Documentation on Solar Forecasting Project

Haoming Shen

04/02/2018

Contents

1 Overview

This project aims at improving intra-day Ground Horizontal Irradiance (GHI) forecasting using machine learning based algorithms. Predicting the amount solar energy in the next few hours/days is of great importance to power system operations and control. Currently two methods are implemented and documented below. If you have any questions with respect to this project, please contact me through email (my unique name is 'hmshen' I should able to help during this summer.

NOTICE: If you need to use the code below (even part of it), I would recommend you read it critically before using it.

1.1 Github repo for this project

`https://github.com/hm-shen/weather_forecast/tree/master/src`

1.2 Dependencies

Tensorflow, Sklearn, Pandas, Numpy, Scipy, ...

I would recommend you use Anaconda Python.

2 SVM Based Solar Irradiance Forecasting

This part is based on Hourly Solar Irradiance Prediction Based on Support Vector Machine and Its Error Analysis

2.1 Getting Started

There are three modes: `weather_prediction`, `houldout_trainint`, `grid_search`. `weather_prediction` will load the input data and complete solar irradiance prediction; `holdout_training` will split the input data into holdout training and run test on the trained model; `grid_search` will search in the given set of parameters and find the best one.

To run this project, you can directly `python main.py` after manually setting the running mode parameters in `main.py` file.

2.2 Implementation

Given a set of training data, K-means algorithm is used to separate data into three cluster where each of them implies a specific weather type (cloudy, partly cloudy, sunny). Then, code will train a Support Vector Regression model for each type of weather.

3 LSTM Based Cloud Fraction Forecasting

Since both hourly GHI and hourly cloud fraction can be considered as time series, solving it using Long short-term memory (LSTM) becomes natural. This part implements a simple LSTM model for supervised cloud cover and GHI forecasting.

3.1 Getting Started

To perform cloud fraction prediction on NREL data sets, please run the following command from folder `cloud_forecasting/src/`:

```
python -p /path/to/NREL_data/ \
       -f 'average or variance' \
       -o /path/to/output/ \
       -c /path/to/configuration file/ \
       -n 'name of the dataset' \
```

Several example bash scripts are included in `/src` folder.

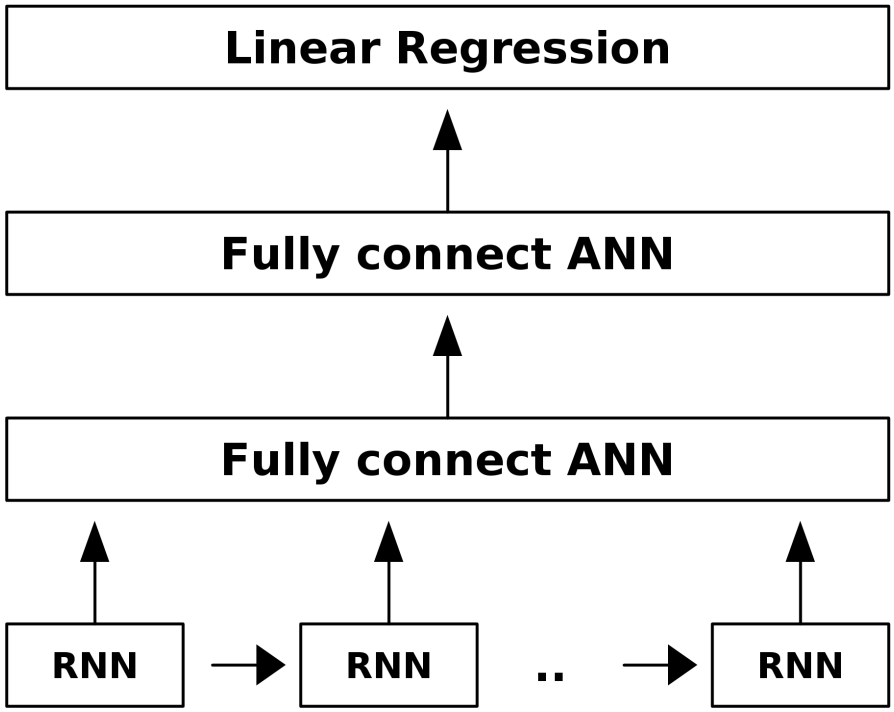
3.2 Implementation

Note that this implementation of LSTM based time series is based on `https://github.com/tgjeon/TensorFlow-Tutorials-for-Time-Series`

3.2.1 Overall Framework of LSTM

Tutorial on LSTM model can be found here (click me!)

All diagram of the architecture is shown below



where the LSTM cell is represented as "unrolled" RNN cells. So for each input x_t, y_t will be generated by LSTM cell and feed into two layers of fully connected artificial neural networks (ANN). Then the output is used as the input to a linear regressor.

3.2.2 Parameters

Parameters for LSTM are listed below:

Table 1: Parameters for setting up LSTM

Parametter	Description
time steps	how many time steps is used to predict (features)
rnn layers	the configuration of rnn layers using a list of dict
dense layers	number of units in each dense layer

3.3 Some details on data preprocessing

Those NREL data contained in the `data/` folder is a little bit messy in the sense that there may be invalid cloud fraction data in each day (e.g. `nan`, `-1`). Thus, to remove days with too many messy data, there are two variables, `ubd_min`, `lbd_max`, responsible for removing all invalid days: all days where the first valid data appearing later than `ubd_min` is removed; similarly, all days where the last valid data appearing before `lbd_max` is removed. This way, we select days with number of valid data at least (`lbd_max` - `ubd_min`). Also note that these two variables are related to the dataset you are using and thus should be set by hand in the source code `/src/driver.py`.