NANOSYSTEMS:
PHYSICS, CHEMISTRY, MATHEMATICS

Lobanov I.S. *Nanosystems:*
Phys. Chem. Math., 2023, **14** (6), 613–625.
http://nanojournal.ifmo.ru
DOI 10.17586/2220-8054-2023-14-6-613-625

Original article

Toward nanomagnetic implementation of energy-based machine learning

Igor S. Lobanov[1,a,b]

[1]Faculty of Physics, ITMO University, Lomonosova Str. 9, Saint Petersburg, 191002 Russia

[a]igor.lobanov@metalab.ifmo.ru, [b]lobanov.igor@gmail.com

PACS 75.78.-n, 05.65.+b

ABSTRACT Some approaches to machine learning (ML) such as Boltzmann machines (BM) can be reformulated as energy based models, which are famous for being trained by minimization of free energy. In the standard contrastive divergence (CD) learning the model parameters optimization is driven by competition of relaxation forces appearing in the target system and the model one. It is tempting to implement a physical device having natural relaxation dynamics matching minimization of the loss function of the ML model. In the article, we propose a general approach for the design of such devices. We systematically reduce the BM, the restricted BM and BM for classification problems to energy based models. For each model we describe a device capable of learning model parameters by relaxation. We compare simulated dynamics of the models using CD, Monte-Carlo method and Langevin dynamics. Benchmarks of the proposed devices on generation and classification of hand-written digits from MNIST dataset are provided.

KEYWORDS Machine learning, Boltzmann machine, energy based model, dissipative training.

ACKNOWLEDGEMENTS The work is supported by Russian Science Foundation grant 22-22-00565:
https://rscf.ru/en/project/22-22-00565/

FOR CITATION Lobanov I.S. Toward nanomagnetic implementation of energy-based machine learning. *Nanosystems: Phys. Chem. Math.*, 2023, **14** (6), 613–625.

1. Introduction

The recent machine learning (ML) development revolutionized the field of artificial intelligence and unleashed rapid progress in natural language processing, robot motion planning, art generation and so on. Part of the rapid development can be attributed to the appearance of new ML models such as transformers, but the appearance of human-like behavior is connected with the increase of capacity of the models [1]. At the moment, training of large models is an expensive task requiring 10^5 hours of A100 GPU and costs millions of dollars. To obtain even more complex models further increase of performance and memory size of hardware simulating the models is required. The state of the art hardware originates from computer graphics accelerators and is not optimal for computation of essentially analogue artificial neural networks (ANN) and similar models. This leads in particular to the huge energy consumption of electric computers based ML compared to biological neural networks.

Various new approaches for implementation of ML using new physical principles are proposed, e.g. photoelectronic circuits speed up image processing thousands times faster than Tesla A100 [2]. One of the approaches lies in designing ML models on top of magnetic devices [3, 4]. The approach extends the boundaries of spintronics, which is a likely replacement of modern electronics. Some ML models are especially appealing for implementation as magnetic devices since they originated from physical models. One approach uses topological solitons to transmit impulses between artificial neurons [5]. Probably before the emergence of such neuromorphic devices, we will see the appearance of magnetic racetrack memory working on the same principle [6, 7]. The ML model, which is most close to a physical system is the Boltzmann machine (BM) [8], that can be considered the stochastic Ising machine, and therefore can be physically implemented as interconnected spin islands [9]. BM has a variety of uses including associative memory, solving combinatorial optimization problems, generation of images, features extraction and so on.

BM consists of spins, which interact with each other and the environment, resulting in Boltzmann distribution of the spins energy. The BM is trained by variation of the interaction constants to match the probability distribution (PD) to a target one. In the simulation, sequence of states of BM are computed by Monte-Carlo methods and its weights are optimized using contrastive divergence (CD) method. The CD method implements a variant of Hebbian rule, which makes training of BM similar to the learning of biological organisms. Earlier attempts to train BM showed that it is not practical for any hard problems, but BM with restricted connections (RBM) can be efficiently simulated and are in use now especially in the form of stacked BMs.

Besides numerical simulation of BM as a variant of ANN, BM has been implemented as a number of physical devices, including the one based on tunnel magnetic junctions [10]. The Ising model also forms a basis for D-Wave quantum computer, which however uses quantum annealing instead of stochastic dynamics for computations [11]. The vast majority of BM implementations do not support training in hardware. Instead they rely on traditional computers to

solve the optimization problem. We will pursue the goal to develop a self-training device capable of adapting to continuous feed of training data. An especially simple device can be obtained, if training can be reformulated as a naturally occurring physical process.

