# *A Deep Learning Approach for solving Non-Linear Partial Differential Equations*

*Final report submitted in the partial fulfillment of the requirement for the award of the degree of*

## Master of Technology

Submitted By


**Mr. Myakala Pruthvi**
17MF3IM07

Under the Supervision of
**Dr. Adway Mitra**

Centre for Excellence in Artificial Intelligence
Indian Institute of Technology Kharagpur


November, 2021

Centre for Excellence in Artificial Intelligence
Indian Institute of Technology Kharagpur
West Bengal, India – 721302

# CERTIFICATE

This is to certify that the project titled "A Deep Learning Approach for solving Non-Linear Partial Differential Equations" is a bona-fide work carried out by ***Mr. Myakala Pruthvi***, Roll no. 17MF3IM07 to the Centre for Excellence in Artificial Intelligence, Indian Institute of Technology Kharagpur, under my supervision and guidance. This report, in my opinion, is worthy of consideration for partial fulfillment of requirements for the degree of Master of Technology in Industrial and Systems Engineering in accordance to the regulations of this institute.

Date: 24th November, 2021
Place: Kharagpur

Dr. Adway Mitra
Centre for Excellence in Artificial Intelligence
Indian Institute of Technology Kharagpur
Kharagpur-721302, India

# ACKNOWLEDGEMENT

I express my deep sense of gratitude to my supervisor Dr. Adway Mitra for his valuable guidance and inspiration throughout the course of this work. I am thankful to him for his help, support and motivation towards innovative thinking. Even in times of the pandemic and online semester, it is all due to his help that I could finish my work. It has been an enriching experience working under him in a cordial environment.

# DECLARATION

I certify that:

- The work contained in this report has been done by me under the guidance of my supervisor

- The work has not been submitted to any other institute for any degree of diploma

- I have conformed to the norms and guidelines given in Ethical Code of Conduct of the Institute

- Whenever we have used materials (data, theoretical analysis, figures, and text) from other sources, we have given due credit to them by citing them in the text of the thesis and giving their details in the references. Further, we have taken permission from the copyright owners of the sources, whenever necessary.

# Abstract

Name of the student: **Myakala Pruthvi** Roll No: **17MF3IM07**

Degree for which submitted: **Dual Degree**

Department: **Department of Industrial and Systems Engineering**

Thesis Title: **A Deep Learning Approach on solving Non-Linear Partial Differential Equations**

Thesis supervisor: **Dr. Adway Mitra**

Month and year of thesis submission: **November 2021**

In general, it is difficult to solve non- linear Partial Differential Equations (PDEs), the easiest being those which are integrable. So to solve these PDEs various numerical techniques have evolved over the past. Here, we looked to apply Deep Learning techniques to solve these equations, in particular the Burger's Equations, at the very time obeying the underlying physics that govern them. Given the data for solution set, we have solved for Burger's Equation, first using simple FNNs, then applied LSTNet –Based model and performed a comparative study with the numerical methods.

# Table of Contents

# 1. Introduction

It has always been challenging to solve Non-Linear Partial Differential Equations, to an extent being that one of the Millennium problems is to find a reasonable solution for Navier Stokes Equation. Here, we attempt to solve these PDEs using deep learning techniques. As an exercise I have looked to apply these methods to solve the Burger's Equation, which has applications ranging from Acoustics to even traffic flow.

The field of Machine Learning has shown tremendous results in a number of fields from computer vision to probabilistic modelling. Over the recent course of time, Machine Learning has found its way in scientific research too. With the recent developments in computing resources and availability of data, and advances in machine learning & data analytics have yielded great results across diverse scientific disciplines.

We employ deep neural networks and leverage their well-known capability as universal function approximators [1], thus we use the Physics Informed Neural Networks [2] (which are nothing but an extension of simple Feed Forward Neural Networks) to solve the supervised learning tasks while respecting the underlying physical laws. Then we applied LSTNet model and compared the results with the traditional numerical methods.

# 2. Research Objectives

In this study, we tackled the problem of solving Non-Linear PDEs using deep learning models such as LSTNet. The PDE in discussion here is the Burger's Equation, which was originally proposed as a simplified form of the famed Navier Stokes Equation in the context of turbulent flows. The main objective of this study moving forward is to find a reasonably efficient way to solve this PDE using the recently deep learning techniques. First we applied simple FNNs to solve the PDE, then applied LSTNet model to predict the velocity field.

