QBUS64840 Predictive Analytics

Recurrent Neural Networks

Dr. Minh-Ngoc Tran
University of Sydney Business School

Recommended reading

- Online textbook Section 9.1 and 9.3: introduces (very briefly) some concepts in neural networks.
- *Deep Learning*, Chapter 10 by Goodfellow, Bengio and Courville, freely available at `https://www.deeplearningbook.org`

Table of contents

Learning objectives

- Know how to do forecasting with recurrent neural networks (RNN)
- How RNN can be used for financial time series data

Recurrent neural networks (RNN)

Recurrent neural network (RNN)

- There are at least two approaches to modeling time series data.
- One approach is to represent time effects *explicitly* via some simple functions, often linear functions, of the lagged values of the time series.
  - This is the mainstream time series data analysis approach in the statistics literature
  - Well-known models: AR or ARMA, etc.

Recurrent neural network (RNN)

- The alternative approach is to represent time effects *implicitly* via latent variables (also called hidden states)
  - hidden states are designed to store the memory of the dynamics in the data
  - updated in a recurrent manner using the information carried over by their values from the previous time steps and the information from the data at the current time step
- Models are called Recurrent neural networks (RNN), first developed in cognitive science and successfully used in computer science and other fields

Recurrent neural network (RNN)

- Let the time series data be $\{D_t = (x_t, y_t), t = 1, 2, ...\}$ where x_t is the vector of inputs and y_t the output.
  - E.g., y_t: sales at time t
  - $x_{t,1} = y_{t-1}$, $x_{t,2}$: ads hours at time $t-1$, $x_{t,3}$: consumption expenditure index at time $t-1$, etc.
- For ease of understanding, it might be useful to think of x_t as scalar; however, RNN is often efficiently used to model multivariate time series
- If the time series of interest has the form $\{y_t, t = 1, 2, ...\}$, it can be written as $\{(x_t, y_t), t = 2, ...\}$ with $x_t = y_{t-1}$, or $\{(x_t, y_t), t = p+1, ...\}$ with $x_t = (y_{t-1}, y_{t-2}, ..., y_{t-p})$.
- The goal is to estimate the prediction $\mathbb{E}(y_t | x_t, D_{1:t-1})$.

Recurrent neural network (RNN)

- First, let's use a feedforward neural network (FNN) to transform the raw input data x_t into a set of hidden units h_t (for the purpose of predicting y_t).
- But we need to take into account the dynamics/serial correlation of the time series data
- The main idea behind RNN is to let the set of hidden units h_t to feed itself using its value h_{t-1} from the previous time step $t-1$.
- Hence, RNN can be best thought of as a FNN that allows a connection of the hidden units to their value from the previous time step, which enables the network to possess the memory.

Recurrent neural network (RNN)

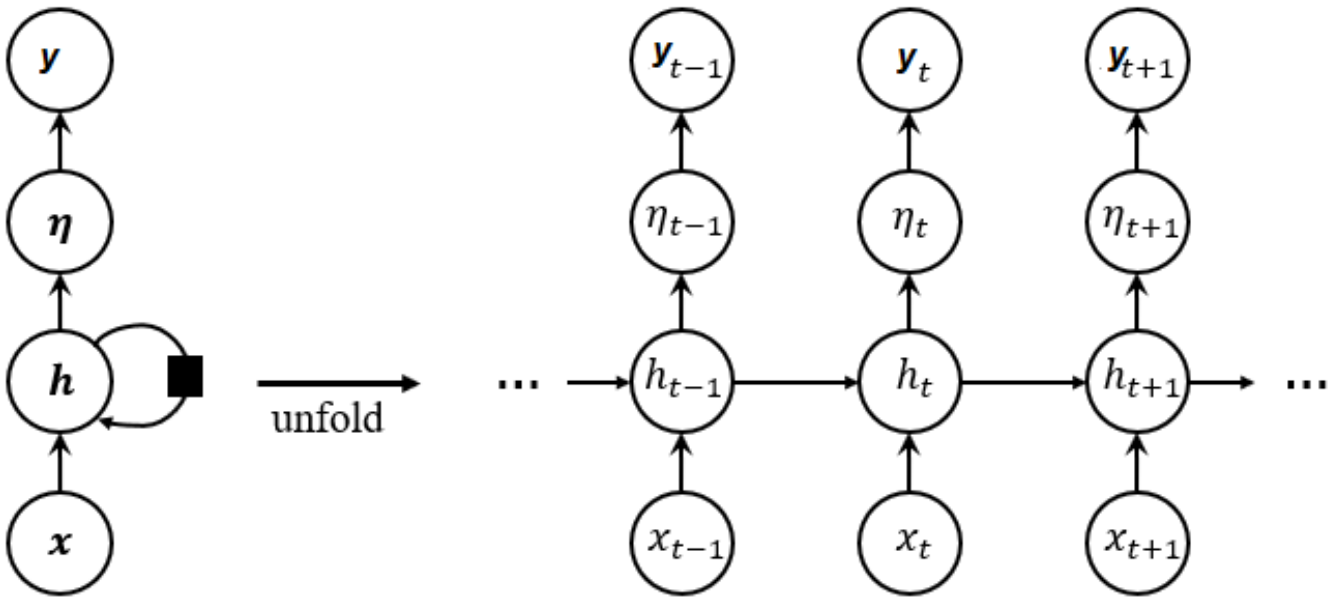
- Mathematically, this basic RNN model is written as

