Fast Detection of Phase Transitions with Multi-Task Learning-by-Confusion

Julian Arnold
Department of Physics
University of Basel
Klingelbergstrasse 82, 4056 Basel, Switzerland
`julian.arnold@unibas.ch`

Frank Schäfer
CSAIL
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139, USA
`franksch@mit.edu`

Niels Lörch
Department of Physics
University of Basel
Klingelbergstrasse 82, 4056 Basel, Switzerland
`niels.loerch@unibas.ch`

Abstract

Machine learning has been successfully used to study phase transitions. One of the most popular approaches to identifying critical points from data without prior knowledge of the underlying phases is the *learning-by-confusion* scheme. As input, it requires system samples drawn from a grid of the parameter whose change is associated with potential phase transitions. Up to now, the scheme required training a distinct binary classifier for each possible splitting of the grid into two sides, resulting in a computational cost that scales linearly with the number of grid points. In this work, we propose and showcase an alternative implementation that only requires the training of a *single* multi-class classifier. Ideally, such multi-task learning eliminates the scaling with respect to the number of grid points. In applications to the Ising model and an image dataset generated with Stable Diffusion, we find significant speedups that closely correspond to the ideal case, with only minor deviations.

1 Introduction

An exciting application of machine learning in physics is the detection of phase transitions from data [1–11]: the state of a physical system is sampled at different values of a tuning parameter that determines the distribution of samples. The states are fed to an algorithm that identifies the parameter values at which the system undergoes a phase transition.

One of the most popular methods to accomplish this task is *learning-by-confusion* [3], which has successfully revealed phase transitions in a large variety of physical systems using data from both simulation [3, 12–26] and experiment [19]. So far, its implementation has been computationally costly, as it involves training K distinct binary classifiers and analyzing their accuracy, where $K + 1$ is the number of sampled values of the tuning parameter.

In this work, we propose an alternative implementation of the learning-by-confusion scheme that works by training a *single* K-class classifier, which thus promises a speedup by a factor of K in the ideal case. We demonstrate a significant speedup in two applications: First, the thermal phase transition of the Ising model, which is theoretically well-understood and allows for a comparison

with established results. Second, an image dataset generated with Stable Diffusion [27], which serves as a more challenging example where no prior knowledge of transitions is available.

2 Unbiased Learning-by-Confusion with Multi-Task Learning

In the simplest use case of the learning-by-confusion method [3], a physical system undergoes a phase transition as a function of a single real-valued parameter θ. To detect the critical point θ^* at which the phase transition occurs, the θ-axis is discretized into $K + 1$ different points and at each point, M samples are drawn from the system. With $K + 1$ points, there are K possibilities to separate the θ-axis in two non-empty, contiguous regions $\Theta_k^< = \{\theta | \theta < \theta_k^*\}$ and $\Theta_k^> = \{\theta | \theta > \theta_k^*\}$, each corresponding to a tentative location $\theta_k^* = (\theta_k + \theta_{k+1})/2$ of the phase transition that lies between the grid points θ_k and θ_{k+1}. For each of these splittings, a separate classifier is trained to distinguish the two corresponding classes of samples, see Fig. 1(a). Intuitively, whichever classifier k is least confused, i.e., achieves the lowest error rate on evaluation, must have been trained on the most natural splitting of the data. Therefore, the value θ_k^* associated with the lowest error rate is our best guess for the location of the phase transition.

The loss function of the kth classifier is an unbiased binary cross-entropy loss

$$\mathcal{L}_k = -\frac{1}{2} \sum_{y \in \{>,<\}} \frac{1}{|\mathcal{D}_k^y|} \sum_{\boldsymbol{x} \in \mathcal{D}_k^y} \log\left[\hat{p}_k(y|\boldsymbol{x})\right], \tag{1}$$

where $|\mathcal{D}_k^y|$ is the size of the dataset $\mathcal{D}_k^y$ of samples $\boldsymbol{x}$ corresponding to Θ_k^y, and $\hat{p}_k$ is the estimated class probability of the classifier. The normalization $\frac{1}{|\mathcal{D}^y|}$ compensates for the imbalance of classes that would otherwise bias the signal and may lead to a failure of the confusion scheme [26]. Similarly, the error rate of the kth classifier can be estimated as

