Workshop 8

Learning From Big Data

Recurrent Neural Networks (RNNs)

*Please do the exercises or tasks without "Optional" mark at first. After that, if you still have some time, please try the tasks with "Optional".

As lectured, **RNNs** can predict the future, for examples: analyze time series data of stock prices and help you to make decision of when to buy or sell, based on the predicted trend; in autonomous driving systems, they can anticipate car trajectories and help avoid accidents.

Review the lecture-slides, understand

- An architecture of RNNs in terms of time *t*
- The types of RNNs
- The challenges or issues on training a RNN, typically, **"Vanishing Gradient Problem"**
- Long Short-term Memory Unit (**LSTM**)

Then, you do the following exercises.

Part 1: Building RNNs with Keras

In this part, we're going to try to predict the upward and downward trends that exist in the Google stock price, by building a Recurrent Neural Network (RNN) with Keras.

Indeed, there is a Brownian motion that states that the future evaluations of the stock price are independent from the past. So it's actually impossible to predict exactly the future stock price. Otherwise, we would all become billionaires. But it's actually possible to predict some trends.

We're going to train our data, given in …/Lab8/RNNs_with-Keras, on five years of the Google stock price, from the beginning of 2012 to the end of 2016. And then, we will try to predict the stock price of the first month of 2017.

Actions:

- Open a notebook, RNN.ipynb, given in …/Lab8/RNN-with_Keras, where we learn how to build a recurrent neural network, train the data, and predict the future, based on the time-series data.
- Study the codes in each cell of the notebook, including the comments, while you run them.
- Try to understand the codes and answer the questions I give in the comments.

Part 2: Implementing RNNs with TensorFlow

In this Part, we do a number of exercises for building RNNs by using TenorFlow.

Task 1: Basic RNNs

1. There are different ways to implement RNNs model. In the lecture, four approaches were introduced. Based these discussions, study the codes together with the text I added with the following thetitles:

    - Manual RNN
    - Using static_run()
    - Packing Sequence
    - Using Dynamic_run()
    - Setting the sequence length

2. Have a look at the code of "**Training a sequence classifier**", and see how to **train a RNN model.**

Task 2: Multi-layer RNNs and Time Series

Review the lecture slides of Section 3 "**Implementing RNNs in Tensoflow**", where two cases were discussed for training RNNs, (i) Training a sequence classifier; (ii) Training to predict time series. Then, study the codes in the note book.

1. In case 1, its code is given under the title "**Multi-layer RNN**". See how to build a multi-layer RNN by using TensorFlow API, such as BasicRNNCell, and MNIST data train the network.

2. In case 2, we have the following samples for training them to predict time series in the notebook, and they are:

    - Using an OutpuProjectionWrapper
    - Without Using an OutpuProjectionWrapper
    - Generating a creative new sequence

    Study these codes and understand how to achieve the predictions of the time series.

Task 3: (Optional) Deep RNNs

- I have edited a PDF file, Deep_RNNs.pdf, as your reference to your further study on this topic, give in …/Lectures/Session_9. Also, the codes, discussed within the text, are present in the last part of the notebook, Recurrent_Neural_Networks.ipynb, which may be used for your understanding and practice with Depp RNNs.