$$\begin{aligned} h_t &= h(vx_t + wh_{t-1} + b) \\ \eta_t &= \beta_0 + \beta_1 h_t \\ y_t &= \eta_t + \epsilon_t \end{aligned}$$

  where the ϵ_t are white noise with mean 0 and variance σ^2.
- v, w, b, β_0 and β_1 are model parameters (need to be estimated),
- $h(\cdot)$ is a non-linear activation function (e.g. tanh or sigmoid functions)
- Usually we can set $h_0 = 0$, i.e. the neural network initially doesn't have any memory.

Recurrent neural network (RNN)

Graphical representation of the basic RNN model: the black square indicates the delay of one time step.



Recurrent neural network (RNN)

Special case: If the time series of interest has the form $\{y_t, t = 1, 2, ...\}$ and we take $x_t = y_{t-1}$ as the input:

- RNN model

$$\begin{aligned} h_t &= h(vy_{t-1} + wh_{t-1} + b) \\ \eta_t &= \beta_0 + \beta_1 h_t \\ y_t &= \eta_t + \epsilon_t \end{aligned}$$

- Forecast and variance

$$\widehat{y}_{t|1:t-1} = \mathbb{E}(y_t|y_{1:t-1}) = \eta_t$$

$$\mathbb{V}(y_t|y_{1:t-1}) = \sigma^2$$

Recurrent neural network (RNN): training

- Let $\{D_t = (x_t, y_t), t = 1, 2, \ldots T\}$ be the training dataset
- Sum of squared errors

$$\text{SSE} = \sum_{t=1}^{T} (y_t - \widehat{y}_{t|1:t-1})^2$$

- The model parameters $\theta = (v, w, b, \beta_0, \beta_1)$ are estimated by minimizing the SSE
- Training a RNN is similar to training a Feedforward Neural Network.

Advanced variants of RNN

- The RNN considered so far is a basic RNN: simple and might be not flexible enough. Training a basic RNN is often challenging: its gradient is either vanishing or exploding
- Many variants of RNN are proposed to overcome the issues above: Long Short-Term Memory (LSTM) is one of the most widely used RNNs, applied to large-scale industry-level applications: language processing, video data processing, etc.
- We won't discuss LSTM in detail here, but show how to use it in the tutorial.

