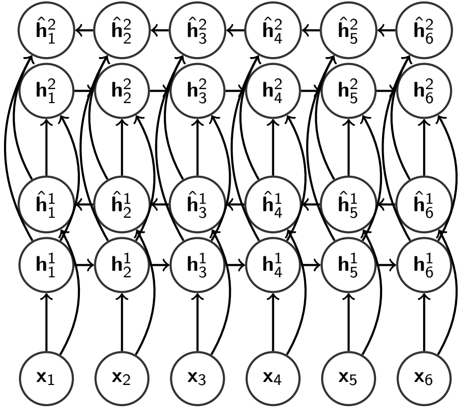
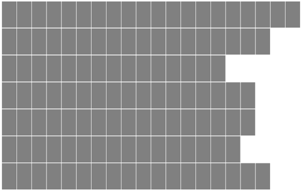


Figure : Two Bidirectional Recurrent Neural Networks stacked on top of each other.

Parallelizing RNN computations

Apply RNNs to *batches* of sequences

Present the data as a 3D tensor of $(T \times B \times F)$. Each dynamic update will now be a matrix multiplication.



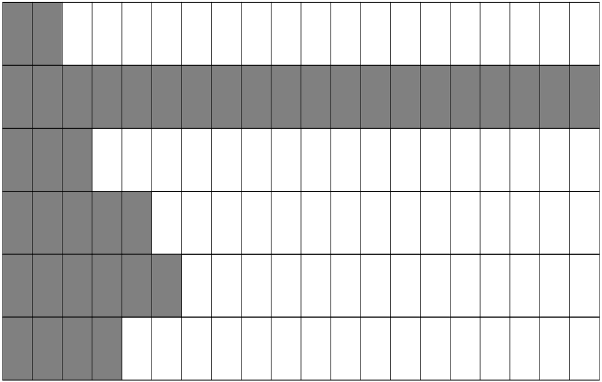
Binary Masks

A *mask* matrix may be used to aid with computations that ignore the padded zeros. In Theano this may be required to keep computations *differentiable*.

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0

Binary Masks

It may be necessary to (partially) sort your data.



Final Notes

- Fuel has a transformer that automatically padds a batch of sequences and adds a mask
- Since masks are often used for multiplication, their type should often be floating point
- Be careful that your implementation doesn't nest `scan` nodes

Final Notes

- Fuel has a transformer that automatically padds a batch of sequences and adds a mask
- Since masks are often used for multiplication, their type should often be floating point
- Be careful that your implementation doesn't nest `scan` nodes
- Have fun!