MULTIVARIATE TIME SERIES FORECASTING WITH N-BEATS ARCHITECTURE

Duy Ngoc Tran - Supervised by Dr. Wei Huang

I. INTRODUCTION

Time series forecasting is a technique to predict a response variable over a period of time based on historical data. It plays a vital role in many domains, ranging from financial and quantitative trading to air quality prediction of pollutants - PM2.5.

This paper focuses on a multivariate time series forecasting algorithm for air quality performed on **Beijing Air Quality Dataset** by using machine learning and deep learning models. More importantly, this paper also implements multivariate **N-BEATS** architecture - a state-of-the-art model, which is a pure neural network that beats Sequence to Sequence(seq2seq) models such as ES-RNN in its original paper .

Keywords: multivariate time series forecasting, N-BEATS, deep learning models

II. METHODOLOGY

Based on a time series $\{\boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_w}\}$ where $\boldsymbol{x_i}$ is a vector of n features at time i, a time series forecasting algorithm needs to learn the input and returns a function that maps input to predicted values for response variable y from time $w+1$ to $w+k$ for some pre-chosen $k \geq 1$. In our experiment, the default value for w is 96, and k is 8, which means based on data from the previous 96 hours, we are going to predict PM2.5 in the next 8 hours.

$$y_{w+i} = f_i(\boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_w}) + \epsilon_i \text{ for } 1 \leq i \leq k \tag{1}$$

where y_{w+i} is the response variable value for time $w+i$, w is the window size, k is the future length we need to predict, f_i is the prediction function at time $w+i$ learned from data, and ϵ_i is the random error - $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$

Data preprocessing:

The dataset that we use for experimentation is Beijing Multi-Site Air-Quality Dataset. The dataset consists of 35065 rows in total, and was collected hourly from January 1, 2013 to December 31, 2017. The target variable is "PM2.5" column, and all other variables are used as input for machine learning models as outlined in Eq.(1).

Figure 1: Data preprocessing pipeline

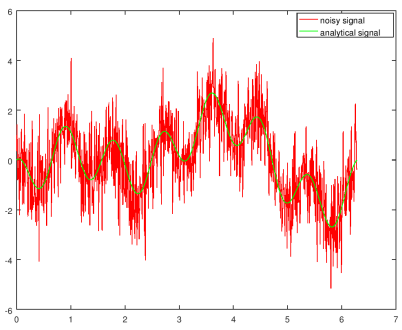


Figure 2: FFT Denoising Method

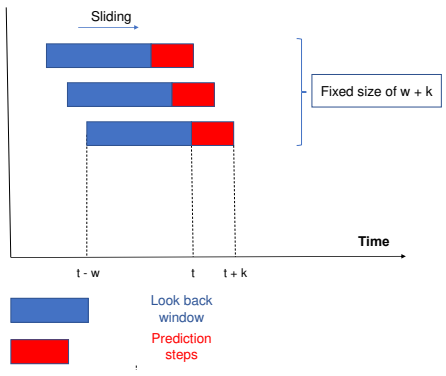


Figure 3: Sliding window technique

III. N-BEATS ARCHITECTURE VISUALIZATION

1. Basic block:

The basic block is a multi-layer FC (fully-connected) network, which comprises of 2 main parts and for summarization purpose, we focus on how the lth block operates. It receives its input $\boldsymbol{x}_l$ and returns backcast and forecast outputs, which are $\hat{\boldsymbol{x}}_l$ (block's best estimate of $\boldsymbol{x}_l$) and $\hat{\boldsymbol{y}}_l$ (block's forward forecast) respectively.

