Exercise Sheet #5

Due: 27.11.2023 11h00

Jan Pawlowski, Tilman Plehn
Jonas Spinner

General Regulations.

- Please hand in your solutions in groups of three people. A mix of attendees from different tutorials is fine. We will not correct submissions from single students.
- Your solutions to theoretical exercises can be either handwritten notes (scanned to `pdf`), typeset using LaTeX, or directly in the jupyter notebook using Markdown.
- For the practical exercises, the data and a skeleton for your jupyter notebook are available at https://github.com/heidelberg-hepml/mlph2023-Exercises. Always provide the (commented) python code as well as the output, and don't forget to explain/interpret the latter, we do not give points for code that does not run. Please hand in both the notebook (`.ipynb`) and an exported pdf. Combine your the pdfs from theoretical and notebook exercises into a single pdf.
- Submit all your files in the Übungsgruppenverwaltung, only once for your group of three. Please list all names and tutorial numbers to simplify things for us.

1 Loss functions

(a) Neural network classifiers are usually trained on the binary cross-entropy (BCE) loss

$$\mathcal{L} = \Big\langle -\log D(x) \Big\rangle_{x\sim p_1} + \Big\langle -\log(1-D(x)) \Big\rangle_{x\sim p_2}, \tag{1}$$

where p_1, p_2 are the two distributions to be classified and $D(x) \in [0,1]$ is the output of the neural network. Rewrite the averages as integrals and minimize the BCE loss with respect to D to derive an expression for the output $D(x)$ of a perfectly trained network depending on p_1, p_2. Use this result to express the likelihood ratio $t_{\text{LR}}(x) = p_1(x)/p_2(x)$ in terms of $D(x)$. (2 pts)

(b) Generative neural networks are trained to approximate a distribution p_{train} with another distribution p_{model} encoded in the network, where the distribution p_{train} is only implicitly available through samples $x \sim p_{\text{train}}$. Most types of generative neural networks are trained on the log-likelihood loss

$$\mathcal{L} = \Big\langle -\log p_{\text{model}}(x) \Big\rangle_{x\sim p_{\text{train}}}. \tag{2}$$

Verify that this loss has the unique minimum $p_{\text{model}} = p_{\text{train}}$. Hint: Enforce the normalization of p_{model} with Lagrange multipliers. (2 pts)

2 Phase Classification in ϕ^4 theory

In this exercise we study the phase transition of a scalar field ϕ on a lattice. The action is given by

$$S = \sum_{x\in\Lambda} \Big[-2\kappa \sum_{\mu=1}^{d} \phi(x)\phi(x+\hat{\mu}) + (1-2\lambda)\phi(x)^2 + \lambda\phi(x)^4 \Big], \tag{3}$$

where Λ denotes the set of all lattice sites, $\hat{\mu}$ is the unit vector in direction μ, $\kappa > 0$ is the Hopping parameter, and $\lambda > 0$ parametrizes the strength of the quartic interaction. The phase transition can be parametrized by the parameter κ, with κ^* being the critical value where the phase transition happens.

(a) Load the training and testing data, which consist of quadratic lattices sites with 16 edges. The training dataset consists of two sets of 1000 lattice sites in the unbroken ($\kappa = 0.24$) and broken phase ($\kappa = 0.3$). The testing dataset has sets of 100 lattice sites for values ranging from $\kappa = 0.24$ to $\kappa = 0.3$ in equidistant steps. Visualize lattice sites for different values of κ. Can you see the phase transition? (1 pts)

(b) Design a Convolutional Neural Network (CNN) that takes lattice sites as input and predicts the probability of the lattice being in the broken phase. Why would you choose a CNN for this task? Explain your choice for the kernel size. (2 pts)

(c) Train your CNN on the training dataset using a BCE loss. (1 pts)

(d) Evaluate your network on the testing dataset. Compute mean and standard deviation of lattices with same κ and use this to infer the critical value κ^*. Hint: Lecture (2 pts)

3 Uncertainties in Amplitude Regression

(a) A straight-forward method to quantify uncertainties from the training process is to retrain the network several times with different initializations. Train an ensemble of 10 amplitude regression networks as in sheet 4, exercise 2d. Then evaluate each network on the test dataset, and take mean and standard deviation over the ensemble to estimate mean and uncertainty of the prediction for each event. Being encoded in the differences of the 10 sets of predictions, we can study these uncertainties at different levels along our workflow. Hint: You can find the solutions for sheet 4, exercise 2 on github.

(i) Event-level: Compare the predicted uncertainty band $[-\sigma_{\log A}, \sigma_{\log A}]$ with the observed deviations $\log A_{\text{pred}} - \log A_{\text{true}}$. Does the band cover the deviations?

(ii) Histogram-level: Create a histogram in $\log A$ for the predictions of each network in the ensemble, then take mean and standard deviation over the bin heights. Compare this effective ensemble uncertainty to the statistical uncertainty. Hint: Histogram counts are Poisson distributed, i.e. the statistical uncertainty on a bin with N counts is $\sqrt{N}$. (2 pts)

A more sophisticated method of tracking these uncertainties is to use Bayesian Neural Networks (BNNs). They replace the deterministic values of the learnable weights θ in classical neural networks with a learnable distribution $q(\theta|x_{\text{train}})$, which is usually chosen to be an uncorrelated multi-dimensional gaussian distribution. A BNN is trained to minimize the difference between the approximate posterior $q(\theta|x_{\text{train}})$ and the true posterior $p(\theta|x_{\text{train}})$, quantified by the KL divergence of the two distributions. At evaluation time, one samples several neural networks θ_i from the posterior distribution $q(\theta|x_{\text{train}})$.

(b) Bayesian Neural Networks are trained on the ELBO objective. Derive the ELBO loss function starting from the identity

$$D_{\text{KL}}\Big[q(\theta|x_{\text{train}}), p(\theta|x_{\text{train}})\Big] \geq 0, \tag{4}$$

where $p(\theta|x_{\text{train}})$ is the true posterior and $q(\theta|x_{\text{train}})$ is the approximate posterior parametrized by a neural network. What does ELBO stand for? Simplify your loss function assuming that $q(\theta)$ factorizes into uncorrelated one-dimensional gaussians. Which of the two terms in your loss function is dominating in the limits of zero and infinitely many bayesian parameters θ? Hint: Sheet 1, exercise 1. (2 pts)

(c) Explain how you would implement, train and evaluate a Bayesian Neural Network in practice. Compare to the deterministic network from part (a). Discuss advantages and disadvantages of training BNNs compared to training ensembles of deterministic networks. (1 pts)

(d) Implement and train a Bayesian Neural Network on the Amplitude Regression dataset of sheet 4. Visualize your predictions for $\log A$ on one sample θ_i from the BNN posterior $q(\theta|x_{\text{train}})$. (2 pts)

(e) Sample 30 deterministic networks θ_i from the learned posterior $q(\theta|x_{\text{train}})$ and evaluate them on the test dataset. Repeat the steps in part (a) to compare the BNN uncertainties with the ensemble. (1 pts)