# 3. Literature Review

The term physics-informed machine learning has been used recently by Wang et al. [4] in the context of turbulence modeling. Other examples of machine learning approaches for predictive modeling of physical systems include [5-8]. All these approaches employ machine learning algorithms like support vector machines, random forests, Gaussian processes, and FNNs/RNNs/CNNs. We have noticed that this approach introduces a regularization mechanism that allows us to use relatively simple Feed-Forward Neural Network architectures and train them with small amounts of data.

Here we look at the concepts involved/applied in our study

## 3.1. Feedforward Neural Networks:

The feedforward network, is the simplest form of a Neural network. The main difference between a Feedforward and a Recurrent neural network is in the way they channel information through a series of mathematical operations performed at the nodes of the network. As shown in the figure, there are no loops between the nodes and hence, no piece of information ever passes through a neuron twice. Due to the architecture of the network, the output of one layer does not affect the preceding one. An important point to note regarding these networks is that they do not possess any "memory". This means that all the outputs that a Feedforward network gives in independent of the previous outputs it produces.

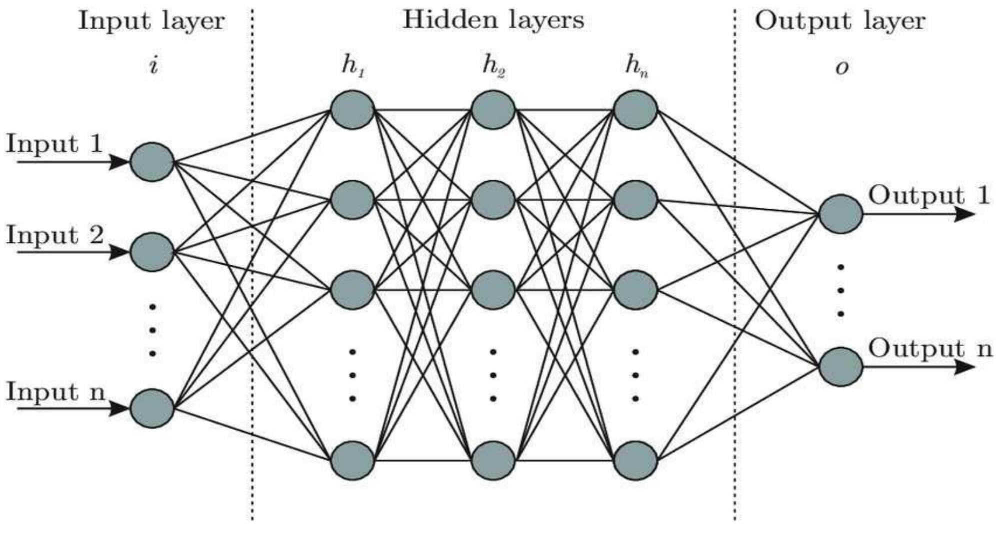



Fig: Neural Network Architecture

The $x_i$ denote the input nodes. This is where the input data enters the network. Depending upon the network architecture, i.e., the number of hidden layers, the number of nods per layer, and the connection between the nodes, we define weight matrices for each layer denoted by $W_{i,j}^{(l)}$, where i and j are the $i^{th}$ and $j^{th}$ nodes of the l th and the (l-1)th layers respectively. The input for the $i^{th}$ node then becomes the weighted sum of the connections along with a bias value denoted by $b$ $(l)$ which is generally 1. Therefore, we get the input for the ith node in the l th layer as: $z_i = \sum_{j=0}^{n} x_i W_{i,j}^{(l)} + b^{(l)}$ And its output as: $a_i^l = g(z_i)$, where g is the activation function, usually a sigmoid transformation. Nodes of different layers can have different functions.

## 3.2 Long Short-Term Memory

The Long Short-Term Memory Neural Network is a special type of Recurrent Neural Networks which have a special ability of actually storing important events in pattern recognition and use them for prediction. This is especially helpful in time series analysis where there might be significant gaps in between important events. These networks contain a gated cell which stores information outside the normal flow of the network. The cell decides what to store, reading, writing and erasing of data just like that of computer memory.

