Physics-informed Convolutional Neural Networks for Temperature Field Prediction of Heat Source Layout without Labeled Data

Peter Knauss,[*] Thomas Sievert,[†] Wenzhong Hu,[‡] and Aashay Arora[§]
University of California, San Diego
(Group 7)

(Dated: March 24, 2023)

This project attempts to recreate the study by Zhao et. al to predict the steady state solutions of the heat equation for source layouts without any labeled data using a physics-informed convolutional neural network. In particular, we employ the use of a loss function containing a penalty term derived from the heat difference equation which guides the UNet architecture based neural network to learn to map intensity distributions to the solution space with accuracy close to that achieved using traditional numerical and data-driven techniques.

I. INTRODUCTION

Partial Differential Equations (PDEs) are used to model a wide range of physical phenomena, including fluid dynamics, heat transfer, and quantum mechanics etc. One application of one family of PDEs, called the heat equation, is to analyze the thermal design of electronic systems in the domain of the various electrical components and find energy efficient designs. Therefore, knowing how to solve them fast and accurately for many design layouts becomes necessary. Although numerical methods to solve the heat equation have existed for decades, these are computationally expensive. Recent developments in the area of physics-informed neural networks (PINNs) offers an avenue for finding faster ways to solve this family of PDEs.

In this paper, based on the study by Zhao et. al ([1]), we attempt to find the steady-state solutions of the heat equation in a two-dimensional conducting domain with appropriate boundary conditions using data-driven (supervised) and unsupervised approaches. We compare the performance of the two methods with traditional numerical methods, namely the finite differences method (FDM). The unsupervised approach, which is the focus of this study uses a UNet based CNN and a physics-informed loss function to guide the CNN towards the temperature field solution, dicussed more in the Methods (III) section.

II. DATASET

For the temperature field prediction of heat source layout (HSL-TFP), the training of the unsupervised model is data-free, i.e. no labeled datasets are used. The physical knowledge is inbuilt into construction of the loss function which guides the network to the solution.

In order to evaluate the performance of the model use randomly generated heat source layout maps, Heat sources are placed on a square shaped domain of $0.1 \times 0.1m^2$ surrounded by adiabatic walls, with an isothermal sink in the middle of the left boundary and 12 rectangular heat sources placed randomly in the domain as showcased in Figure 1. We use 10000 maps of which 8000 are used for training, and the remaining are used for validation and testing. The solution from the CNN is compared against the FDM solution solved within the domain with appropriate boundary conditions. Two cases of layouts were studied in the project: case 1 includes 20 $0.01 \times 0.01m^2$, $10000w/m^2$ heat sources, and case 2 includes 12 heat sources of varied sizes and intensities.

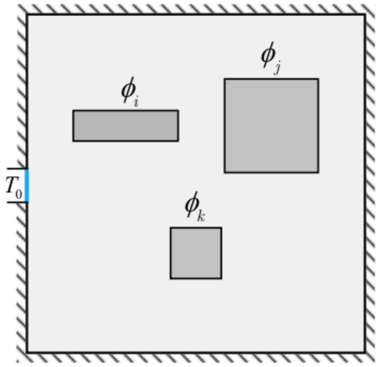


FIG. 1. Schematic of the square domain with several heat sources.

III. METHODS

A. Physics-informed Loss

1. Boundary Conditions

A Dirichlet boundary condition was imposed for $(x_i, y_i) \in D_{iso}$, where D_{iso} referred to isothermal sinks. The condition applied in the project was

[*] pknauss@ucsd.edu
[†] tsievert@ucsd.edu
[‡] wehu@ucsd.edu
[§] aaarora@ucsd.edu

$$T(x_i, y_i) = T_0 = 298K, (x_i, y_i) \in D_{iso} \tag{1}$$

A Neumann boundary condition was imposed for $(x_i, y_i) \in D_{adia}$ to reflect the adiabatic walls surrounding the domain, where $D_{adia} = \partial D \setminus D_{iso}$ included all points on the boundary excluding the isothermal sink.

$$\Phi(x_i, y_i) \cdot \hat{\mathbf{n}} = -\lambda \nabla T(x_i, y_i) \cdot \hat{\mathbf{n}} = 0, (x_i, y_i) \in D_{adia} \tag{2}$$

Where $\lambda = 1Wm^{-1}K^{-1}$ was the thermal conductivity set to the domain.

