Search Competitions

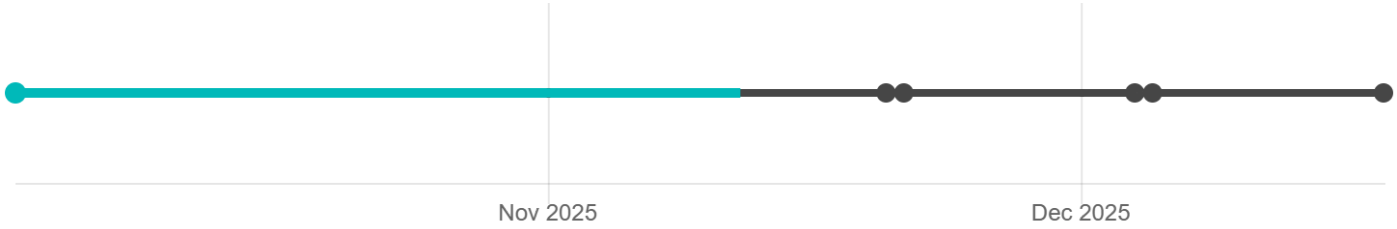
Benchmarks/Competitions Datasets oskar_uibk25

ML FOR PHYSICS - UIBK W25

33 PARTICIPANTS

7 SUBMISSIONS

ORGANIZED BY: Gorkamunoz
CURRENT PHASE ENDS: 20. November 2025 Kl. 01:00 CET
CURRENT SERVER TIME: 11. November 2025 Kl. 12:14 CET
Docker image: codalab/codalab-legacy:gpu



Get Started | Phases | My Submissions | Results | Forum

- Overview
- Data
- Code evaluation
- Terms
- Files

Task 1 - Classification of Ising configurations

The task here is predict the phase of an Ising configuration. We consider here a 2D square lattice of M x M spins under periodic boundary conditions and the Hamiltonian:

$$H = J \sum_{\langle ij \rangle} \sigma_i \sigma_j,$$

where `< ij >` refers to nearest-neighbors. As you may know from your condensed matter courses, this system exhibits a phase transition at the critical temperature

$$T_c = \frac{2J}{k \ln(1+\sqrt{2})}.$$

Your goal is to predict the phase of a given M x M configuartion. We will consider that

$$\text{label}_i = \begin{cases} 0 & \text{if } T_i < T_c \\ 1 & \text{if } T_i \geq T_c \end{cases}$$

Data format

The Public Data (Get Started > Files) consists of two files:

- `classification_input.pt`: this `torch` file contains spin configurations in a tensor of shape `N x M x M`, where N is the number of configurations and M x M the number of spins.
- `classification_true.pt`: this `torch` file contains the labels (phase) in a tensor of shape `N`.

Your submission code should expect an input of same as `classification_input.pt`, although note that N will change.

Metric

As a classification problem, your predictions will be evaluated with the Binary Cross Entropy loss

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{N}\sum_{i=1}^{N} \left[y_i \log(p_i) + (1-y_i) \log\left(1-p_i\right) \right],$$

where y_i is the groundtrouth phase and p_i your predicition (either 0 or 1).

Baseline

The magnetization is the order parameter of the 2D Ising model and hence can be used to classify the phase of a given configuration. We will hence use this as a baseline and aim to improve such estimator.

Task 2 - Regression of the anomalous exponent of stochastic trajectories

The task here is to predict the anomalous exponent &alpha of diffusing trajectories generated via Fractional Brownian motion. Anomalous diffusion is connected to non-equilibrium phenomena, flows of energy and information, and transport in living systems. Typically, anomalous diffusion is characterized by a nonlinear growth of the mean squared displacement MSD with respect to time t:

$$\mathrm{MSD} \propto t^{\alpha}$$

where α ∈ [0, 2]. Your goal in this task is to train a model that gives the best estimate of α for each input trajectory. I you want to know more about this problem, the regression of α was one of the main topics of the AnDi Challenge. You can take a look at the paper to also see how different teams developed the best possible methods to characterize α.

Data format

The Public Data (Get Started > Files) consists of two files:

- `regression_input.pt`: this `torch` file contains the trajectories in a tensor of shape `N x T`, where N is the number of trajectories and T is their length.
- `regression_true.pt`: this `torch` file contains the labels (anomalous exponent &alpha) in a tensor of shape `N`.

Your submission code should expect an input of same as `regression_input.pt`, although note that N will change

Metric

Your predictions will be evaluated by means of the mean squared error (MSE):

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\mathrm{pred}_i - \mathrm{true}_i)^2$$

Baseline

ML is not the only way to compute the anomalous exponent, although as shown in the link above, it is actually the current best estimator. Before ML, the most typical method was to fit the MSD in logarithmic scale (see equation above).
The baseline for this problem will be given by such method. If you want to explore how good it performs you can use the `andi_datasets` library:

```
from andi_datasets.analysis import msd_analysis

preds = msd_analysis().get_exponent(trajs[:,:,torch.newaxis])  # Expects input N x T x dimension, hence the
```

Hint: it is quite frequent that the models overfit to predicting all ones (the average exponent over the dataset). This gives rise to an MSD ~ 1/3.

Task 3 - Reinforcement Learning

TBD