Panama Electricity Load Forecasting

CAINE, Wilbert
20584260
wcaine@connect.ust.hk

I. Dataset and Preprocessing

1. Dataset

The dataset of this project originated from Kaggle with title Panama Electricity Load Forecasting. Originally, the datasets provided includes the following csv files with the corresponding columns or features.

a. train.csv
   i. 'datetime': date and time for each row
   ii. 'nat_demand': electric load which is also the target output in this project
   iii. 12 columns or numerically continuous features: weather parameters describing the temperature, humidity, wind, and precipitation
   iv. 2 binary columns: boolean of whether the day is special days, namely holidays and school days
   v. 1 integer-values column: holiday id with 22 unique values
b. Test_Jan.csv: same structure as train.csv, except that the target output or 'nat_demand' is filled with zeros
c. Predict_Jan.csv: ground truth for Test_Jan.csv
   i. 'datetime': date and time for each row
   ii. 'nat_demand': electric load which is also the target output in this project

All of the dataset provided has no null values and the gap within each datetime is one hour, i.e., there are 24 rows of data per day.

2. Preprocessing

Although there are 3 separate datasets provided, Test_Jan.csv and Predict_Jan.csv consist of only one-month period of electricity load. To conduct a robust experiment, the project use train.csv which covers almost a period of five years. The 'datetime' column in train.csv is merely a unique id for each row, however, the day of the week and hour are extracted as features for training. The reason is simply because they may contain valuable information. Two of the useful signals include the regular working day of the week and hours at Panama when the electricity load might be higher for higher consumption by the citizen. The 'datetime' column is then discarded. Thus, there are 17 features while the ground truth is the electricity load. Finally, the clean dataset is divided into 3 subsets, namely training, validation, and testing dataset which corresponds to the first 70%, next 15%, and the rest 15% respectively.

II. Machine Learning Tasks

The problem to solve follows the task proposed by the Kaggle page. Recently, energy crisis is occurring in many countries all over the world. While people are finding other energy resource and minimizing energy consumption in daily life, there are other ways to preserve energy. Ideally, one of the possible ways to reduce energy loss is to deliver an electricity load of roughly the equivalence the electric consumption. Therefore, forecasting electricity load has been a focus nowadays, and people often turn to machine learning methodologies to solve this issue. In the end, we will also report the generalization ability in which the models are used to predict the electricity load of the next 24 hours given the prior information of a few hours. This report will show an attempt of predicting the electricity load using common machine learning frameworks.

III. Implementation Details

1. Computing Environment

The project is entirely conducted using the available services provided by Google, such as Drive and Colab without GPU usage (although most part of the code supports running on GPU).

2. Machine Learning Methods

The models are implemented using PyTorch library with default version in Colab. The machine learning methods mainly consist of Fully Connected Layers and LSTM. The reasoning behind the proposed models are based on the assumption that the prior information may be useful to predict the future demand of electricity.

3. Parameter Settings

Each model is trained with the same number of epochs of 100 and batch size of 32. The main training and validation process are adopted from the Tutorial with some modification. L1-norm between the ground truth and prediction is used as the objective function.

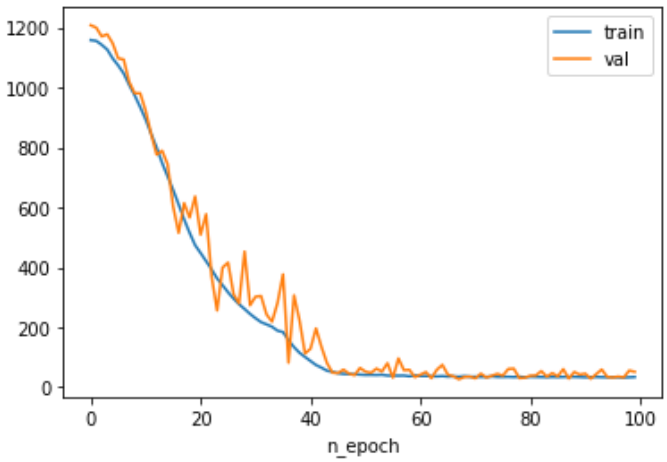
IV. Experiments

The training process is conducted by repeated randomly sampling a timeframe or a series of timeframes depending on the models from the dataset, while the validation and testing steps loop through all the corresponding dataset. To find an appropriate model, this experiment includes mainly two different models, such as Fully Connected Layer, defined as FC-Net, and Recurrent Network, named as R-Net in this report and code. The plots shown in this part is the L1-Norm difference of the ground truth and prediction.