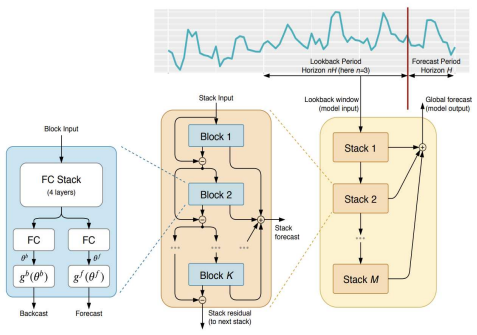


Figure 4: N-BEATS architecture

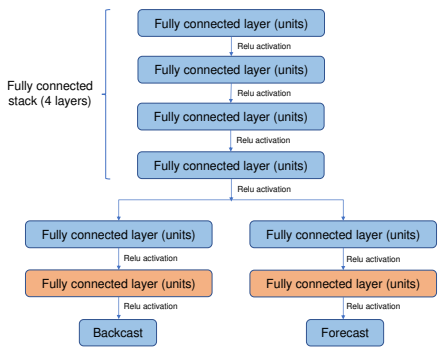


Figure 5: Basic block architecture

FC Network is a fully connected network denoted as blue in Figure 5, which is a network of standard fully connected layer with ReLU activation function (Nair Hinton, 2010). For the first block of the architecture, $\boldsymbol{x_l}$ is the overall model input, and after passing through corresponding FC Network, $\boldsymbol{\theta_l^f}$ and $\boldsymbol{\theta_l^b}$ are obtained.

$$\boldsymbol{\theta_l^f} = h_f(\boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_w}) \qquad \boldsymbol{\theta_l^b} = h_b(\boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_w}) \tag{2}$$

for h_f and h_b be functions of FC forward and backward network, respectively.

Basis layers (orange part in Figure 5) consist of backward g_l^b and forward g_l^f layers, which receive corresponding backward $\boldsymbol{\theta_l^b}$ and forward $\boldsymbol{\theta_l^f}$ to output $\boldsymbol{x}_l$ and $\boldsymbol{y}_l$, respectively. Further examples of g_l^b and g_l^f for time series data will be introduced in section IV.

IV. N-BEATS ARCHITECTURE (cont.)

Basis layers can vary depending on configuration of the architecture. For purpose of time series, trend and seasonality decomposition are introduced to the basic block. Given $\boldsymbol{\theta}_{s,l}^{\boldsymbol{f}}$ and $\boldsymbol{\theta}_{s,l}^{\boldsymbol{b}}$ be polynomial coefficients predicted by FC forward and backward network of layer l stack s, the trend and seasonality forecast in matrix form are represented in Eq.(3) and Eq.(4).

Trend Decomposition (pre-chosen p):

$$\boldsymbol{y}_{s,l}^{tr} = \boldsymbol{T}\boldsymbol{\theta}_{\boldsymbol{s,l}}^{\boldsymbol{f}} \qquad \boldsymbol{x}_{s,l}^{tr} = \boldsymbol{S}\boldsymbol{\theta}_{\boldsymbol{s,l}}^{\boldsymbol{b}} \tag{3}$$

for **t** = $[0\ 1\ 2\ ...\ k-1]^T/k$; **T** = $[\mathbf{1}\ \mathbf{t}\ ...\ \mathbf{t}^p]$ **s** = $[0\ 1\ 2\ ...\ w-1]^T/w$; **T** = $[\mathbf{1}\ \mathbf{s}\ ...\ \mathbf{s}^p]$

Seasonality Decomposition:

$$\boldsymbol{y}_{s,l}^{seas} = \boldsymbol{U}\boldsymbol{\theta}_{\boldsymbol{s,l}}^{\boldsymbol{f}} \qquad \boldsymbol{x}_{s,l}^{tr} = \boldsymbol{V}\boldsymbol{\theta}_{\boldsymbol{s,l}}^{\boldsymbol{b}} \tag{4}$$

for **U** = $[\mathbf{1}, \cos(2\pi\mathbf{t}), ..., \cos(2\pi\lfloor k/2-1\rfloor\mathbf{t}), \sin(2\pi\mathbf{t}), ..., \sin(2\pi\lfloor k/2-1\rfloor\mathbf{t})]$

V = $[\mathbf{1}, \cos(2\pi\mathbf{s}), ..., \cos(2\pi\lfloor w/2-1\rfloor\mathbf{s}), \sin(2\pi\mathbf{s}), ..., \sin(2\pi\lfloor w/2-1\rfloor\mathbf{s})]$

2. Doubly residual stacks

A stack is made up from multiple basic blocks. There are 2 residual branches in the architecture, backcast and forward operation shown as below

$$\boldsymbol{x}_{s,l} = \boldsymbol{x}_{s,l-1} - \hat{\boldsymbol{x}}_{s,l-1} \quad ; \quad \boldsymbol{y}_s = \Sigma_l \mathbf{y}_{s,l} \tag{5}$$

Regarding the former, input to each subsequent basic block is the element-wise difference between the true input and backcast prediction from the previous basic block, which supports more fluid gradient propagation. Similarly, stack forecast output $\boldsymbol{y}_s$ is the sum of all of its blocks' forecasts as shown in Eq.(5) . Finally, the global forecast $\mathbf{y}_{global}$ (model output) can be calculated as the sum of its stacks' forecast outputs.