2. Physics Loss Function

In obtaining a temperature-field prediction, the project was essentially solving a boundary value problem with Poisson's equation as a prototype

$$\nabla(\lambda \nabla T(x, y)) = \phi(x, y) \tag{3}$$

Which followed the basic form of Poisson's Equation over a continuous domain. Here $\phi(x, y)$ was the correction due to the heat sources presented in the layout, equaled the intensity (in Wm^{-2}) of the heat source for $(x, y) \in D_{heatsource}$; $\phi(x, y) = 0$ for $(x, y) \in D \setminus D_{heatsource}$.

Solving the equation over a mesh, Poisson's equation was discretized using the finite difference method in 2D, which utilized five points (the target point and its four neighbors) to define an analog to the total derivative. WLOG, the second partial derivative along the x-axis was approximated from the Taylor expansion

$$\begin{aligned} &\frac{T(x_i + h, y_j) - 2T(x_i, y_j) + T(x_i - h, y_j)}{h^2} \\ &\qquad = \frac{\partial^2 T(x_i, y_j)}{\partial x^2} + O(h^2) \end{aligned} \tag{4}$$

Similarly for the second partial derivative along the y-axis. The combined discretized equation was

$$\begin{aligned} 4T(x_i, y_j) - T(x_i - h, y_j) - T(x_i + h, y_j) \\ -T(x_i, y_j - h) - T(x_i, y_j + h) = \frac{h^2 \phi(x_i, y_j)}{\lambda} \end{aligned} \tag{5}$$

Where uniform discretization over the two directions was applied. It was empirically found that the function was better trained with an intermediate variable defined

$$\begin{aligned} T'(x_i, y_j) = T(x_i - h, y_j) + T(x_i + h, y_j) \\ +T(x_i, y_j - h) + T(x_i, y_j + h) + \frac{h^2 \phi(x_i, y_j)}{\lambda} \end{aligned} \tag{6}$$

And thus apply the following matrix as the convolution kernel

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Overall, the physics loss was obtained as

$$\begin{aligned} \mathcal{L} = \frac{1}{|D_I \cup D_{adia}|} \sum_{(x,y)} ||T(x_i, y_j) - \frac{1}{4} T(x_i, y_j)'|| \\ (x_i, y_j) \in D_I \cup D_{adia} \end{aligned} \tag{7}$$

3. Loss Implementation

The Dirichlet BCs were enforced at the beginning of each iteration so that the solution on the isothermal sinks remained constant; the physics loss was thus defined over the interior of the domain plus the adiabatic walls only. The loss was also found to share essentially the same form as the update rule for Poisson's Equation using the Jacobi iteration method; a parallel interpretation of how our network was guided by the loss function via back propagation during training.

The optimization of the loss landscape was a challenge as in PINNs in general. An online hard example mining (OHEM) algorithm was applied to manipulate the weight across regions with different losses, improving the effectiveness of the optimizer over regions with large losses (the hard examples) during back propagation. The physics loss with OHEM incorporated was

$$\begin{aligned} \mathcal{L} = \frac{1}{|D_I \cup D_{adia}|} \sum_{(x,y)} w_{ij} ||T(x_i, y_j) - \frac{1}{4} T(x_i, y_j)'|| \\ (x_i, y_j) \in D_I \cup D_{adia} \end{aligned} \tag{8}$$

Where $w_{ij} = \eta_1 + \eta_2 \dfrac{\delta_{ij} - max(\delta)}{max(\delta) + min(\delta)}$ was the weight, and η_1, η_2 are hyperparameters for shifting and scaling; $\delta_{ij} = ||T(x_i, y_j) - T(x_i, y_j)'||$ was the error.

B. Neural Network Architecture

We used a CNN with U-Net architecture as it is able to properly see both local and global differences in temperature gradients, which influence the temperature at any given point.