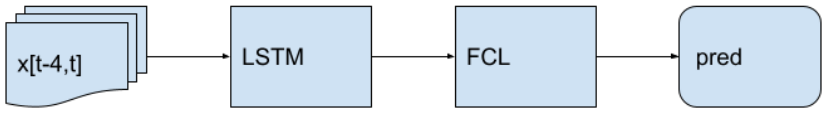
1. FC-Net

First of all, we define a feature set as 17 current feature and one prior information. The prior information is simply the electricity demand at the previous timestamp. More specifically, if the timestamp of the current feature is at time t, then the prior electricity load is at t-1. Therefore, the input is a one-dimensional vector of length 18. The FC-Net is constructed by a linear layer, two sets of linear layers followed by batch normalization, and a linear layer. ReLu is employed as the activation function.

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Linear-1                [-1, 32]             608
              ReLU-2                [-1, 32]               0
            Linear-3                [-1, 32]           1,056
       BatchNorm1d-4                [-1, 32]              64
              ReLU-5                [-1, 32]               0
            Linear-6                 [-1, 8]             264
       BatchNorm1d-7                 [-1, 8]              16
              ReLU-8                 [-1, 8]               0
            Linear-9                 [-1, 1]               9
             ReLU-10                 [-1, 1]               0
================================================================
Total params: 2,017
Trainable params: 2,017
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.00
Forward/backward pass size (MB): 0.00
Params size (MB): 0.01
Estimated Total Size (MB): 0.01
----------------------------------------------------------------
```

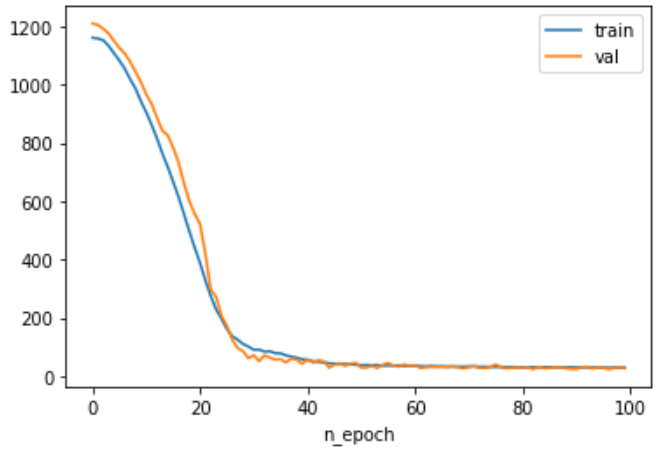


2. R-Net



R-Net starts with a batch normalization layer and LSTM layer followed by fully connected layers. The LSTM model gives a final output with size 32, while the remaining layer of FCL simply followed the structure of FC-Net with minor changes to match the dimension of the final output of LSTM. R-Net receives input of a series of feature sets with length of a pre-specified window frame, which is 5 by default. FCL will receive input of from the last hidden layer of LSTM.

```
R_Net(
  (bn1): BatchNorm1d(18, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (lstm): LSTM(18, 32, batch_first=True)
  (fc1): Linear(in_features=32, out_features=32, bias=True)
  (fc2): Linear(in_features=32, out_features=32, bias=True)
  (bn2): BatchNorm1d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (fc3): Linear(in_features=32, out_features=8, bias=True)
  (bn3): BatchNorm1d(8, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (fc4): Linear(in_features=8, out_features=1, bias=True)
  (relu): ReLU(inplace=True)
)
```

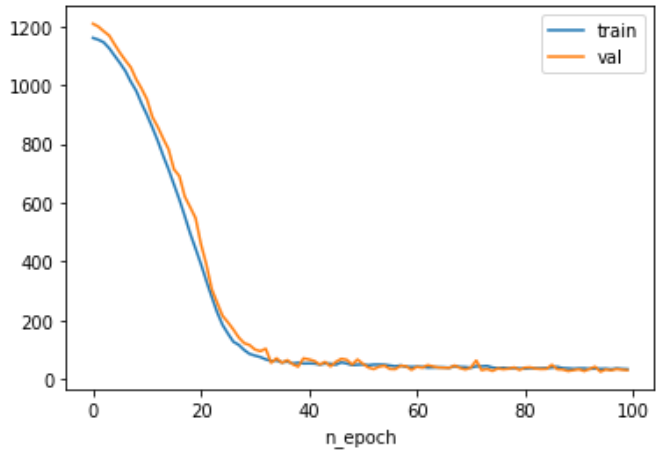


3. R2-Net

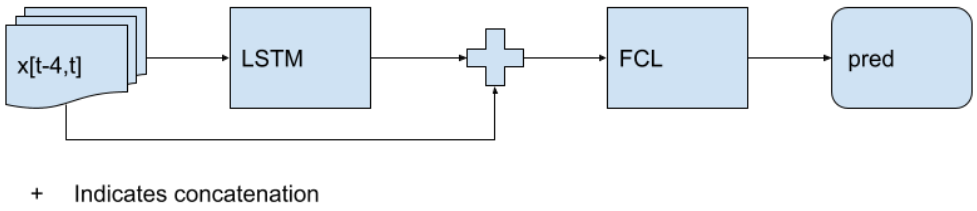
R2-Net has the same structure as R-Net, however, the LSTM layer has been increased to 2 layer LSTM.

