National Chung Hsing University

College of Science

Department of Applied Mathematics

Structural Machine Learning Models and Their Applications (6582)

Final Report

Name	: KVDTT Abeywardhana.
Student ID	: 7108053024.

Implement results

I have selected two articles which are mainly focus on Neural Network with tensorflow. Here onwards, I discussed about their implement results.

__First paper:__

Physics-informed neural network: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations

Brief Introduction:

In this paper mainly focus on how to find Data-driven solution and data-driven discovery of partial differential equation (PDE). That is, if we have a PDE, there is a particular solution (Data-driven solution) which is satisfied the PDE with the given initial and boundary condition. And also, when we have a Data-driven images, we can find the which kind of PDE is behave in there. For that they have introduce a Neural network with tensorflow.

My implementation:

Inside their code, I have to change some library of functions to execute the newer version of tensorflow.

Inside the code	**Paper**	**My code implements**
Import data	Chebfun package - Matlab	Tailored Finite Point Method - Matlab
Activation functions	Sin, Tanh	Sin. Tanh, relu
tf placeholders for Identification	Tf.placeholder	Tf.compat.v1.placeholder
Optimizer for Identification	Tf.contrib.opt.ScipyOptimizer -Interface	Tf.compat.v1.train.Gradient -DescentOptimizer Tf.keras.optimizer.SGD
Optimizer for Identification	Tf.train.Adamoptimizer	Tf.compat.v1.train.Adamoptimizer

However, when I run the code it ended up with an “Eger execution” error. So, I have downgrade the tensorflow and worked on their code.

Results:

I have selected the Burgers equation and implement its results for three types of low diffusion terms. In the paper they have imported Data-driven image which were Burger equation solved by Matlab using Chebfun package. For my data, I have construct my own data by solving the Burgers equation using Tailored Finite Point method (More detail's in the presentation file).

In each figure, I have randomly selected 100 points from the boundary and trained he data and with the inner data and then predicted the data-driven solution. In below we can see the results. Also, in each figure contain Top layer and Bottom layer. Top: Predicted solution $u(t, x)$ by PINN for various diffusion in TFPM. Bottom: Comparison of the predicted and exact solutions corresponding to the three temporal snapshots depicted by the white vertical lines in the top panel.

