Stock Prediction Using Recurrent Neural Network

Weiguang Guan
guanw@sharcnet.ca

Code & data: http://www.sharcnet.ca/~guanw

SHARCNET™

Outline

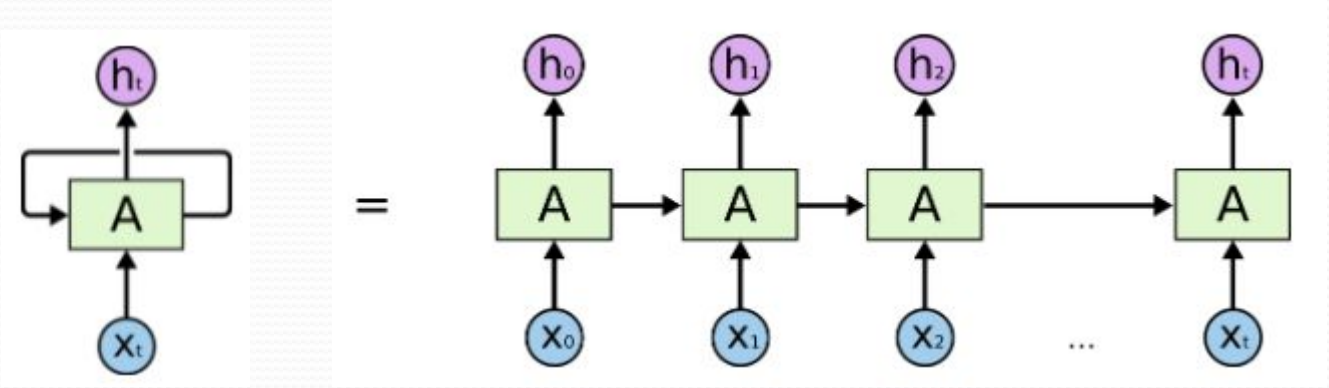
- Introduction
  - What is RNN
  - How does LSTM work
- Stock prediction with LSTM
  - Goal
  - Data
  - Architecture
  - Results
- Conclusion

Reference

- Tesorflow documentation
- "*Understanding LSTM Networks*", Christopher,Olah, http://colah.github.io/posts/2015-08-Understanding-LSTMs/ (I use a lot of diagrams from this blog in this presentation)
- "*Predict Stock Prices Using RNN*", Lilian Weng, https://lilianweng.github.io/lil-log/2017/07/08/predict-stock-prices-using-RNN-part-1.html
- "*Deep learning with long short-term memory networks for financial market*", Thomas Fischer and Christopher Krauss, https://www.iwf.rw.fau.de/files/2015/12/11-2017.pdf

SHARCNET™

What is RNN (Recurrent NN)