BM belongs to the class of so-called energy-based models (EBM) [12]. A useful feature of the models is that minimization of the loss function is equivalent to energy minimization [13, 14]. The observation makes it possible to apply a physically inspired method to ML, e.g. simulation of Langevin dynamics achieving better results than other likelihood models [15].

Our initiative is to implement BM loss minimization as a relaxation in a magnetic device. In the article [16], we proposed an extended BM machine whose weights are included to degrees of freedom (DoF). We showed that the energy minimization of the system leads to memorization of the trained samples. This kind of device can be used as an associative memory, but it is not suitable for answering questions with a non-deterministic answer. In the present article we generalize the approach to a stochastic model capable of generation of arbitrary probability distributions. Similar idea was carried out in [17] that made it possible to implement plasticity of the energy landscape of a few atom BM demonstrating self-learning to some extent.

In the article, we systematically derive EBMs for BM with only visible neurons (Section 2), BM containing hidden neurons (Section 3) and BM with loss function specialized for classification problems (Section 4). For all the variants of BM, we propose an approach for physical implementation of a device, whose relaxation matches minimization of the corresponding loss functions. We also provide examples of application of the approach, studying a simple model of tuning variation of normal distribution in Section 2.1 and benchmarking hand-written digits generation (Section 3.2) and classification (Section 4.1).

2. Dissipative training

Consider an arbitrary physical system with energy functional $E[x]$ depending on the system state x. Suppose the system is in thermal equilibrium with a thermal reservoir, hence the state of the system is random and is described by Boltzmann probability distribution:

$$p(x) = Z^{-1} Z(x), \quad Z(x) = e^{-\beta E[x]},$$

where $\beta = 1/k_B T$ is the inverse temperature, and Z is the partition function:

$$Z = \sum_x Z(x) \quad \Rightarrow \quad \sum_x p(x) = 1.$$

We use notation for summation over a discrete state space for simplicity of presentation, however, the theory is valid for arbitrary measurable state space and the Lebesgue–Stieltjes integral can be substituted for sums, and an appropriate probability measure for $Z(x)$. Helmholtz free energy is defined by

$$F = -\frac{1}{\beta} \log Z \quad \Rightarrow \quad p(x) = e^{-\beta(E[x]-F)}. \tag{1}$$

The Shannon entropy is defined by

$$H(p) = \mathbb{E}_{x\sim p}[-\log p(x)] = -\sum_x p(x) \log p(x).$$

It coincides with the Gibbs entropy up to the Boltzmann constant factor: $S = k_B H$. Due to (1), the entropy is related to the free energy and mean value of energy with respect to the distribution p:

$$H(p) = \beta \mathbb{E}_p[E - F] = \beta(\mathbb{E}_p[E] - F).$$

The mean energy can be computed in terms of the partition function:

$$\frac{\partial(\beta F)}{\partial \beta} = -\frac{\partial \log Z}{\partial \beta} = -Z^{-1} \frac{\partial Z}{\partial \beta} = Z^{-1} \sum_x Z(x) E[x] = \mathbb{E}_p[E]. \tag{2}$$

In machine learning, the system can be used as a generator of samples with PD p. In practice, we want the distribution to match a given distribution $\tilde{p}$. To compare the model distribution p with a target one $\tilde{p}$, the Kullback–Leibler (KL) divergence can be used:

$$D_{KL}(\tilde{p} \parallel p) = \mathbb{E}_{x\sim\tilde{p}}\left[\log \frac{\tilde{p}(x)}{p(x)}\right] = \sum_x \tilde{p}(x) \log \frac{\tilde{p}(x)}{p(x)} = H(\tilde{p}, p) - H(\tilde{p}),$$

where $H(\tilde{p})$ is the entropy for the distribution $\tilde{p}$ and the cross-entropy is defined by:

$$H(\tilde{p}, p) = \mathbb{E}_{x\sim\tilde{p}}[-\log p(x)] = -\sum_x \tilde{p}(x) \log p(x).$$

The KL divergence is not-negative and is equal to zero if and only if the distributions p and $\tilde{p}$ coincide. Although D_{KL} is not a metric, since it is not symmetric, the two mentioned properties make the KL divergence a good choice for loss function for system parameters optimization for the system to learn the target distribution $\tilde{p}$. The learning is possible, if

the system depends on parameters θ then we can minimize loss function with respect to the parameters. The parameters will be discussed below, for the moment, we can work in very general settings.