$$p_k^{\mathrm{err}} \approx \frac{1}{2} \sum_{y \in \{>,<\}} \frac{1}{|\mathcal{D}_k^y|} \sum_{\boldsymbol{x} \in \mathcal{D}_k^y} \mathrm{err}_k(y, \boldsymbol{x}), \tag{2}$$

where for each sample $\boldsymbol{x}$ the error $\mathrm{err}_k(y, \boldsymbol{x})$ is 0 if it is classified correctly and 1 if it is classified erroneously. During training, $\mathcal{D}_k^y$ in Eq. (1) refers to a training set, while it typically refers to a separate evaluation set when estimating the error via Eq. (2).
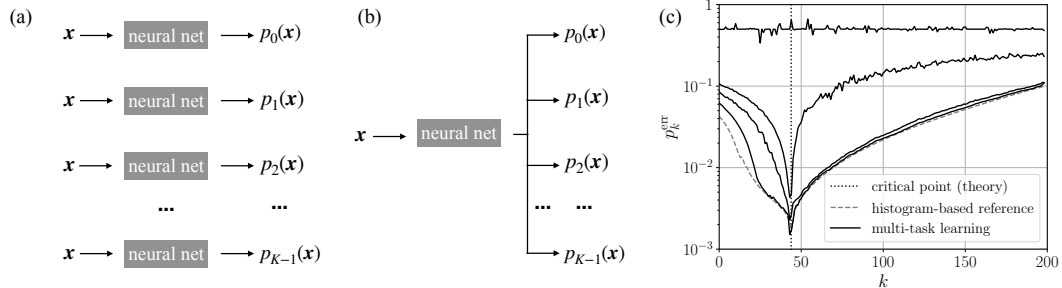

Figure 1: Schematic illustration of the learning-by-confusion method for detecting phase transitions (a) with original single-task architecture and (b) with our proposed multi-task architecture. (c) Classification error for the Ising model at each node using a single network trained on spin configurations. The solid lines correspond to, from top to bottom, the result after 0, 1, 5, and 50 epochs of training averaged over 5 independent runs. The vertical dotted line indicates the true location of the phase transition and the dashed line corresponds to an estimate of the Bayes-optimal error rate obtained using a histogram-based generative classifier in energy space [20, 26].

Instead of training a new classifier for each tentative splitting k, we propose to train a single classifier with K outputs $\{\hat{p}_k\}_{k=0}^{K-1}$, cf. Figs. 1(a) and (b). The loss function for this multi-task architecture is $\mathcal{L} = \frac{1}{K} \sum_{k=0}^{K-1} \mathcal{L}_k$. The evaluation of the error rate remains the same as before. Multi-task learning [28, 29] is expected to be highly efficient because the K classification tasks are very similar to each other and only differ in a slight alteration of the tentative splitting of the parameter space. Thus, the learned features are very much transferable between tasks.

3 Benchmark and Application

3.1 Ising Model

As a test system that has been extensively analyzed theoretically and for which established benchmarks are available, we consider the two-dimensional square-lattice ferromagnetic Ising model. It is described by the energy function $E(\boldsymbol{\sigma}) = -J \sum_{\langle ij \rangle} \sigma_i \sigma_j$, where the sum runs over all nearest-neighboring sites (with periodic boundary conditions), J is the interaction strength ($J > 0$), and $\sigma_i \in \{+1, -1\}$ denotes the discrete spin variable at lattice site i. The Ising model exhibits a phase transition between a paramagnetic phase at high temperature T and a ferromagnetic phase at low temperature [30].

To generate the dataset for the Ising model, we sample spin configurations $\boldsymbol{\sigma}$ on a 60×60 lattice from Boltzmann distributions at 200 equally-spaced parameter values between $0.05 \, k_{\mathrm{B}}T/J$ and $10 \, k_{\mathrm{B}}T/J$ via Markov chain Monte Carlo, see Appendix A.1.2 for details. Figure 1(c) shows how the learning-by-confusion signal of a multi-task convolutional neural network trained on this dataset evolves with training epochs, see Appendix A.1.2 for implementation details. Eventually, the node achieving the lowest error rate coincides with the critical point.

To compare the single- and multi-task approach, we train 4-layer convolutional networks with otherwise identical architectures and training settings. Figure 2(a) shows how the error rate evolves as a function of the training epoch at nodes below, near, and above the critical point. In all cases, single-task learning-by-confusion shows a slightly faster rate of convergence early on during training. Below the phase transition, multi-task learning-by-confusion does not achieve an error rate as low as single-task learning-by-confusion. In contrast, near and above the phase transition, multi-task learning-by-confusion eventually catches up and even achieves lower error rates.