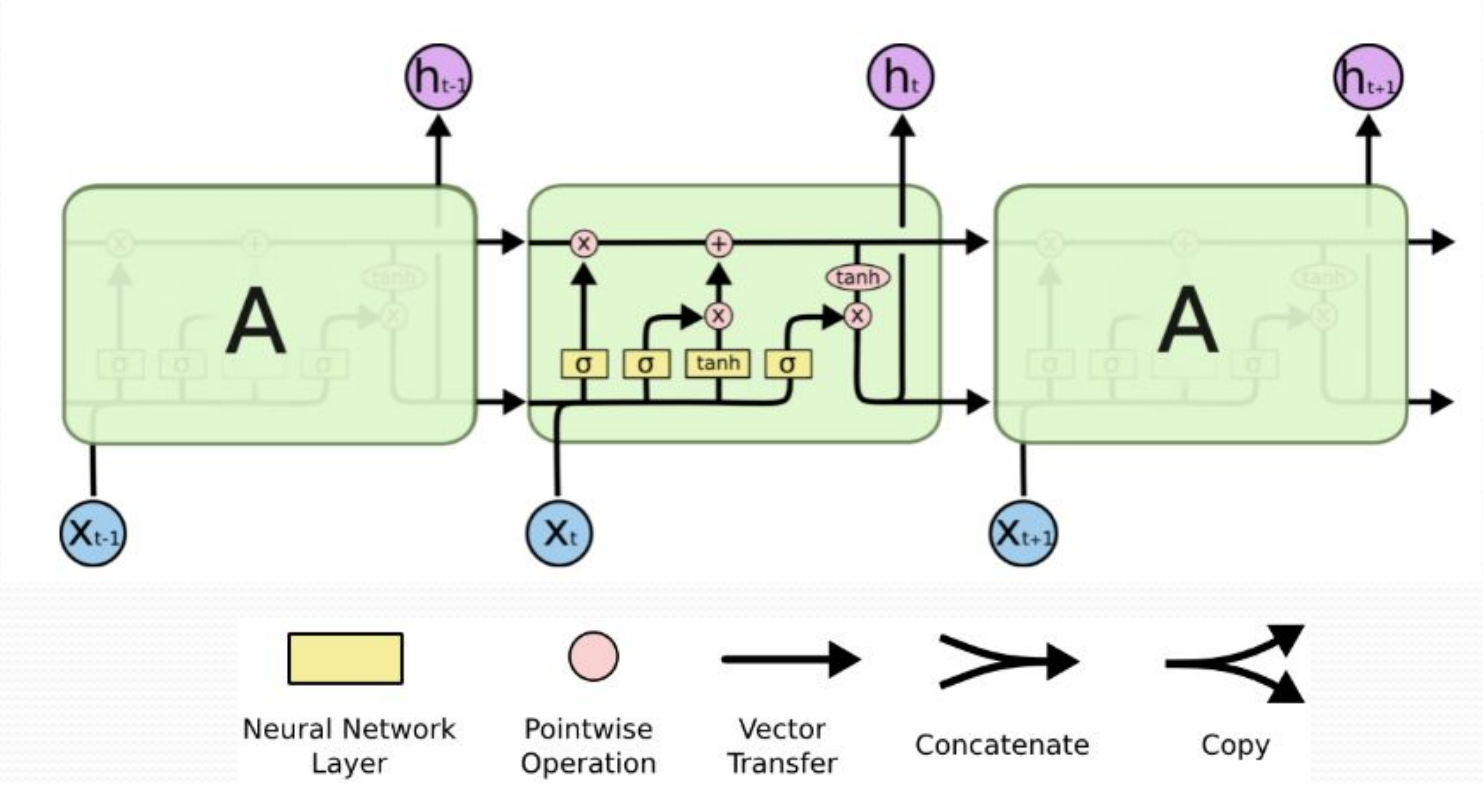


With loops

Unrolled RNN

What is LSTM

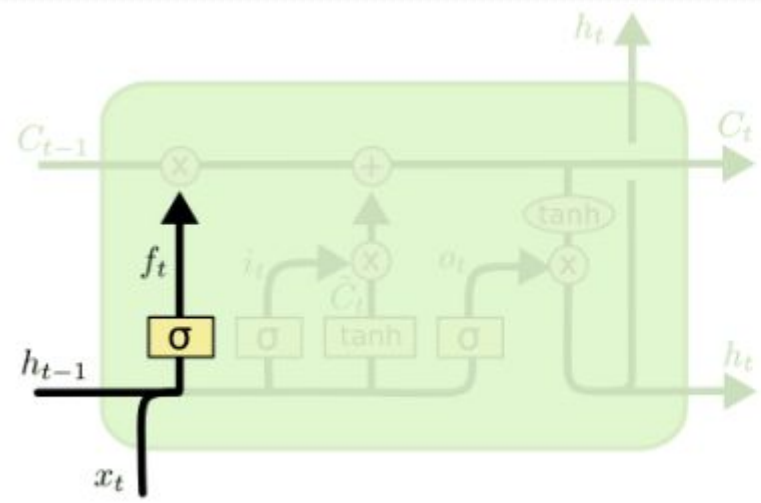
- Long Short-Term Memory is a special type of RNN
- Capable of learning long-term dependencies



SHARCNET™

Closer look at part of LSTM

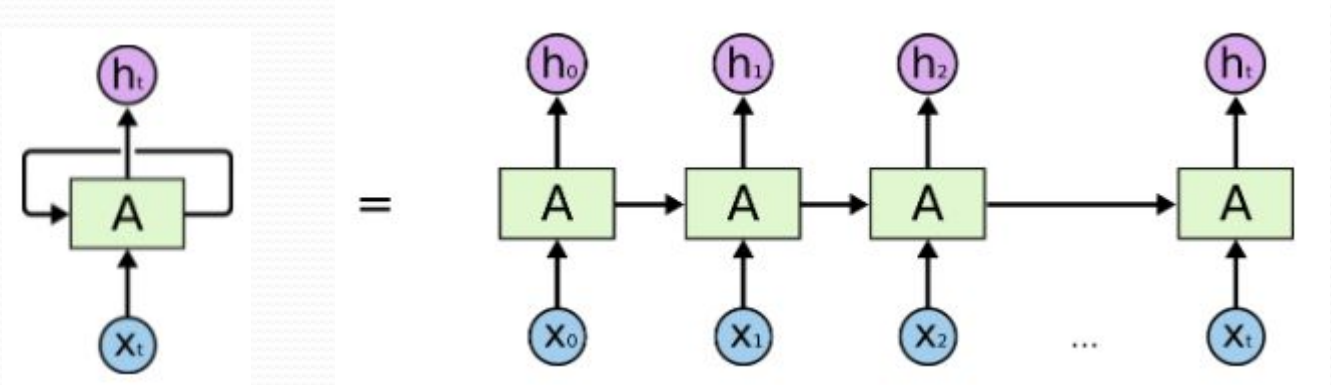
- Gate that controls how much information to keep
- Gates are defined by weights and biases that form the trainable variables