1. When the diffusion = 0.1 for the Burgers equation (PDE).

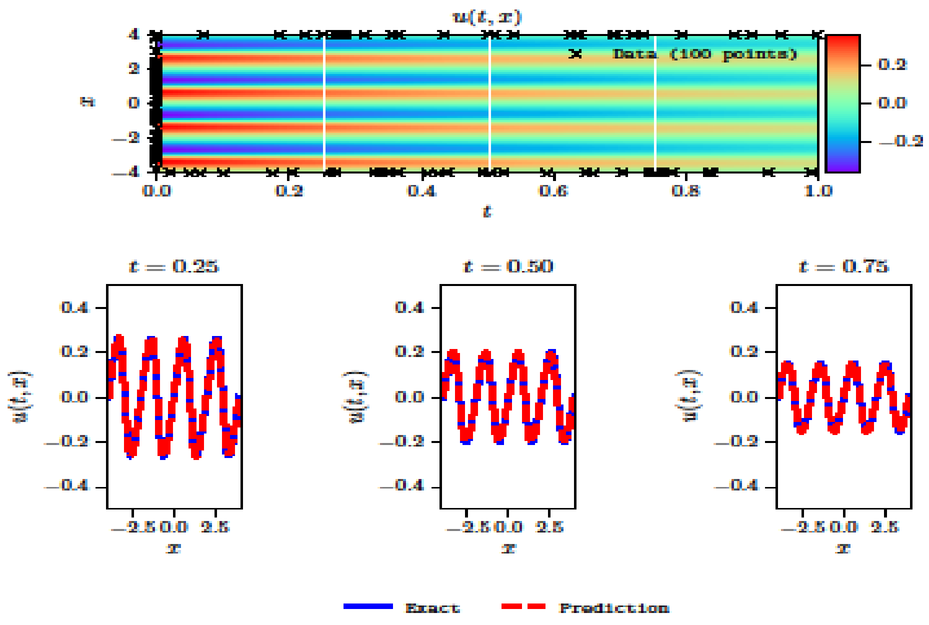

Figure 1 : Results for 0.1 diffusion

2. When the diffusion = 0.01 for the Burgers equation (PDE).

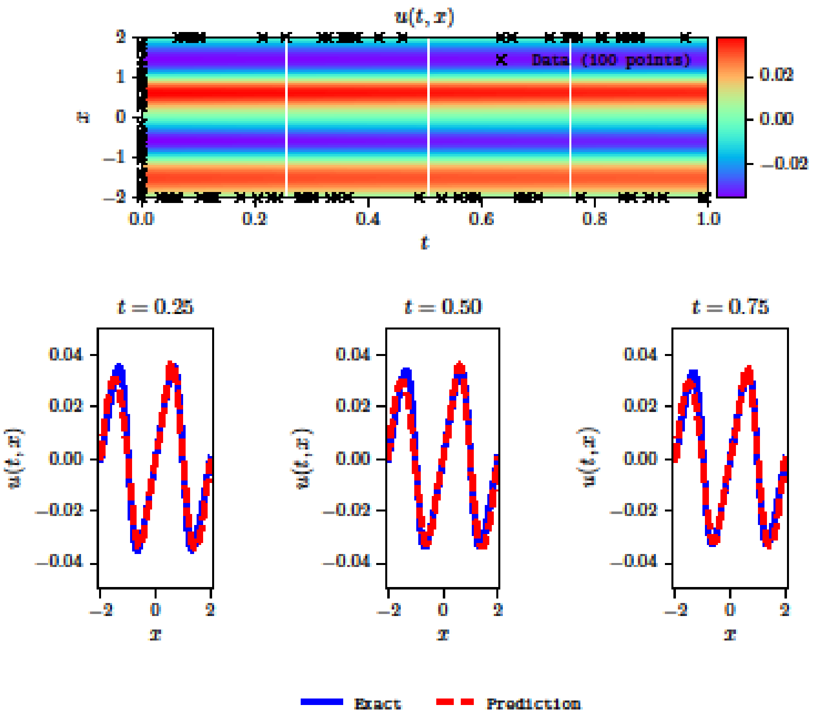

Figure 2 : Results for 0.01 diffusion

3. When the diffusion = 0. 001 for the Burgers equation (PDE).

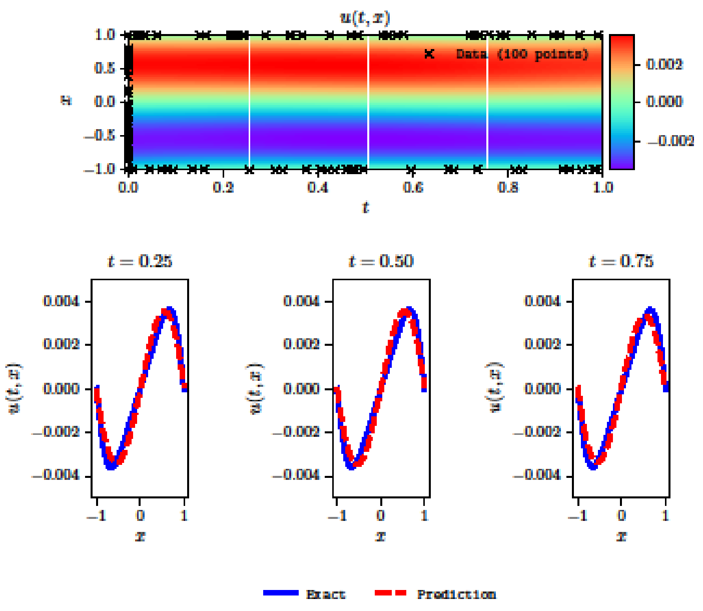

Figure 3 : Results for 0.001diffusion

Conclusion:

For the low diffusion term, we can catch the Data-driven solution of PDE by machine learning methods.

<u>Second paper:</u>

Deep Hidden Physics Models: Deep learning of Nonlinear Partial Differential Equations.

Brief Introduction:

In this paper,

- Method is how to discover closed form mathematical models of the physical world expressed by partial differential equations from scattered data collected in space and time.
- we construct structured nonlinear regression models that can uncover the dynamic dependencies in a given set of spatio-temporal data, and return a closed form model that can be subsequently used to forecast future states.
- Here, we need training data set and test data to predict the underline PDE.
- We have to rely on previous method discussed in first paper called Physics Informed Neural Network (PINN).