$$\mathbf{y}_{global} = \Sigma_s \boldsymbol{y}_s = \Sigma_s \Sigma_l \boldsymbol{y}_{s,l} = \Sigma_{s_1}\Sigma_{l_1}\boldsymbol{y}_{s_1,l_1}^{tr} + \Sigma_{s_2}\Sigma_{l_2}\boldsymbol{y}_{s_2,l_2}^{seas} = \Sigma_{s_1}\Sigma_{l_1}\boldsymbol{T}\boldsymbol{\theta}_{\boldsymbol{s_1,l_1}}^{\boldsymbol{f}} + \Sigma_{s_2}\Sigma_{l_2}\boldsymbol{U}\boldsymbol{\theta}_{\boldsymbol{s_2,l_2}}^{\boldsymbol{f}} \tag{6}$$

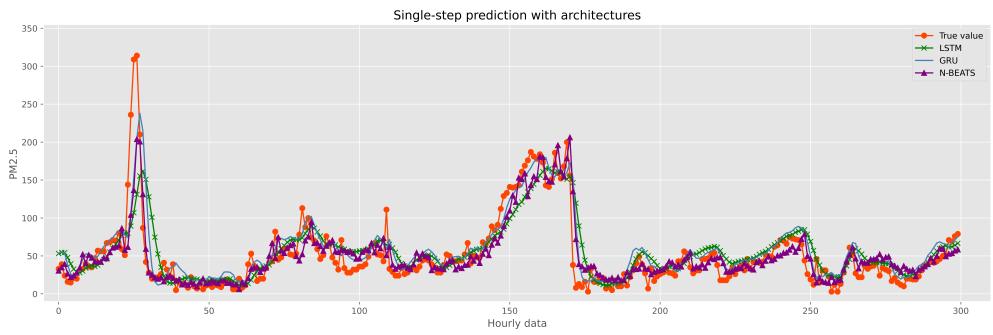
both $\boldsymbol{\theta}_{\boldsymbol{s_1,l_1}}^{\boldsymbol{f}}$ and $\boldsymbol{\theta}_{\boldsymbol{s_2,l_2}}^{\boldsymbol{f}}$ depend on $\mathbf{X} = \{\boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_w}\}$, which is used to conduct back-propagation to train the neural network.

V. EXPERIMENTATION AND RESULTS

- In this experiment, we evaluate N-BEATS architecture's performance with other recurrent networks: LSTM, and GRU on different prediction steps. Although GRU outperforms LSTM, based on results from table and 2 figures below, it is N-BEATS that is the best model among experimented deep learning architectures for both a prediction step of 1 hour and 8 hours.
- Moreover, as the prediction step increases, the errors observed in all models also increase, which is consistent with our intuition that the more uncertain about the future, the greater error models are likely to make.

	LSTM		GRU		N-BEATS	
Prediction steps	MAE	RMSE	MAE	RMSE	MAE	RMSE
1 hour	21.88	34.84	16.83	26.69	17.63	24.98
8 hours	38.23	58.98	36.14	56.83	35.01	54.36

- Another key conclusion is that the greater the window size w, the greater error models attain, which is reasoned by conditional on more recent data, the response variable and older predictors are weakly related. Therefore, using a large window size w will introduce a higher dimension of covariates with few useful information, resulting in a decline in all architectures' performances.



- Due to computational complexity, large data size, and limited resources, further experiments with different look-back windows and prediction steps are currently restricted. Therefore, this paper suggests N-HIST architecture, a recent development of N-BEATS which can further address computational cost and improves accuracy.

VI. REFERENCES

[1] Boris N. Oreshkin, Dmitri Carpov, Nicolas Chapados, Yoshua Bengio (2020) *N-BEATS: Neural basis expansion analysis for interpretable time series forecasting*

[2] Time Series Forecasting (TSF) Using Various Deep Learning Models (2022) *Jimeng Shi, Mahek Jain, Giri Narasimhan*

[3] Keshav G (2020) *N-BEATS: NEURAL BASIS EXPANSION ANALYSIS FOR INTERPRETABLE TIME SERIES FORECASTING* [Online]

[4] Arne Vogel (2017) *Denoising Functions in Matlab With FFT* [Online]