**National Chung Cheng University**

**Introduction to Deep Learning**

**Mid-Term Programming Exam (Take-Home)**

**2021/4/23 ~ 2021/5/7 (deadline: 23:59:59) 50% of total grade**

You are provided a time-series forecasting problem centered around restaurant visitors. The data comes from two separate sites:

* Hot Pepper Gourmet (hpg): similar to Yelp, here users can search restaurants and also make a reservation online
* AirREGI / Restaurant Board (air): similar to Square, a reservation control and cash register system

You must use the reservations, visits, and other information from these sites to forecast future restaurant visitor totals on a given date. The dataset covers the dates from 2016 until April 2017. The dataset should be divided into training data (full year of 2016) and testing data (whatever is given for 2017).

There are days in the test set where the restaurant were closed and had no visitors. These are ignored in scoring. The training set omits days where the restaurants were closed.

File Descriptions

This is a relational dataset from two systems. Each file is prefaced with the source (either air\_ or hpg\_) to indicate its origin. Each restaurant has a unique air\_store\_id and hpg\_store\_id. Note that not all restaurants are covered by both systems, and that you have been provided data beyond the restaurants for which you must forecast. Latitudes and Longitudes are not exact to discourage de-identification of restaurants.

air\_reserve.csv

This file contains reservations made in the air system. Note that the reserve\_datetime indicates the time when the reservation was created, whereas the visit\_datetime is the time in the future where the visit will occur.

* air\_store\_id - the restaurant's id in the air system
* visit\_datetime - the time of the reservation
* reserve\_datetime - the time the reservation was made
* reserve\_visitors - the number of visitors for that reservation

hpg\_reserve.csv

This file contains reservations made in the hpg system.

* hpg\_store\_id - the restaurant's id in the hpg system
* visit\_datetime - the time of the reservation
* reserve\_datetime - the time the reservation was made
* reserve\_visitors - the number of visitors for that reservation

air\_store\_info.csv

This file contains information about select air restaurants. Column names and contents are self-explanatory.

* air\_store\_id
* air\_genre\_name
* air\_area\_name
* latitude
* longitude

Note: latitude and longitude are the latitude and longitude of the *area* to which the store belongs

hpg\_store\_info.csv

This file contains information about select hpg restaurants. Column names and contents are self-explanatory.

* hpg\_store\_id
* hpg\_genre\_name
* hpg\_area\_name
* latitude
* longitude

Note: latitude and longitude are the latitude and longitude of the *area* to which the store belongs

store\_id\_relation.csv

This file allows you to join select restaurants that have both the air and hpg system.

* hpg\_store\_id
* air\_store\_id

air\_visit\_data.csv

This file contains historical visit data for the air restaurants.

* air\_store\_id
* visit\_date - the date
* visitors - the number of visitors to the restaurant on the date

Submission:

You need to submit the code of your model and fill in the following information:

**Model Information:**

* Number of layers: 5
* Number of units in each layer: 512 -> 256 -> 128 -> 32 -> 1
* Activation functions used: LeakyReLU
* Loss function: RMSLE
* Cost function: None

**Training Epochs**: 15

**Training Accuracy**: 0.5135

**Testing Accuracy**: 0.5116

(with StandardScaler, LeakyReLU and non Dropout-Layer)

**Optimization techniques employed**:

1. StandardScaler or non-normalization
2. Dropout
3. ReLU or LeakyReLU

Difference in accuracies after each optimization technique that you applied:

1. Optimization technique name: StandardScaler or non-normalization

Before optimization: Training/Testing Accuracies = 1.9776 / 1.9706

After optimization: Training/Testing Accuracies = 0.5370 / 0.5154

Any other changes: 只正規化數值(int or float)的部分，字串型態不做學習，且已使用dropout及ReLU

1. Optimization technique name: Dropout used in model or not

Before optimization(use Dropout(0.2)):

Training/Testing Accuracies = 0.5370 / 0.5154

After optimization(non-use Dropout):

Training/Testing Accuracies = 0.5145 / 0.5142

Any other changes: 已做正規化且使用ReLU

1. Optimization technique name: ReLU or LeakyReLU

Before optimization(ReLU):

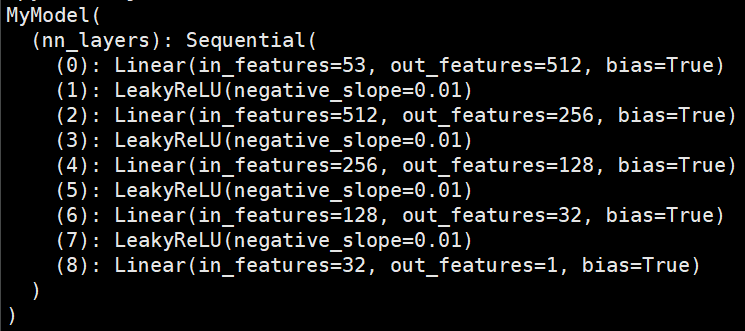
Training/Testing Accuracies = 0.5132 / 0.5138

After optimization(LeakyReLU):

Training/Testing Accuracies = 0.5135 / 0.5116

Any other changes: 已正規化且不使用Dropout

Anything special about your model:



在經過測試後，發現必須正規化訓練集，不然根本無法訓練模型。我建立的模型為五層的全連接層，分別為53 -> 512、512 -> 256、256 -> 128、128 -> 32、32 -> 1。其中雖然不合常理，但在這個模型及訓練集的環境下，不使用Dropout層，預測結果會有較低的RMSLE，故最終沒有使用Dropout層。且經過測試，LeakyReLU有較佳的訓練表現，故最終選用LeakyReLU而不是 ReLU。

Comments on the course:

雖然這堂課是使用Keras，且網路上的範例也多為使用Keras，但因為我對於Pytorch比較熟悉，所以只參考網路上對於資料集處理的作法，模型的建構完全為自己利用Pytorch實作。

透過本次作業，得以了解深度模型的實作細節，且從頭到尾完成模型訓練，並且也以此熟悉模型訓練的相關工具。