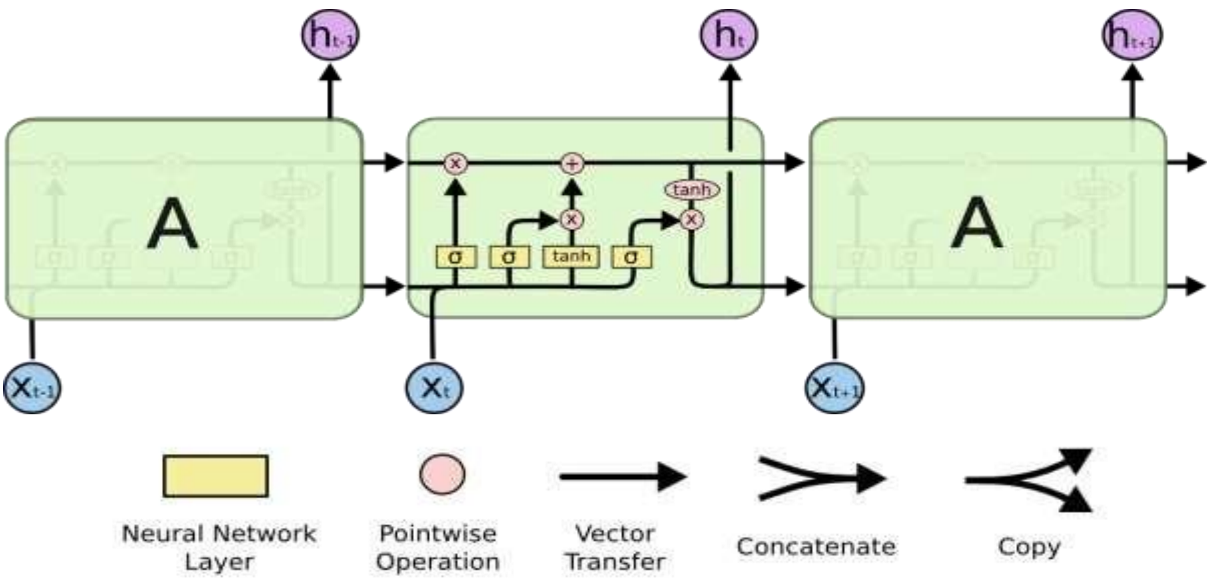



Fig: LSTM Architecture

With this process, we can see how normal LSTM models are able to “store” the data of the long-term pattern

## 3.3 LSTNet Model

This network is basically divided into three building blocks, convolutional component, recurrent component, and a feed forward component. The first layer of the network is the Convolutional Component. With the help of convolutional layer, the model aims to extract the local dependencies between the variables. Alongside, it also helps in the extraction of short-term pattern in the time-series. The output of the convolutional component is simultaneously fed into parallel layers of the recurrent component comprising of LSTM layers.

One recurrent layer is a LSTM layer which ensures the proper mapping of the features. The other recurrent layer comprises of the LSTM layer with skip links between these layers. It is specifically designed to learn the long term dependencies along with the periodic patterns. Then a dense layer is used to combine both these two recurrent layers and produce the output. The output of the layer is the prediction.