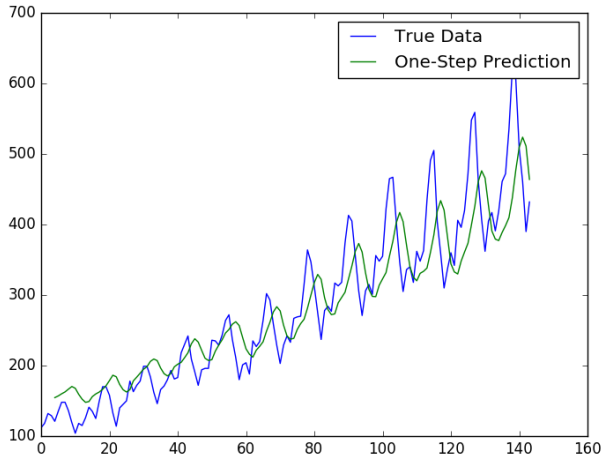
An example

- Dataset: International Airport Arriving Passengers: in csv format
- Prepare Data: `Xtrain, Ytrain, Xtest` and `Ytest` as we have done for traditional neural networks
- Define the RNN architecture:
  - We use one layer of LSTM.
  - Choose the size of time-window $p = 4$: This should be determined by exploring ACF or PACF
  - Choose the dimension of hidden states $d = 10$
  - ```
    model = Sequential()
    model.add(LSTM(input_dim=4, output_dim=10,
    return_sequences=False))
    model.add(Dropout(0.2))
    model.add(Dense(output_dim=1)
    model.add(Activation("linear"))
    ```
    ```

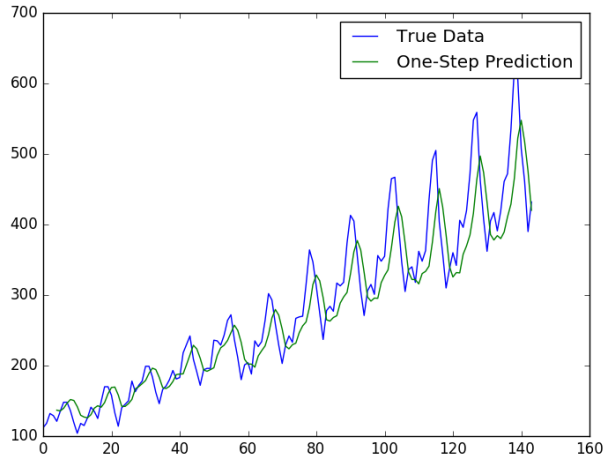
An example: Training

- One statement to train the model
  `model.fit(Xtrain, Ytrain, batch_size=100, nb_epoch=10, validation_split=0.05)`
  - `batch_size` defines the batch size in stochastic optimisation process. You may use default value
  - `nb_epoch` defines the number of epoches of optimisation. Start with a smaller value for testing
  - `validation_split=0.05` means in the training process, about 5% data from `Xtrain` will be used for validation
- Please read the program `Lecture12_Example01.py`

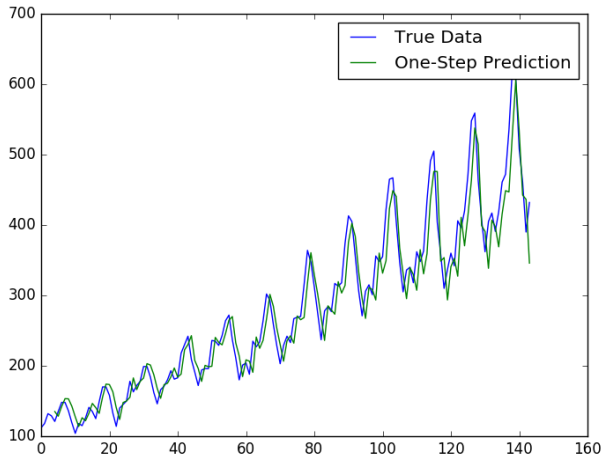
An example: Forecasting



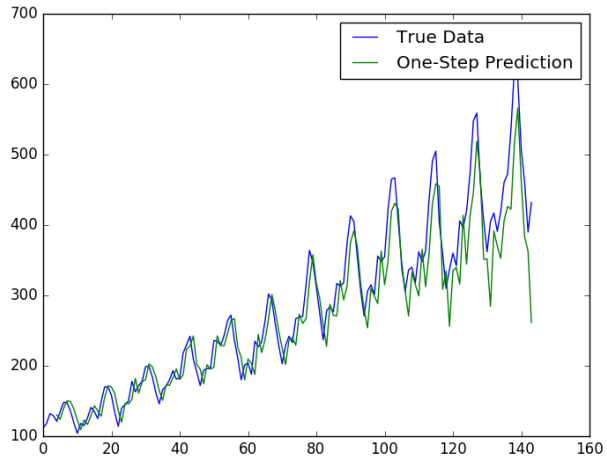
Epoch = 1



Epoch = 20



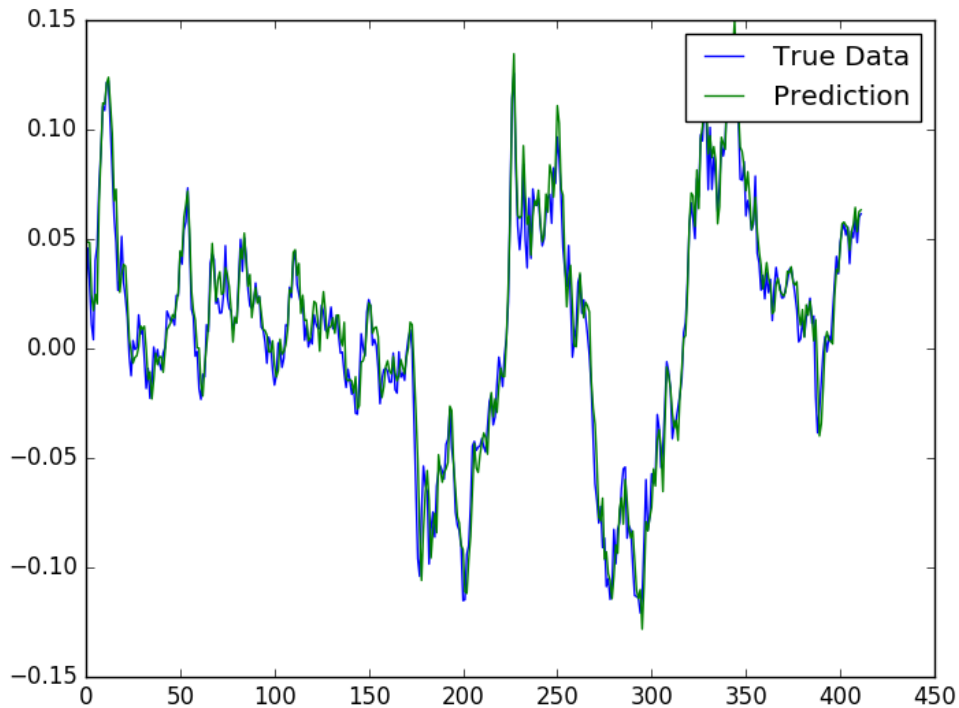
Epoch = 100



Epoch = 200

An example: Share Price

- Please read the program `Lecture12_Example02.py` where we use two layers of LSTMs



Final exam

- Everything covered in the lectures and tutorials, including Python, can be tested, except the technical slides with "*"
- This is an open book exam with some limitations (as in the midterm exam).
- The exam is proctored using ProctorU - please read carefully the Canvas announcements related to the final exam.