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Confusing concepts and notations

- num_units in tf.contrib.rnn.LSTMCell
- num_steps: number of repeats in the unrolled version
- input_size: length of X_t (X_t is a vector)

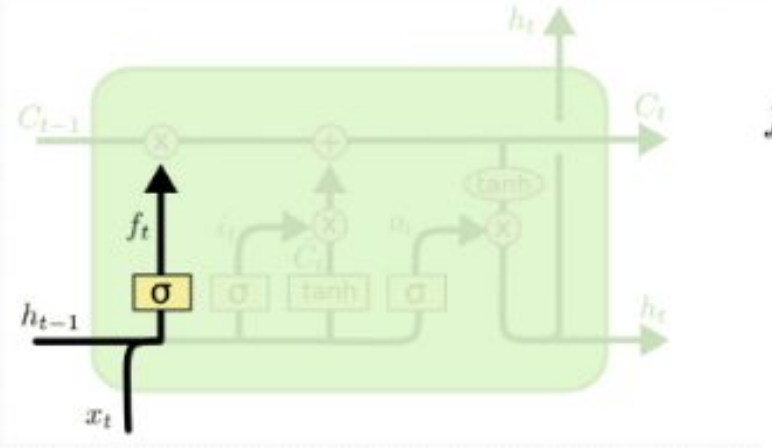


LSTM cell LSTM layer

SHARCNET™

Confusing concepts and notations

- num_units in tf.contrib.rnn.LSTMCell
  - size of lstm cell
  - output size



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Suppose
num_units = 10
input_size = 5
then
$[h_{t-1}, x_t]$ is vector of 15
W_f is matrix of 10x15
b_f is vector of 10
f_t is vector of 10

SHARCNET™

Hands-on

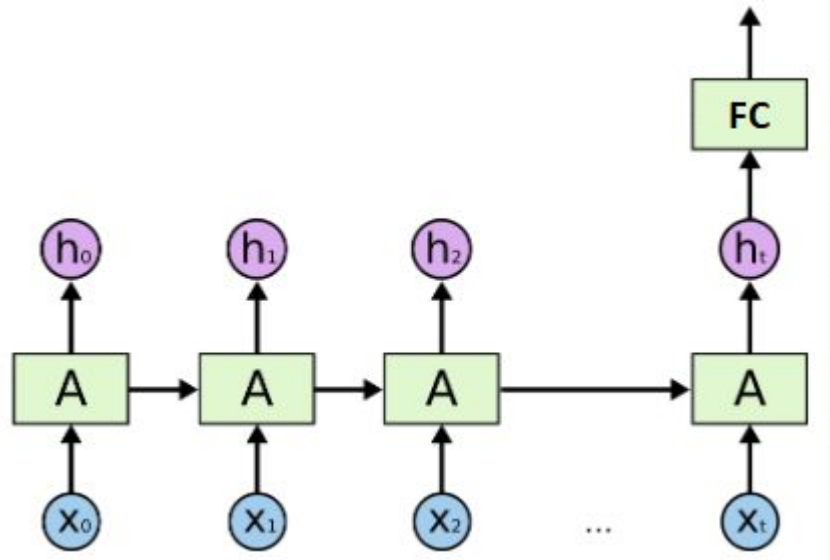
- Code
  - main.py: the main program
  - loadData.py: for data formatting
  - lstmRNN.py: for building and training LSTM RNN
- Data
  - data/SP500.csv
- Trained model
  - “model” subfolder

Some Tensorflow APIs

- Create an LSTM cell:
  tf.contrib.rnn.LSTMCell(num_units, …)
- Wrap a cell with Dropout:
  tf.contrib.rnn.DropoutWrapper(cell, output_keep_prob, ...)
- Construct an RNN:
  tf.nn.dynamic_rnn(cell, inputs, …)

Apply LSTM to stock prediction

- Goal: to demonstrate the use of LSTM, not to develop a winning model
- Data: S&P 500 index, downloaded from Yahoo
- Architecture



SHARCNET™

Data Loading/Formatting

- $[p_1, p_2, \ldots, p_n]$, $n = 17292$, closing price 1950-2018
- $[\tilde{p}_1, \tilde{p}_2, \ldots, \tilde{p}_n], \tilde{p}_i = \frac{p_i - p_{i-1}}{p_{i-1}}$
- Samples (num_steps = 100)

Data are split into two sets as training and evaluation

Conclusion

- “Random walk” theory
- What published works (by Krauss) tell us
- Want to do serious work?
  - Kaggle competition: https://www.kaggle.com/c/two-sigma-financial-news
  - Quantopian: https://www.quantopian.com/