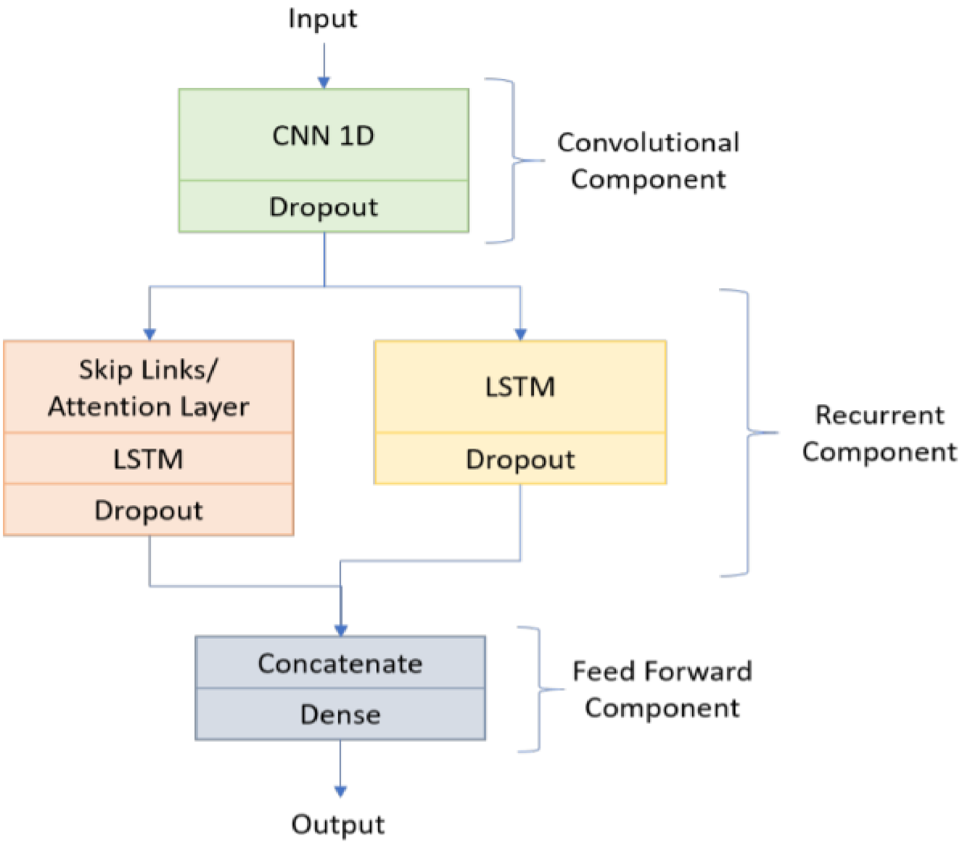



Fig: LSTNet Architecture

## 3.4 Runge Kutta Time Stepping Scheme

The general form of Runge – Kutta method with q stages for equation

$$u^{n+C_i} = u^n - \Delta t \sum_{j=1}^{q} a_{ij} \mathcal{N}[u^{n+C_j}]. \qquad i = 1, 2 \ldots q$$
$$u^{n+1} = u^n - \Delta t \sum_{j=1}^{q} b_j \mathcal{N}[u^{n+C_j}].$$

Here, $u^{n+1}(x) = u\,(t^n + C_j \Delta t, x)$ for $j = 1 \ldots$ q. This encapsulates both implicit and explicit time stepping schemes depending on the choice of parameters $\{a_{ij}, b_j, c_j\}$

# 4. Problem Setup

We look to solve the following one dimensional Burger's Equation.

$$u_t + \mathcal{N}[u; \lambda] = 0 \qquad x \in \Omega, t \in [0, T] \qquad \ldots\ldots (1)$$

Where u(t, x) is the latent (hidden) solution, $\mathcal{N}[.]$ is a non-linear differential operator parameterized by $\lambda$ and $\Omega$ is a subset of $\mathbb{R}^D$.

For Burger's Equation we have $\mathcal{N}[u; \lambda] = \lambda_1 u u_x - \lambda_2 u_{xx}$ and $\lambda = (\lambda_1, \lambda_2)$.

Here, we solve the Burger's Equation with Dirichlet Boundary condition, where $\lambda_1 = 1 \; and \; \lambda_2 = \frac{0.01}{\pi}$, u(0,x)=-sin($\pi x$) and u(t,-1)=u(t,1)=0.

Here, we applied simple Feed Forward Neural Networks architecture and LSTNet model to predict the velocity field i.e., u(t, x).

# 5.Solution Methodology

Consider the left hand side of equation 1. i.e.;

$$f \coloneqq u_t + \mathcal{N}[u; \lambda] \qquad \text{...... (2)}$$

We proceed my approximating u(t, x) by a deep neural network, in the form of equation 2. The error in neural networks u(t, x) can be realized by minimizing the root relative squared error loss, where $u_m$ is the mean of all the $u^i$s.

$$RRSE = \frac{\sqrt{\sum_{i=1}^{N}\left|u(t_u^i, x_u^i) - u^i\right|^2}}{\sqrt{\sum_{i=1}^{N}\left|u(t_u^i, x_u^i) - u_m\right|^2}} \qquad \text{.......(3)}$$

We convert $u(t, x)$ into Feed forward neural networks $u(t, x; W, b)$ by using automatic differentiation. The model should exactly preserve the continuity and momentum equations by construction. Using training, given a dataset $\{t_i, x_i, u_i\}$ and $\{t_j, x_j, f_j\}$ we find best fit $W^*$ and $b^*$. We use only few hundreds of collocation points of $N$ and optimize by advanced batch gradient based optimization algorithm and use hyperbolic tangent activation function.