- Exam sample for practice will be available.

Final exam: some advice

- Start your revision as soon as possible (is it too late? :-)
- Study carefully the final exam practice questions.
- In the exam, read through the whole exam and carefully read the instructions.
- Organise your time effectively, i.e. allocate time for each question.
- Always answer ALL the questions. Even if you think you know nothing about a topic, you might get A FEW marks by providing a partially complete answer OR making some sensible comments. Remember, any unanswered question scores zero!
- Try to leave some time to review your answers at the end.

Consultation hours

- I still run consultation as usual until the exam day. Or send me an email to make appointment at other times.
- Check with your tutor/lecturer as they might extend their consultation hours as well.

Study hard and play hard!
All the best with your exams!

Hope you'll be like this after the final exams...

Don't forget to give the teaching team your feedback on the course! If you can, please do it now.

You might win a Macbook!

USS PRIZES

To thank you for participating, each unit of study survey (USS) completed between 1 January 2017 and 30 June 2017 will give you an entry into the Semester 1 USS prize draw, to be drawn on Thursday 31 August 2017.

1st prize - 13-inch MacBook Air

2nd prize - iPad mini 4 Wi-Fi 64GB

3rd prize - 38mm Apple Watch Sport

To be entered in the draw, simply complete a unit of study survey (USS) for one or more of your units of study.

The more surveys you complete, the more chances you have to win!

If you have any questions, please email qa.surveys@sydney.edu.au.

WINNERS OF THE UNIT OF STUDY SURVEY PRIZE DRAW (SEMESTER TWO) IN 2016 were:		
1st Prize	Kevin Barker	13-inch MacBook Air
2nd Prize	Soe Yan Kyi Ling	iPad mini 4 Wi-Fi 64GB
3rd Prize	Yufan Zheng	Apple Watch Sport