```
R_Net(
  (bn1): BatchNorm1d(18, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (lstm): LSTM(18, 32, num_layers=2, batch_first=True, dropout=0.2)
```

```
  (fc1): Linear(in_features=32, out_features=32, bias=True)
  (fc2): Linear(in_features=32, out_features=32, bias=True)
  (bn2): BatchNorm1d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (fc3): Linear(in_features=32, out_features=8, bias=True)
  (bn3): BatchNorm1d(8, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (fc4): Linear(in_features=8, out_features=1, bias=True)
  (relu): ReLU(inplace=True)
)
```



4. RS-Net

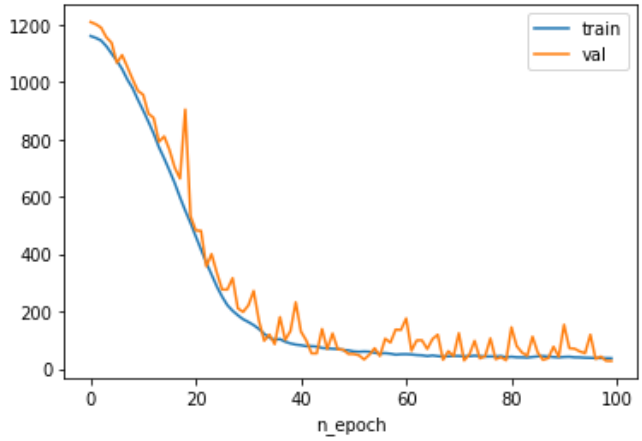


\+ Indicates concatenation

RS-Net is defined to be R-Net with a skip connection similar to those in residual unit. The skip connection concatenates the newest feature set, i.e., at timestamp t-1, and the output of the LSTM layer. After concatenation, the features are forwarded to the FCL layer which is also modified to receive the correct input. dimension.

```
R_Net(
  (bn1): BatchNorm1d(18, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (lstm): LSTM(18, 32, batch_first=True)
  (fc1): Linear(in_features=50, out_features=32, bias=True)
  (fc2): Linear(in_features=32, out_features=32, bias=True)
  (bn2): BatchNorm1d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (fc3): Linear(in_features=32, out_features=8, bias=True)
  (bn3): BatchNorm1d(8, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (fc4): Linear(in_features=8, out_features=1, bias=True)
```

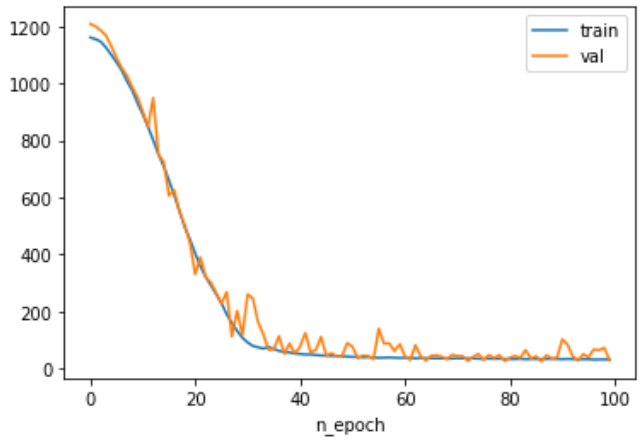
```
  (relu): ReLU(inplace=True)
)
```



5. RS2-Net

RS2-Net has the same structure as RS-Net, however, the LSTM layer has been increased to 2 layer LSTM.

