A Hands-on Introduction to Physics-informed Machine Learning

Ilias Bilionis, Atharva Hans, Predictive Science Laboratory School of Mechanical Engineering Purdue University May 26, 2021
Objective

Learn how physical information in the form of differential equations can be used to regularize neural networks.
Reminder - What are neural networks?

Inputs $x$ | Outputs $y$

Affine transformations followed by non-linearities

Parameters: $\theta$

Function: $y = N(x; \theta)$
Reminder - How do we train neural networks?

• Depends on what the task is... Focusing on regression.

• You have some data consisting of inputs $x_{1:n} = (x_1, \ldots, x_n)$ and outputs $y_{1:n} = (y_1, \ldots, y_n)$.

• Say you want to find a neural net $N(x; \theta)$ that goes from input to output.

• Minimize a loss function:

$$L(\theta) = \frac{1}{n} \sum_{i=1}^{n} \left[ y_i - N(x_i; \theta) \right]^2$$
Reminder - How do we train neural networks?

- Minimize a loss function:
  \[ L(\theta) = \frac{1}{n} \sum_{i=1}^{n} \left[ y_i - N(x_i; \theta) \right]^2 \]

- We typically use a form of **stochastic gradient descent**

\[
\theta_{t+1} = \theta_t - a_t \frac{1}{m} \sum_{j=1}^{m} \nabla_{\theta} \left[ y_{ij} - N(x_i; \theta_t) \right]^2
\]

Learning rate which has to satisfy certain constraints (Robbins-Monro, 1951)

Randomly sampled batch of inputs/outputs

Automatic differentiation for getting gradients (PyTorch, TensorFlow)
Illustrative Example 1: Solving an ODE

From ODE to a loss function

• Consider the initial value problem:

\[ \frac{d\Psi}{dx} = f(x, \Psi) \]

\[ \Psi(0) = A \]

• Automatically satisfy the initial condition by parameterizing the solution as:

\[ \hat{\Psi}(x; \theta) = A + xN(x; \theta) \]

• The idea is to find \( \theta \) by minimizing the *integrated squared residual* of the ODE:

\[ L(\theta) = \int_0^1 \left( \frac{d\hat{\Psi}(x; \theta)}{dx} - f(x, \hat{\Psi}(x; \theta)) \right)^2 dx \]
Solving the problem with stochastic gradient descent

• The idea is to find $\theta$ by minimizing the integrated squared residual of the ODE:

$$L(\theta) = \int_0^1 \left[ \frac{d\hat{\Psi}(x; \theta)}{dx} - f(x, \hat{\Psi}(x; \theta)) \right]^2 dx$$

• The following algorithm converges (Robbins-Monro, 1951)

$$\theta_{t+1} = \theta_t - \frac{a_t}{n} \sum_{i=1}^n \nabla_{\theta} \left[ \frac{d\hat{\Psi}(x_i; \theta_t)}{dx} - f(x_i, \hat{\Psi}(x_i; \theta_t)) \right]^2$$

Spatial locations uniformly sampled in $[0, 1]$ at each iteration.
$\frac{d\Psi(x)}{dt} = \exp\left\{-\frac{x}{5}\right\} \cos x - \frac{\Psi(x)}{5}$

$\Psi(0) = 0$
Illustrative Example 2: Solving an elliptic PDE
From PDEs to a loss function -
Integrated squared approach

\[-\nabla \cdot [a(x) \nabla u(x)] + c(x)u(x) = f(x)\]

\[L(\theta) = \int \left\{ \nabla \cdot [a(x) \nabla \hat{u}(x; \theta)] + c(x)\hat{u}(x; \theta) + f(x) \right\}^2 dx\]

From PDEs to a loss function - Energy approach

\[- \nabla \cdot \left[ a(x) \nabla u(x) \right] + c(x)u(x) = f(x)\]

\[L(\theta) = \int \left\{ \frac{1}{2} a(x) \nabla \hat{u}(x; \theta) + c(x)\hat{u}^2(x; \theta) - f(x)\hat{u}(x; \theta) \right\} dx - \int_{\Gamma_N} g_N\hat{u}(x; \theta) d\Gamma_N\]

S. Karumuri, R. Tripathy, I. Bilionis, Simulator-free Solution of High-dimensional Elliptic Partial Differential Equations using Deep Neural Networks, 2020
I can already solve ODEs/PDEs. Why is this useful?
Illustrative Example 3: Solving PDEs for all possible parameterizations

\[-\nabla \cdot \left[ a(x; \xi) \nabla u(x; \xi) \right] + c(x)u(x; \xi) = f(x)\]

\[u = 0, \forall x_1 = 1,\]

\[u = 1, \forall x_1 = 0,\]

\[\frac{\partial u}{\partial n} = 0, \forall x_2 = 1.\]

\[\log a(x, \xi) = \begin{array}{c}
\text{or} \\
\text{or} \\
\text{or} \\
\text{...}
\end{array}\]
Representing the solution of the PDE with a DNN

$$u(x, \xi)$$

Input

Hidden layers

Output

$${L_1} \rightarrow {L_2} \rightarrow {L_3} \rightarrow {L_4}$$
From PDEs to a loss function - Energy approach

\[-\nabla \cdot \left[ a(x; \xi) \nabla u(x; \xi) \right] + c(x)u(x; \xi) = f(x)\]

\[L(\theta) = \mathbb{E}_{\xi} \left[ \int \left\{ \frac{1}{2} a(x, \xi) \nabla \hat{u}(x, \xi; \theta) + c(x)\hat{u}^2(x, \xi; \theta) - f(x)\hat{u}(x, \xi; \theta) \right\} \, dx - \int_{\Gamma_N} g_N \hat{u}(x, \xi; \theta) \, d\Gamma_N \right]\]

S. Karumuri, R. Tripathy, I. Bilionis, Simulator-free Solution of High-dimensional Elliptic Partial Differential Equations using Deep Neural Networks, 2020
One network for all kinds of random fields

Figure 7: (2D SBVP - GRF \( \times [0.05, 0.08] \)). Each row corresponds to a randomly chosen realization of log-input field (left column) from the GRF of length-scales \([0.05, 0.08]\) test dataset and the corresponding solution response from FVM and DNN (middle and right columns).

Figure 8: (2D SBVP - Warped GRF). Each row corresponds to a randomly chosen realization of log-input field (left column) from the warped GRF test dataset and the corresponding solution response from FVM and DNN (middle and right columns).

One network for all kinds of random fields

Figure 9: (2D SBVP - Channelized field) Each row corresponds to a randomly chosen realization of input field (left column) from the channelized field test dataset and the corresponding solution response from FVM and DNN (middle and right columns).

Figure 10: (2D SBVP - Multiple length-scales GRF) Each row corresponds to a randomly chosen realization of log-input field (left column) from the multiple length-scales GRF test dataset and the corresponding solution response from FVM and DNN (middle and right columns).

What are the applications of this?

- High-dimensional uncertainty propagation through PDEs.
- Solving free boundary and Stefan problems (Wang, Perdikaris, 2021).
- PDE-constrained optimization (Hennigh et al., 2020).
- Inverse/model calibration problems (Raise, 2019).
- Data assimilation/filtering (no one yet).
- ...
What is the catch?

• Not as easy as it looks in practice…

• Vanishing gradients…

• Spectral bias of deep nets…

• Fine solution features…

• We address some of these in the hands-on activity by solving the physical equations for a compressible neo-Hookean material.
Hands-on activity led by Atharva Hans

https://nanohub.org/tools/handsonpinns