Since the model PD p is generated by the considered system, the cross-entropy can be expressed in terms of the mean and free energy using (1):

$$H(\tilde{p}, p) = \beta \mathbb{E}_{\tilde{p}}[E - F] = \beta(\mathbb{E}_{\tilde{p}}[E] - F).$$

The expression for the cross-entropy differs from the entropy of p only by the distribution used for averaging the energy.

Suppose that the distribution $\tilde{p}$ is generated by an analogous system, but with a different expression for the energy. We will put tilde above all values related to the system with the PD $\tilde{p}$ over the states, in particular,

$$\tilde{p}(x) = \exp\Big(-\beta(\tilde{E}(x) - \tilde{F})\Big).$$

The KL divergence simplifies to:

$$D_{KL}(\tilde{p} \parallel p) = \overbrace{\beta(\mathbb{E}_{\tilde{p}}[E] - F)}^{H(\tilde{p},p)} - \overbrace{\beta(\mathbb{E}_{\tilde{p}}[\tilde{E}] - \tilde{F})}^{H(\tilde{p})} = \beta \mathbb{E}_{\tilde{p}}[E - \tilde{E}] + \beta(\tilde{F} - F).$$

Therefore, the learning of the distribution $\tilde{p}$ is equivalent to equalization of the free energies for $\tilde{p}$ and p.

Let both energies E and $\tilde{E}$ be two implementations of the same model for different parameters:

$$E(x) = E(x; \theta), \quad \tilde{E}(x) = E(x; \tilde{\theta}).$$

Then the learning is done by optimizing the loss with respect to θ. The simplest optimization procedure is the gradient descend, when the parameters are updated according to the rule:

$$\theta \mapsto \theta - \nu \frac{\partial D_{KL}}{\partial \theta},$$

where ν defines learning rate. Since the values with tilde do not depend on θ, the KL divergence simplifies:

$$\frac{\partial D_{KL}}{\partial \theta} = \beta \frac{\partial}{\partial \theta}(\mathbb{E}_{\tilde{p}}[E] - F).$$

In virtue of independence of $\tilde{p}$ on θ, the differentiation and averaging can be swapped. The free energy derivative can be computed as follows:

$$\frac{\partial F}{\partial \theta} = -\frac{1}{\beta Z}\frac{\partial Z}{\partial \theta} = -\frac{1}{\beta Z}\sum_x \frac{\partial Z(x)}{\partial \theta} = \frac{1}{Z}\sum_x Z(x)\frac{\partial E(x)}{\partial \theta} = \sum_x p(x)\frac{\partial E(x)}{\partial \theta} = \mathbb{E}_p\left[\frac{\partial E}{\partial \theta}\right]. \tag{3}$$

Finally, the parameter update rule is driven by the velocity f:

$$\theta \mapsto \theta + (\nu\beta) f, \quad f = \mathbb{E}_{\tilde{p}}\left[\frac{\partial E}{\partial \theta}\right] - \mathbb{E}_p\left[\frac{\partial E}{\partial \theta}\right]. \tag{4}$$

It is worth noting that the parameter $\tilde{\theta}$ does not present in the final expression. Moreover, the origin of $\tilde{p}$ does not matter below, so we will not restrict $\tilde{p}$ in any way below.

The key observation of the article is that the update rule coincides with the relaxation dynamics for the variable θ. Indeed, suppose θ is a DoF for the considered system, but we assume θ to be a slow variable, so that at each moment of time, the distribution of x is given by the p for the fixed θ. The relaxation dynamics of θ is described by:

$$\dot{\theta} = -\alpha \frac{\partial E}{\partial \theta},$$

where α is the damping parameter. Averaging over the ensemble, we obtain the second addendum in (4). Moreover, assuming ergodicity, the addendum can be approximated by the time average:

$$\mathbb{E}_{x\sim p}\left[\frac{\partial E(x;\theta)}{\partial \theta}\right] = \left\langle \frac{\partial E}{\partial \theta} \right\rangle_t.$$

The first addendum in (4) is more complex, since the distribution $\tilde{p}$ is external for the system. To take $\tilde{p}$ into account, we extend our system, including new degrees of freedom $\tilde{x}$. The vector $\tilde{x}$ belongs to the same state space as x, but we assume $\tilde{x}$ be externally driven, so that its distribution at every moment of time is given by $\tilde{p}$ and they are uncorrelated at different moments of time. Then the first addendum can be obtained as time averaging of:

$$\mathbb{E}_{\tilde{x}\sim\tilde{p}}\left[\frac{\partial E(\tilde{x};\theta)}{\partial \theta}\right] = \left\langle \frac{\partial E(\tilde{x};\theta)}{\partial \theta} \right\rangle_t.$$

We define total energy of the new system according to the requirements:

(1) Energy $E[\tilde{x}]$ is opposite to the energy $E[x]$.
(2) $\tilde{x}$ and x interact with the same parameters state θ.