We also studied the training behavior of multi-task networks that each have an output node corresponding to the true critical point as well as a varying number of additional output nodes. In particular, we recorded the number of epochs it takes to reach different error thresholds at their critical output node. At low thresholds, the difference in the number of epochs between different architectures was negligible. At high thresholds, we observed the number of epochs to marginally increase with the number of output nodes. However, we observed no clear scaling and any overhead was minor (at worst $\approx \times 6$ for some combinations of training hyperparameters and model architectures) as compared to the speedup gained through multi-tasking.
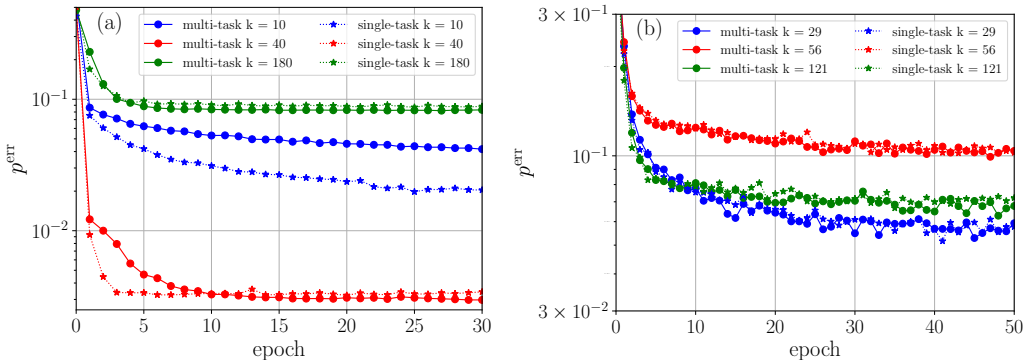

Figure 2: Error rate at representative nodes as a function of training epoch for single-task and multi-task learning-by-confusion for (a) the Ising dataset averaged over 5 runs and (b) the Stable Diffusion dataset averaged over 4 runs. Error bars derived from the standard deviation are negligible (same scale as markers), and underestimate the true confidence intervals, e.g., due to the non-Gaussian nature of the data.

3.2 Stable Diffusion

We now consider an image dataset generated using Stable Diffusion [27], where for each integer θ in $[1900, 2050)$ images are sampled with the prompt "technology of the year θ". For example, node 0 corresponds to the separator between the years 1900 and 1901. While this dataset does not

feature phase transitions in the physical sense, the learning-by-confusion scheme can be used to identify points in parameter space at which the data distribution changes rapidly. Note that no prior benchmark is available for this data and the predicted change points cannot be verified by theory. As the images have complex features comparable to typical image datasets, we load the pretrained ResNet-50 [31] from `PyTorch` [32], and exchange its final layer to fit our task, see Appendix A.1.1 for implementation details.

Figure 3 shows (at least) three major local minima indicating rapid changes in the image dataset; one between the years 1929 and 1930, a second, broader one in the 1990s, and a third one between 2021 and 2022. The Stable Diffusion model has only encountered images of actual technology from before and around 2022 in its training dataset LAION-5B [33], which may explain the third minimum. Due to the small dataset size and the resulting challenges in generalization, the signal is generally less reliable close to the edges.
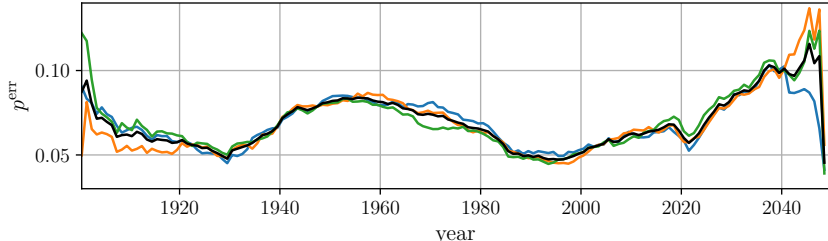

Figure 3: Error rate obtained using multi-task learning-by-confusion for the Stable Diffusion dataset as a function of θ in prompt "technology of the year θ". The colored lines depict the results obtained by training on three separate datasets, each averaged over 10 runs as described in Appendix A.1 in more detail. The black line represents their mean.