```
R_Net(
  (bn1): BatchNorm1d(18, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (lstm): LSTM(18, 32, num_layers=2, batch_first=True, dropout=0.2)
  (fc1): Linear(in_features=50, out_features=32, bias=True)
  (fc2): Linear(in_features=32, out_features=32, bias=True)
  (bn2): BatchNorm1d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (fc3): Linear(in_features=32, out_features=8, bias=True)
  (bn3): BatchNorm1d(8, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (fc4): Linear(in_features=8, out_features=1, bias=True)
  (relu): ReLU(inplace=True)
)
```



V. Results and Discussion

1. Quantitative Evaluation

We compute the error as L1-Norm or MAE for each of the model above and compare the performance with the baseline model*.

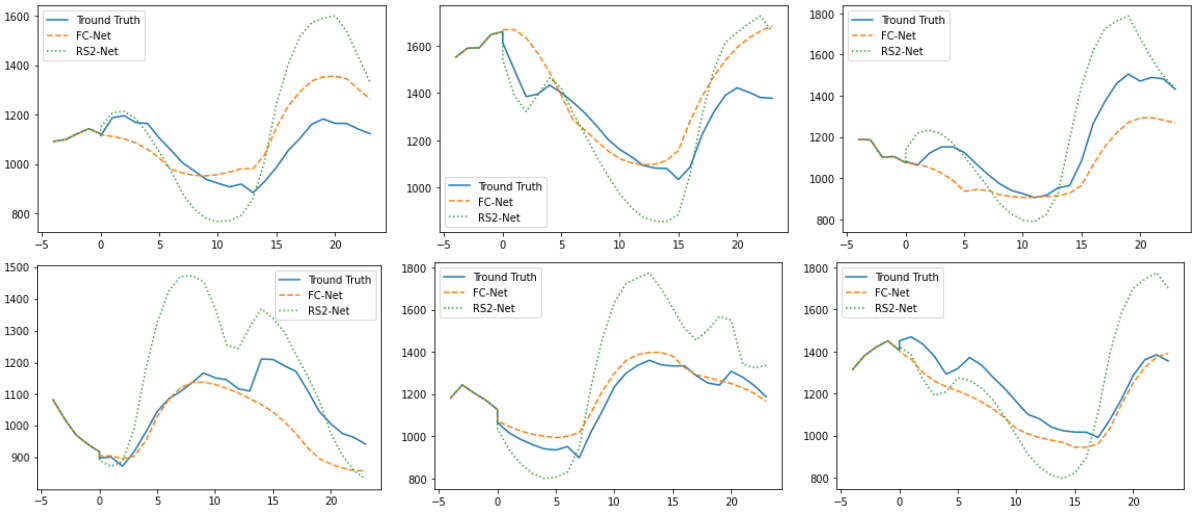
Model	L1-Loss
Prev*	46.8332
FC-Net	**28.3152**
R-Net	31.4342
R2-Net	34.6987
RS-Net	31.6077
RS2-Net	30.8992

Based on the table above, we observe that the models outperform the baseline model significantly. The simple model, FC-Net achieves the best result in this experiment. Additionally, the more complex or larger models, such as R-Net and the extension of it, are not significantly better than the simple model.

Note*: Prev made prediction based on only one prior information, namely the previous electricity load, and return it as the prediction. This method which does not undergo training process is picked as the baseline model.

2. Qualitative Evaluation

To better visualize the performance of the model, we use the pre-trained model to predict the future electricity demand and plot the prediction. For simplicity, Prev is excluded from the plot since it will predict a constant value, and the following plots include only the Ground Truth, FC-Net prediction, and RS2-Net prediction. In FC-Net, we predict the electricity load of the next 24 subsequent timeframes conditioning on 1 observed timeframe. On the other hand, RS2-Net, we predict the electricity load of the next 24 subsequent timeframes conditioning on 5 observed timeframes. The current prediction will be used as a replacement towards the ground truth in the subsequent prediction. In other words, the prediction of the current electricity demand is fed into the model for future and further prediction. Longer testing period than ones in training is employed to demonstrate the generalization capability of each model. The x-axis represents time in hours where non-positive values are the timesteps when the prior information are available and positive values represents the time when the electricity load will be predicted. The y-axis represents the electricity load.



The plots above shows that the models could predict the future electricity load. RS2-Net is shown to be more capable in capturing the minor rise and fall of the demand, but it is possible that the model is too flexible such that the changes are amplified. FC-Net, on the other hand, follows the changes of the electricity demand while it could not capture the minor changes.

3. Discussion

The results show that the proposed FC-Net and R-Net both are capable of predicting the electricity load with prior information. Although the error achieved by FC-Net is slightly better than R-Net, the qualitative assessment allows use to evaluate the models in a different way. When building machine learning model, it is important to note the bias and variance trade-off to achieve a good result. In this project, the R-Net may predict the increase and decrease of the electricity load. However, the flexibility comes with a cost which is a higher variance. This is shown by the extreme rise and fall in the prediction by RS2-Net. On the other hand, FC-Net generates a smoother prediction while moving along with the ground truth. FC-Net could not capture a brief and short dip or increase in the ground truth because it is not flexible enough to capture the minor changes. This project shows that machine learning has the possibility of solving such critical problem. In the future, other machine learning models might be able to forecast electricity load with higher accuracy.

VI. Reference

1. Lecture Notes and Tutorial Notes