Then we applied LSTNet model for the prediction of u(t, x). This model consists of various CNN layers, RNN layers and Skip RNN layers. We have used selu activation function for all the layers mentioned above. Then we calculated the RRSE in equation 3 along with the correlation between the predicted and the true data and also plotted for prediction vs actual data for increasing number of predictions.

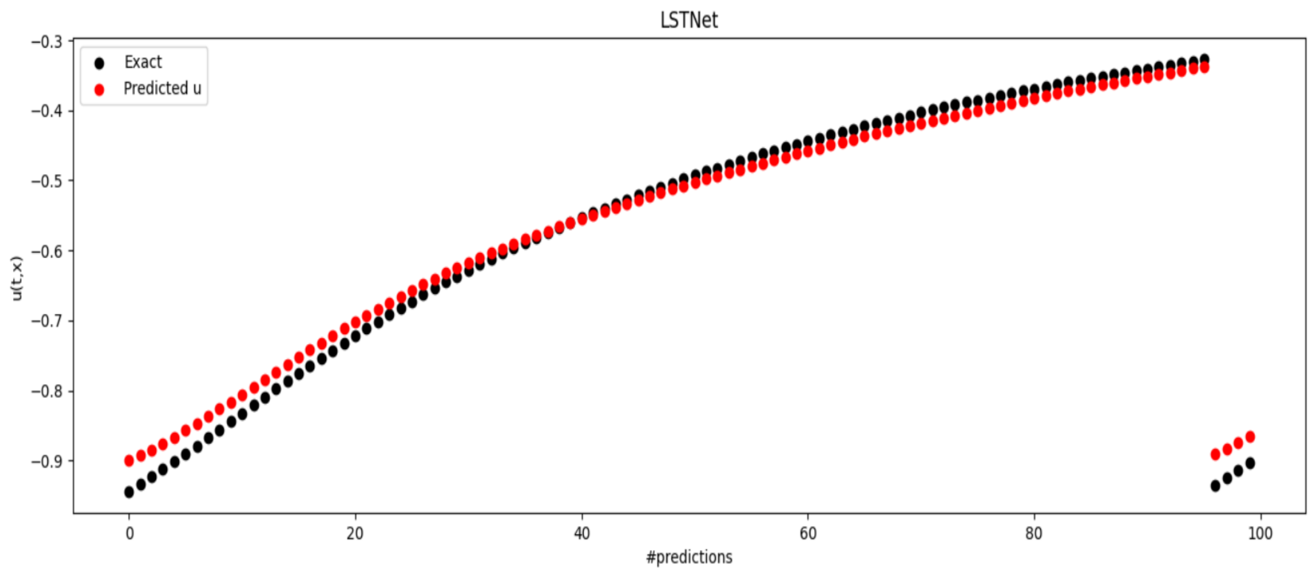
# 6. Data Collection

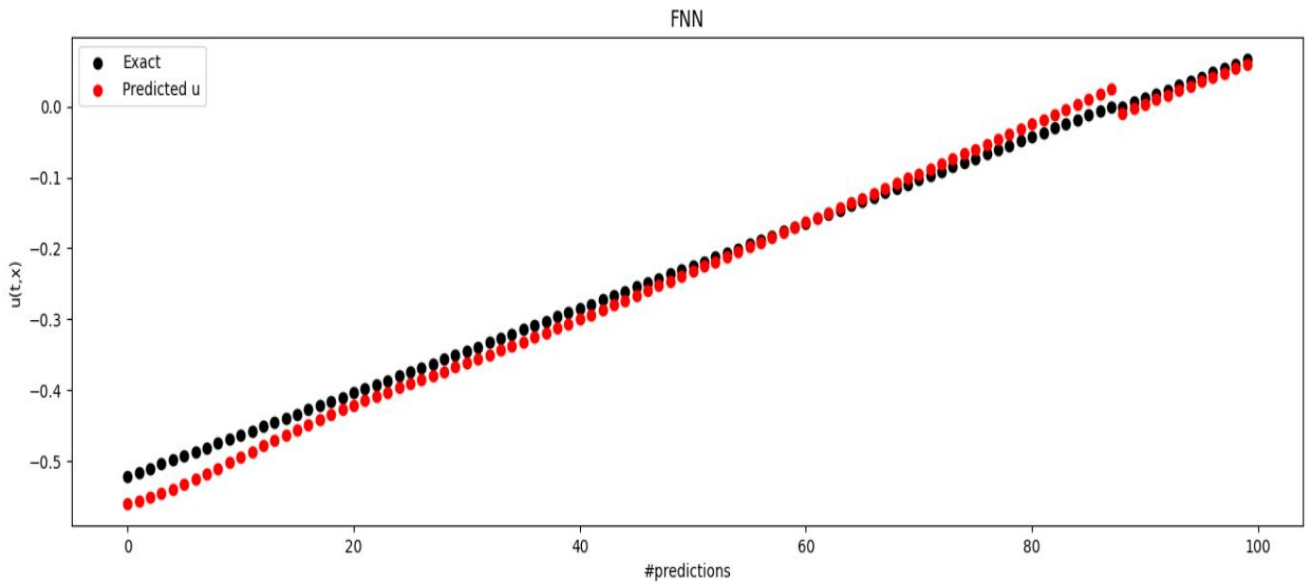
The data required for this prediction has been secured from the paper in reference [2], where we have secured data for velocity field i.e. u(t,x) for x ranging from -1 to 1 in uniform distribution (256 values) and t ranging from 0 to 1 with an increment of 0.01 (100 values). This gave us the solution set of 25,600 values of u(t,x) for each of the points from the range mentioned above.

