Addressing computational challenges in physical system simulations with machine learning

Sabber Ahamed

April 25, 2023

Abstract

We introduce a new machine learning framework designed to help researchers who relys on simulations to study physical systems or processes. High computational costs and the resulting limited data often pose significant challenges to gaining insights into these systems or processes. In this innovative approach, we first train a supervised predictive model using a small-scale simulated dataset with varying input parameters. Subsequently, a deep learning generator is developed using the same data points. The final step involves training a reinforcement learning agent that employs the supervised model and generator to produce more accurate, simulation-like data. By leveraging this framework, researchers can generate more accurate data and know the outcomes without running high computational simulations. Which enabls them to explore the parameter space more efficiently and gain deeper insights into physical systems or processes. In this paper, we demonstrate the effectiveness of the proposed framework by applying it to two case studies in material science and geodynamics. We also discuss the potential applications of this framework in other fields, such as fluid dynamics, climate modeling, and drug discovery.

1. First problem is computational cost hence we can not produce enough simulated data to learn the system or process.
2. Second, we do not know the parameter space outside the train distribution.
3. Therefore, We can not explore the parameter space efficiently and gain deeper insights into physical systems or processes.

1 Introduction

The proposed framework has numerous potential applications across various fields. In material science, it can optimize processing conditions for developing high-performance materials, such as aerospace alloys, by considering factors such as temperature, pressure, and time. In geodynamics, the framework can help researchers understand the complex interplay between mantle convection, tectonic plate motion, and crustal deformation, enabling more accurate modeling and prediction of geodynamic processes.

Furthermore, the framework can be extended to other domains, such as fluid dynamics, climate modeling, and drug discovery, where simulations are crucial for understanding the underlying systems and optimizing their performance. By leveraging the power of machine learning, this framework offers a versatile and efficient solution for researchers to navigate complex parameter spaces and advance their understanding of various physical systems.

2 Generator Framework

The proposed machine learning framework consists of three main components: (1) a supervised predictive model, (2) a deep learning generator, and (3) a reinforcement learning agent. The framework is consists of these interdependent components that work together to generate more accurate like data and predict the outcomes of simulations.

2.1 Overview

First, we generate a small-scale simulated dataset with varying input parameters (X_i). We then train a supervised predictive model, $f(X)$, using this simulated data (X_i) to predict the simulation outcomes (y_i). Following this, we train a deep learning generator, $g(Z)$, using the same simulated data points (X_i) and a noise vector Z sampled from a noise distribution $P_z(Z)$. Lastly, we train a reinforcement learning agent with policy $\pi_\theta(a|s)$ using the supervised model and the generator to produce more accurate, simulation-like data. In this case, a represents the action (input parameter combination) and s represents the state (predicted outcome). The agent's objective is to learn a policy that maximizes the outcome predicted by the supervised model. In the following subsections, we will describe each component, along with the underlying models and methods.

2.2 Supervised Model: predict the outcome of simulations

The supervised predictive model aims to predict the outcome of a simulation based on the input parameters. Let X represent the input parameters and Y the corresponding outcome. We can represent the supervised predictive model as a function $f(X)$, where $f(X)$ approximates the true relationship between X and Y. We use a dataset of N simulated data points, $\{(x_i, y_i)\}_{i=1}^{N}$, to train the model.

2.3 Generator: generate simulation-like data

The deep learning generator aims to create simulation-like data based on the input data. The generator, represented as a function $g(Z)$, takes a noise vector Z as input and generates data points similar to the input data.

For example, we can use a Generative Adversarial Network (GAN) with generator $g(Z)$ and discriminator $D(X)$. The generator and discriminator are trained simultaneously with the following objectives:

$$\min_G \max_D \mathbb{E}_{x\sim P_{\text{data}}(X)}[\log D(x)] + \mathbb{E}_{z\sim P_z(Z)}[\log(1 - D(g(z)))], \tag{1}$$

where $P_{\text{data}}(X)$ is the true data distribution and $P_z(Z)$ is the noise distribution.

2.4 Agent: optimize generator to maximize the outcome

The reinforcement learning agent aims to develop a policy for generating input parameter combinations that maximize the outcome predicted by the supervised model. The agent interacts with the generator and the supervised model to obtain rewards, which are used to update its policy.

For example, we can use a Proximal Policy Optimization (PPO) algorithm. The agent's policy, denoted as $\pi_\theta(a|s)$, maps states s to actions a using parameters θ. The agent's objective is to maximize the expected return:

$$\max_\theta \mathbb{E}s \sim P\theta_{\text{old}}, a \sim \pi_{\theta_{\text{old}}} \left[\frac{\pi_\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)} A^{\pi_{\theta_{\text{old}}}}(s,a) \right], \tag{2}$$

where $A^{\pi_{\theta_{\text{old}}}}(s,a)$ is the advantage function, which estimates the relative value of taking action a in state s under the policy $\pi_{\theta_{\text{old}}}$. The advantage function can be approximated using the Generalized Advantage Estimation (GAE) method:

$$A^{\pi_{\theta_{\text{old}}}}(s_t,a_t) = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_t^{(l)}, \tag{3}$$

where $\delta_t^{(l)} = r_t + \gamma V^{\pi_{\theta_{\text{old}}}}(s_{t+1}) - V^{\pi_{\theta_{\text{old}}}}(s_t)$, γ is the discount factor, and λ is a hyperparameter that controls the trade-off between bias and variance in the advantage estimation.

The agent updates its policy using the PPO algorithm, which employs a trust region optimization method with a clipped objective function to ensure stable updates:

$$L(\theta) = \mathbb{E}s, a \left[\min\left(\frac{\pi\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)} A^{\pi_{\theta_{\text{old}}}}(s,a), \text{clip}\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)}, 1-\epsilon, 1+\epsilon \right) A^{\pi_{\theta_{\text{old}}}}(s,a) \right) \right], \tag{4}$$

where ϵ is a hyperparameter that controls the size of the trust region.

Algorithm 1 Optimizing Simulations using the Machine Learning Framework

```
1: Train the supervised predictive model $f(X)$ using simulated data $(x_i, y_i)i = 1^N$
2: Train the deep learning generator $g(Z)$ using the same simulated data points $X$. Where Z is a noise vector sampled from a noise distribution $P_z(Z)$
3: Initialize the reinforcement learning agent's policy $\pi\theta(a|s)$. Where $a$ is the action and $s$ is the state. Action $a$ is the input parameter combination and state $s$ is the predicted outcome
4: for each iteration do
5:     Generate noise vector $Z$ from the noise distribution $P_z(Z)$
6:     Generate simulation-like data $g(Z)$ using the generator
7:     Obtain the predicted outcome $f(g(Z))$ using the supervised model
8:     Use the reinforcement learning agent to interact with the generator and the supervised model, obtaining rewards based on the predicted outcomes
9:     Update the agent's policy $\pi_\theta(a|s)$ using the PPO algorithm
10: end for
11: return optimized input parameter combinations and predicted outcomes
```

3 Case Studies

3.1 Material Science

1. Application of the framework to specific fields (e.g., material science, geodynamics)

2. Description of the problem, input parameters, and expected outcomes

3. Results and insights gained from applying the framework

3.2 Earthquake Rupture Physics

4 Discussion and Future Work

5 Conclusion

1. Summary of the machine learning framework and its advantages

2. Potential impact on the study of physical systems or processes

3. Future work and extensions of the framework