Figure 2(b) shows the error at representative points, the extrema at nodes 29, 56, 121 (corresponding to the transitions between years 1929-1930, 1956-1957, and 2021-2022) as a function of the training epoch for a single-task and multi-task network with otherwise identical network architecture and training parameters. As there is no significant overhead, the speedup of the multi-task approach is approximately given by the number of grid separators (here $K = 149$).

4   Discussion and Conclusion

In the limit of infinite model capacity, both multi-task and single-task learning models ultimately yield the same predictions, because the multi-task loss corresponds to the average of the single-task losses. However, in real-world scenarios where model expressivity, training time, and data are limited, the learning behavior depends on the particulars of the model and dataset at hand.

For the Ising dataset analyzed with a shallow 4-layer convolutional net, we observed some differences in predictions between the single-task and multi-task architectures, but not near the critical point where it would matter most. At the critical point, we found a minor overhead with respect to the ideal speedup linear in the number of grid points.

The analysis of the Stable Diffusion dataset with the 50-layer ResNet-50 demonstrates the viability of the multi-task learning-by-confusion algorithm to reveal rapid changes in the distribution of a complex dataset, where a theoretical description is not available and a larger model is required to learn the features. In this case, we found no signs of an overhead. This is in line with our general expectation that the relative overhead associated with the multi-task approach decreases as the underlying classification tasks get more complicated.

In conclusion, we find the multi-task implementation of the learning-by-confusion algorithm to provide much faster execution on large parameter grids as compared to its single-task version.

Broader Impact

The characterization of phases of matter and the study of critical phenomena are of great importance in physics. Our work contributes a faster variant of a highly popular unsupervised learning method for the data-driven detection of phase transitions.

By revealing structure in the output images of the Stable Diffusion generative model, we demonstrate an application of the learning-by-confusion method for change point detection beyond physics. For datasets outside statistical physics, the demonstrated speedup of our multi-task approach is particularly impactful as their analysis typically requires a large amount of computational resources.

Acknowledgments and Disclosure of Funding

References

[1] L. Wang, Phys. Rev. B **94**, 195105 (2016).

[2] J. Carrasquilla and R. G. Melko, Nat. Phys. **13**, 431 (2017).

[3] E. P. Van Nieuwenburg, Y.-H. Liu, and S. D. Huber, Nat. Phys. **13**, 435 (2017).

[4] S. J. Wetzel, Phys. Rev. E **96**, 022140 (2017).

[5] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová, Rev. Mod. Phys. **91**, 045002 (2019).

[6] F. Schäfer and N. Lörch, Phys. Rev. E **99**, 062107 (2019).

[7] E. Greplova, A. Valenti, G. Boschung, F. Schäfer, N. Lörch, and S. D. Huber, New J. Phys. **22**, 045003 (2020).

[8] J. Carrasquilla, Adv. Phys.: X **5**, 1797528 (2020).

[9] J. Arnold, F. Schäfer, M. Žonda, and A. U. J. Lode, Phys. Rev. Res. **3**, 033052 (2021).

[10] J. Carrasquilla and G. Torlai, PRX Quantum **2**, 040201 (2021).

[11] A. Dawid, J. Arnold, B. Requena, A. Gresch, M. Płodzień, K. Donatella, K. A. Nicoli, P. Stornati, R. Koch, M. Büttner, *et al.*, arXiv:2204.04198 (2022).

[12] Y.-H. Liu and E. P. L. van Nieuwenburg, Phys. Rev. Lett. **120**, 176401 (2018).

[13] M. J. S. Beach, A. Golubeva, and R. G. Melko, Phys. Rev. B **97**, 045207 (2018).

[14] P. Suchsland and S. Wessel, Phys. Rev. B **97**, 174435 (2018).

[15] S. S. Lee and B. J. Kim, Phys. Rev. E **99**, 043308 (2019).

[16] W. Guo, B. Ai, and L. He, arXiv:2005.10505 (2020).

[17] Y. A. Kharkov, V. E. Sotskov, A. A. Karazeev, E. O. Kiktenko, and A. K. Fedorov, Phys. Rev. B **101**, 064406 (2020).

[18] Comput. Mater. Sci. **198**, 110702 (2021).

[19] A. Bohrdt, S. Kim, A. Lukin, M. Rispoli, R. Schittko, M. Knap, M. Greiner, and J. Léonard, Phys. Rev. Lett. **127**, 150504 (2021).