Our U-Net has each block built off of one max pooling, then 2 sets of 2D convolutions, batch normalizations, then a GELU (Gaussian Error Linear Unit) activation unit. Each of these blocks allows us to see a subset of locality which is then transferred down the layers. Beyond these blocks, we have 4 layers of encoding, one central layer, and 4 layers of decoding. Based on the paper, we thought that this was a good middle ground between enough global temperature comparison and not too long to run the training.

We used the Adam optimizer with an Exponential Decay learning rate change with a gamma of 0.85. For our hyperparameters, we used a batch size of 1, with 4 parallel "workers" working at the same time.

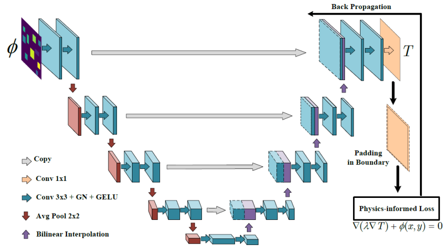


FIG. 2. U-Net pipeline showing the architecture of the CNN.

As this is not a classification problem, we can not use the normal measure of accuracy, so we will use the Mean Absolute Error of both the entire domain (MAE) and the components (CMAE) as well as the AE maximum (Max AE) and the maximum temperature difference across the domain (MT-AE). The paper was able to get the errors shown in Table 1, so we will attempt to get within 1.25x their errors for all metrics.

	MAE	CMAE	Max-AE	MT-AE
Case 1	0.0108	0.0109	0.0280	0.0140
Case 2	0.0275	0.0271	0.0925	0.0336

TABLE I. Paper Errors.

The code for our implementation can be found at `https://github.com/aaarora/hsl-tfp`

IV. RESULTS

Our unsupervised learning results are presented in Table 2 below.

	MAE	CMAE	Max-AE	MT-AE
Case 1	0.0108	0.0108	0.0269	0.0131
Case 2	0.0261	0.0262	0.0752	0.0314

TABLE II. Our Errors.

From this table, we can see that our losses matched or very slightly "beat" the errors presented in the paper. While this is surprising, it is not entirely confusing as we were starting with their work and building off of it while also attempting to get the lowest test error possible so we are able to limit our test errors without worrying about other metrics.

Figures 3 and 4 show heatmaps comparing our FDM prediction against our model's temperature field output for both Case 1 and Case 2 respectively. From these, we can see that our errors are relatively randomly distributed, but still well within our expected loss values.

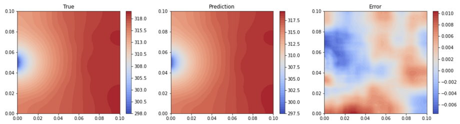


FIG. 3. Case 1 layout loss heatmap.

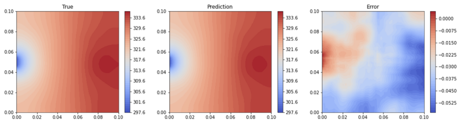


FIG. 4. Case 2 layout loss heatmap.

V. COMPARISONS

A. Comparison with data-driven approach

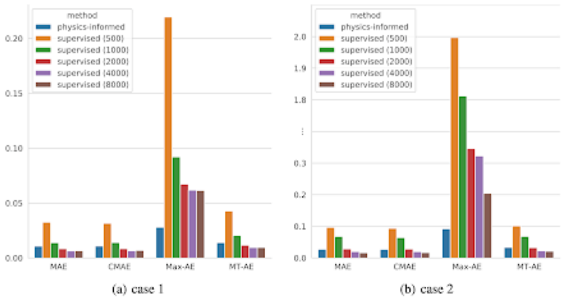
To better benchmark the performance of the unsupervised learning model, we also trained a supervised learning model for this problem. To do this, we computed the FDM prediction for a given input layout, and trained a CNN to match inputs to outputs. This is a classic application of CNNs and a traditional way of applying machine learning.

We ran the SL with a training dataset of 8000, and tested it on a further 1000 layouts. Results for the SL model are given in the table below.

	MAE	CMAE	Max-AE	MT-AE
Case 1	0.0058	0.0061	1.164	0.0096
Case 2	0.0179	0.0178	1.844	0.0268

TABLE III. Supervised learning errors.

