Differential Informed Auto-Encoder

Zhang Jinrui*
jerryzhang40@gmail.com

20241021

Abstract

In this article, an encoder was trained to obtain the inner structure of the original data by obtain a differential equations. A decoder was trained to resample from the domain of original data, to generate new data obey the differential structure of the original data using the Physics Informed Neural Network [3, PINN].

1 Introduction

If the physics formula was obtained in the form of differential equations, Physics Informed Neural network can be built to solve it numerically in a global scale [3, PINN].This process could be view as a decoder in a way that it takes a sample point in the domain of the partial differential equations, and solve it to get all the corresponding output of each input point. If only a small and random amount of training data was obtained, to resample from the domain we need to obtain the differential relationship of the data. This process could be viewed as an encoder that encodes the inner structure of the original data. And the decoder decode it by solving the differential equations.

2 Methodology

2.1 first approach

The first idea is simple. For a one variable function $u(t)$, define a second order differential equations in its general form $(\forall t)(F(\frac{d^2u}{du^2}, \frac{du}{dt}, u) = 0)$.

The data of the function $u(t)$ are given in tuples denote as $(T, U)_i \equiv (T_i, U_i)$. And It's natual to denote the differentials as U_i^t and U_i^{tt}. There are several method to compute these two differentials, including just using the definition of the derivative. In this article, local PCA are compute to obtain these differentials. Local PCA means to find the K nearest neighbors of a given point, which K is a hyperparameter, and perform PCA on these points close to each other

*Liu Zhekai,Fu Xiangshuo

to get the principal direction. The slope of this direction is the derivative U^t in general. Repeate these process on (T, U^t) to obtain U^{tt}

Create a FCN denote as f to represent $F(\frac{d^2u}{du^2}, \frac{du}{dt}, u)$ F to be 0 at every data points and to be 1 all elsewhere is wanted.

To achieve these requirement we evaluate f at all the data points, and train the network evaluate these point to 0. Then randomly sample points from the $\mathbb{R}^3$ and train these points to be 1.

Algorithm 1 f trainer

Require: Input parameters $f, T_i, U_i, U_i^t, U_i^{tt}$

Initialize f randomly
repeat
  $F_i \leftarrow f(U_i, U_i^t, U_i^{tt})$
  $RAND_i \leftarrow$ randomly sample in $\mathbb{R}^3$
  $R_i \leftarrow f(RAND_i)$
  $L \leftarrow meanSquareError(F_i, 0) + 0.1 * meanSquareError(R_i, 1)$
  backPropagation against L to optimize f
until L meets requirement
return f

Once The f was obtained we can perform PINN as a decoder to generate new data.

The experiment code for the Pics in Results can be run by python by program in Github [1, deSineTasks] The requirement environment may be installed by using [2, reqs]

3 Results

3.1 first approach

Train the model on a pure $sin(x)$ and try to get a result meet the initial condition with $U_0^t = 0.5$ which the $0.5 * sin(x)$ would be required output. Result shows in Figure 1.

4 Conclusion

Summarize the key outcomes and potential future work.

References

[1] Zhang Jinrui. Latex project. `https://github.com/unjerry/autoCluster/blob/master/deSineTask.bat`. Accessed: 2024-10-21.

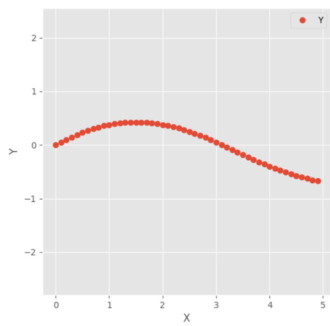

Figure 1: $0.5 * sin(x)$

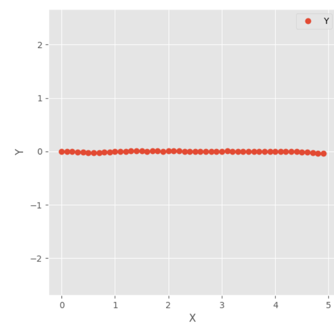

Figure 2: f errors

[2] Zhang Jinrui. Latex project. `https://github.com/unjerry/autoCluster/blob/master/reqs.bat`. Accessed: 2024-10-21.

[3] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10561*, 2017.