My implementation:

As my implements, I have selected numerically solved data form Matlab Chebfun as the training data set. For the test data, I have used my own construct TFP data.

In the paper they have used sin and tan function as the activation function. In this implement I have used RELU activation function as well. The results are given below.

Results:

Comparison results for the exact dynamic of Burgers' equation and its learned dynamics given below. I just only used for 0.001 diffusion equation to achieve the results.

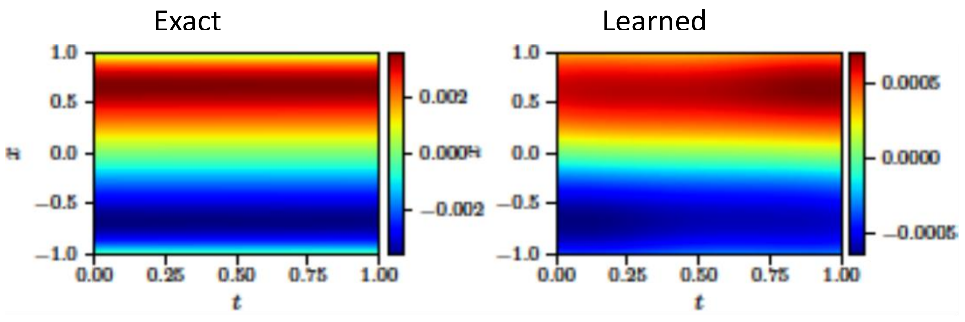

Figure 4: A solution to the Burger's equation (left panel) is compared to the corresponding solution of the learned partial differential equation (right panel).

When I used RELU as the activation function for the 0.1 diffusion, the learned dynamic didn't catch well. I think it is because, RELU function gives negative values zero.

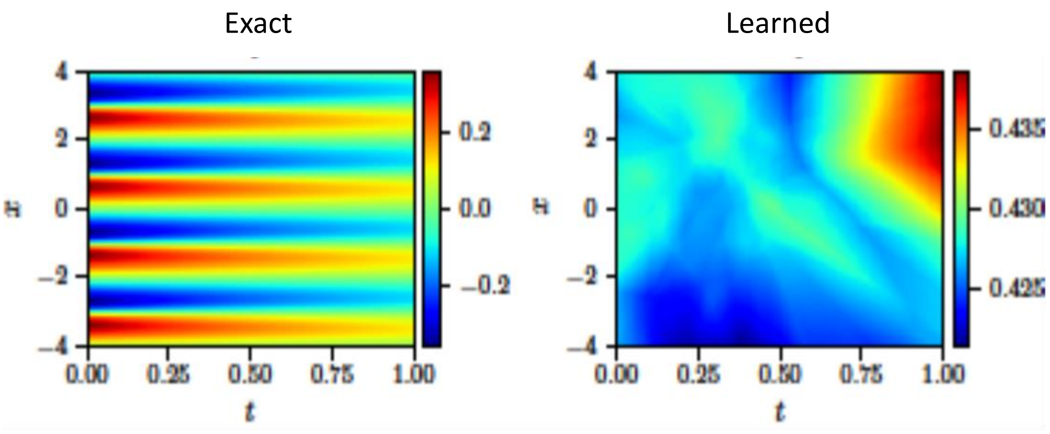

Figure 5: Results for RELU activation function

Conclusions:

- Activation fun ReLU is not suitable for both methods.
- Sinoid and Tanh activations functions give better results.
- Even the low diffusion terms can be approximate by the two methods by importing high resolution data.

References:

[1] M. Raissi, Deep hidden physics models: Deep learning of nonlinear partial differential equation, Journal of Machine Learning Research 19 (2018) 1-24.

[2] M. Raissi, P. Perdikaris, G. Karniadakis, Physics-informed neural networks: A deep learning of nonlinear partial differential equations, Journal of Computational Physics 378 (2019) 686-707.

[3] C.C. Tsai, Y.T. Shih, Y.T. Lin, H.C. Wang, Tailored finite point method for solving one-dimensional Burgers' equation, International Journal of Computer Mathematics 94 (2017) 800-812.