For MAE, CMAE, and MT-AE the supervised model performs slightly better than the unsupervised one. This trend is shown more thoroughly in Fig. 5, where we can see the trade off between unsupervised and supervised superiority. This model trend agrees with the general trend anticipated for PINNs; PINNs perform best in regimes where there's less data, and traditional techniques perform best in regimes with lots of data.



(a) case 1

(b) case 2

FIG. 5. Bar chart showing the performance of unsupervised learning and supervised learning at various training dataset sizes. For this problem, unsupervised learning appears to be comparable to suepervised learning with a training dataset size of around 3k.

Interestingly, the performance of the supervised model is significantly worse than the unsupervised on the Max-AE metric. Max-AE is the maximum single-pixel difference between the true (FDM-predicted) tfp and the model-predicted tfp. Therefore, Max-AE encodes some information about the model's performance locally, while the other metrics are more global. It seems that although supervised learning can – with enough data – surpass physics-informed learning on tasks requiring only a global average level of correctness, however, supervised learning fails to learn much about the local structure.

Representative results for the supervised learning on a specific layout are shown in Fig. 6 and 7.

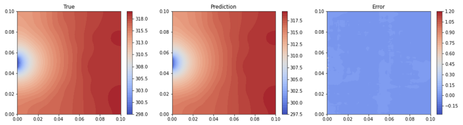


FIG. 6. Case 1 layout loss heatmap for SL.

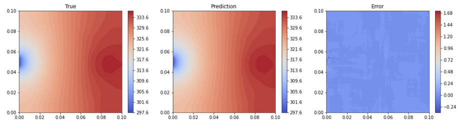


FIG. 7. Case 2 layout loss heatmap for SL.

The difference between the supervised and unsupervised models is seen most clearly by looking at the associated error heat maps. For supervised learning specifically, it seems to always predict a temperature of around 0.24K colder than the FDM prediction. We do not know why this is, but it is likely the main contributor to the poor performance on the Max-AE metric.

B. Comparison with Tensorflow implementation

The code used for all of the work discussed thus far was written in Pytorch. As a further extension of the paper – in addition to grounding this work in things we learned in class – we decided to attempt to port the unsupervised model to Tensorflow. In order to do this, we rewrote the custom model and loss function in tf.Keras. We wanted to also rewrite the dataloading in tf.Keras, but Pytorch's scheme for dataloading is complex and relies on many built-in functions. Thus, we tried to simply turn the loaded pytorch data in numpy arrays and then into tf tensors, but this failed due to the specifics of how the pytorch data is loaded.

As it stands, there is a "tensorflow" branch of the github that contains our work, however the tf version of the model does not run.

VI. CONCLUSION

In this paper we look into a UNet based convolutional neural network to solve for the steady-state solutions of the head equation in a two-dimensional conducting domain with appropriate boundary conditions. We use a physics-informed loss function to guide the CNN to map intensity distributions to solution functions without any labelled data and study to impose hard constraints on the Dirichlet and Neumann boundary conditions to speed up the convergence of the network. The performance we achieve using the CNN is comparable to the data-driven and numerical approach and only varies by 0.03 K across the domain. The supervised learning outperforms the PINN on global metrics with enough data, however the PINN is consistently better at learning local structure. Since the model can output the steady-state solutions without seeing the explicit solutions, it can be employed as a good alternative to solve this family of PDEs quickly. A similar approach can be used to train CNN to solve other PDEs in more complicated domains.

ACKNOWLEDGMENTS

We would like to extend our gratitude to the authors of the original paper, Zhao et. al, for their interesting study, and to Professor Javier Duarte for giving us the opportunity to work on this project. Also thanks to Dr. Amir Gholami for the special lecture on physics-informed neural networks which aided us in understanding the original work.

[1] X. Zhao, Z. Gong, Y. Zhang, W. Yao, and X. Chen, Physics-informed convolutional neural networks for temperature field prediction of heat source layout without la-

beled data, Engineering Applications of Artificial Intelligence **117**, 105516 (2023).