# 7. Results

As mentioned above we have tabulated the predictions for both the approaches mentioned in section 4 of the report along with their plots.

| S.No | Method | RRSE (in %) | Correlation Coefficient | Mean Absolute Error |
|---|---|---|---|---|
| 1. | FNN | 10.739 | 99.87605 | 0.0154 |
| 2. | LSTNet | 9.989 | 99.97175 | 0.0161 |



# 8. Conclusions and Future Work

With similar study carried out in reference [2], where they have arrived at an error of the order $10^{-2}$ to $10^{-3}$ using the FNNs with different Neural architecture and numerical methods like Runge Kutta Time stepping schemes. Our study, if anything at-least matches the accuracy of the research paper in reference.

It is clear from the above results that Deep Learning can definitely be pursued as one of the methods of solving Non-linear Partial Differential Equations like the Burger's Equation. It is one of those avenues in Computational Physics which has a huge scope of solving the traditional problems and performing better than the existent numerical methods.

Our work in the next semester will focus on fast emulation of these neural networks and also focus on techniques to improve the accuracy in these predictions.

# 9. References

[1] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Netw. 2 (1989) 359–366.

[2] Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations – ScienceDirect.

[3] A.G. Baydin, B.A. Pearlmutter, A.A. Radul, J.M. Siskind, Automatic differentiation in machine learning: a survey, 2015, arXiv:1502.05767.

[4] J.-X. Wang, J. Wu, J. Ling, G. Iaccarino, H. Xiao, A comprehensive physics-informed machine learning framework for predictive turbulence modeling, 2017, arXiv:1701.07102.

[5] Z.J. Zhang, K. Duraisamy, Machine learning methods for data-driven turbulence modeling, in: 22nd AIAA Computational Fluid Dynamics Conference, 2015, p. 2460. [6] M. Milano, P. Koumoutsakos, Neural network modeling for near wall turbulent flow, J. Comput. Phys. 182 (2002) 1–26.

[7] P. Perdikaris, D. Venturi, G.E. Karniadakis, Multifidelity information fusion algorithms for high-dimensional systems and massive data sets, SIAM J. Sci. Comput. 38 (2016) B521–B538.

[8] R. Rico-Martinez, J. Anderson, I. Kevrekidis, Continuous-time nonlinear signal processing: a neural network based approach for gray box identification, in: Neural Networks for Signal Processing IV. Proceedings of the 1994 IEEE Workshop, IEEE, 1994, pp. 596–605.

# 10.Appendix

- Link to data set: https://drive.google.com/file/d/1molVs93e86WsRC-7nrqbfRp156R-90jq/view?usp=sharing
- Link to code(FNN): https://colab.research.google.com/drive/1LNyA00p-4jPl_rkhICWq8hh1H7OmkwfY?usp=sharing
- Link to code(LSTNet): https://colab.research.google.com/drive/1wUzX2oYYf57R0h4zgLfK7COvvdwTBZ8k?usp=sharing