[20] J. Arnold and F. Schäfer, Phys. Rev. X **12**, 031044 (2022).

[21] M. Richter-Laskowska, M. Kurpas, and M. M. Maśka, Phys. Rev. E **108**, 024113 (2023).

[22] M. A. Gavreev, A. S. Mastiukova, E. O. Kiktenko, and A. K. Fedorov, New J. Phys. **24**, 073045 (2022).

[23] D. Zvyagintseva, H. Sigurdsson, V. K. Kozin, I. Iorsh, I. A. Shelykh, V. Ulyantsev, and O. Kyriienko, Commun. Phys. **5**, 8 (2022).

[24] H. Schlömer and A. Bohrdt, SciPost Phys. **15**, 099 (2023).

[25] W. Guo and L. He, New J. Phys. **25**, 083037 (2023).

[26] J. Arnold, F. Schäfer, A. Edelman, and C. Bruder, arXiv:2306.14894 (2023).

[27] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, arXiv:2112.10752 (2021).

[28] R. Caruana, Mach. Learn. **28**, 41 (1997).

[29] S. Ruder, arXiv:1706.05098 (2017).

[30] L. Onsager, Phys. Rev. **65**, 117 (1944).

[31] K. He, X. Zhang, S. Ren, and J. Sun, in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* (2016).

[32] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, in *Adv. Neural Inf. Process. Syst.*, Vol. 32, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Curran Associates, Inc., 2019).

[33] C. Schuhmann, R. Beaumont, R. Vencu, C. Gordon, R. Wightman, M. Cherti, T. Coombes, A. Katta, C. Mullis, M. Wortsman, *et al.*, Adv. Neural Inf. Process. Syst. **35**, 25278 (2022).

[34] J. Arnold, F. Schäfer, and N. Lörch, "Python implementation of multi-task learning-by-confusion," (2023), `https://github.com/multitaskLBC` – GitHub repository.

[35] D. Kingma and J. Ba, arXiv:1412.6980 (2014).

A Appendix

A.1 Implementation

A `Python` implementation of the single-task and multi-task learning by confusion method is available at [34]. The hyperparameters used to generate the figures in this article are summarized in Table 1. Here, M_{train} and M_{valid} refer to the number of samples per grid point within the training and validation set, respectively. For training, we use the Adam optimizer [35] with a learning rate given in Table 1.

Table 1: Hyperparameters employed in this paper (default settings are used except where explicitly stated).

Figure	M_{train}	M_{valid}	batch size	learning rate	training epochs
1(c)	1000	1000	1024	1×10^{-4}	50
2(a)	1000	1000	1024	1×10^{-4}	30
2(b)	1000	1000	1024	5×10^{-4}	50
3	3×16	3×11	256	5×10^{-5}	150

A.1.1 Stable Diffusion

The total Stable Diffusion dataset contains 3×27 images per year. In Fig. 3, the assignment of training and validation set is randomized before each run, so the distinction between training set and validation set does not strictly apply. The number of training epochs was set to 150 as the overall validation loss started increasing again around that point. Because the accuracy at individual nodes may have peaked earlier already, each curve represents the minimal error across all epochs. In Fig. 2, we only use one of the three subsets making up the Stable Diffusion dataset corresponding to the seed numbers $N < 27$ in the accompanying `Python` script, and the images with seed numbers $N < 16$ are fixed as the training set. A `Python` script with seeds to generate the dataset and a notebook to perform training can be found at [34].

A.1.2 Ising Model

The Ising dataset is generated by sampling spin configurations from Boltzmann distributions at various temperatures via the Metropolis-Hastings algorithm. The lattice is initialized in a state with all spins pointing up and updated by drawing a random spin that is flipped with probability $\min(1, e^{-\Delta E/k_\mathrm{B}T})$, where ΔE is the energy difference resulting from the spin flip. In a thermalization period, we sweep the complete lattice 10^5 times. Afterward, we collect samples, increase the temperature, and start another thermalization period.

For the Ising model, we utilized fixed training and validation sets, as randomization in the split was unnecessary due to the abundance of data. The convolutional neural network architecture utilized to produce the results in Figs. 1(c) and 2(a) can be found in the accompanying code [34].

A.2 Compute Resources

For our computations, we use an NVIDIA GTX 3090 GPU and an Intel i9-10900K CPU, where a training and validation epoch finishes within a few seconds for both the Ising and Stable Diffusion dataset.