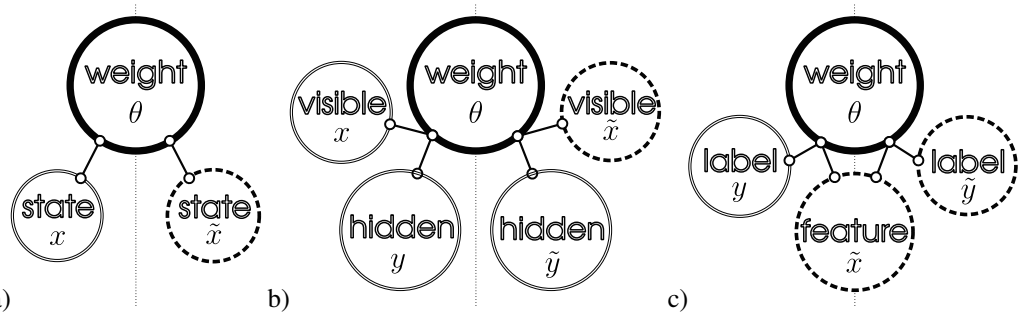

FIG. 1. Proposed designs of self-training BM. Circles represent DoF: bold line marks slow variable/weights, double-line marks fast variables, punctured boundary marks externally driven DoF. The system is split to two lobes: the right one is a copy of the left one with externally driven DoFs substituted for some fast variables. Segments between circles indicate interactions. (a) BM learning PD demonstrated on $\tilde{x}$ and generating the PD on x. (b) BM contains hidden neurons y and $\tilde{y}$; training data is demonstrated on $\tilde{x}$, the result is read from x. (c) BM for classification problem; features $\tilde{x}$ and labels $\tilde{y}$ from the training set are driven by external forces; predicted PD over labels is read from y.

Then the total energy is defined by:

$$E^T = E(x;\theta) - E(\tilde{x};\theta).$$

Summing up all properties above, we conclude that f can be obtained by averaging of the relaxation force:

$$f = \left\langle \frac{\partial E^T}{\partial \theta} \right\rangle_t.$$

We conclude that the relaxation dynamics of the extended system is a continuous version of the stochastic gradient descent for the minimization of the KL divergence. Since the approach is quite general, it opens many opportunities for implementation of self-learning devices. In the following sections, we consider several examples.

2.1. Standard deviation learning

Consider simple case of a single continuous random variable x having normal distribution with mean m and standard deviation $\sigma = \theta^{-\frac{1}{2}}$:

$$p(x) = Z^{-1} \exp\left(-\frac{\theta(x-m)^2}{2}\right).$$

The PD can be considered Boltzmann distribution for the energy

$$E = \frac{\theta(x-m)^2}{2},$$

and the constant temperature $\beta = 1$. The partition function in the case is well known:

$$Z = \sqrt{2\pi/\theta\beta}.$$

The optimization of the mean m is relatively simple and was already solved in [16]. Here we let $m = 0$ and focus on optimization of θ. The free energy of the model system:

$$F = \frac{1}{2\beta}\left(\log\theta + \log\beta - \log(2\pi)\right).$$

According to (2) the mean energy is:

$$\mathbb{E}_p[E] = \frac{\partial(\beta F)}{\partial\beta} = \frac{1}{2\beta}. \tag{5}$$

Suppose target distribution $\tilde{p}$ is also normal with standard deviation $\tilde{\sigma} = \tilde{\theta}^{-\frac{1}{2}}$. The expectation value of E with respect to $\tilde{p}$ is obtained by rescaling:

$$\mathbb{E}_{\tilde{p}}[E] = \left\langle \frac{\theta}{\tilde{\theta}}\tilde{E} \right\rangle_{\tilde{p}} = \frac{1}{2\beta}\frac{\theta}{\tilde{\theta}}.$$

KL divergence between the distributions is given by:

$$D_{KL}(\tilde{p} \parallel p) = \beta\mathbb{E}_{x\sim\tilde{p}}[E(x) - \tilde{E}(x)] + \beta(\tilde{F} - F) = \frac{1}{2}\left(\frac{\theta}{\tilde{\theta}} - 1\right) + \frac{1}{2}\left(\log\tilde{\theta} - \log\theta\right).$$

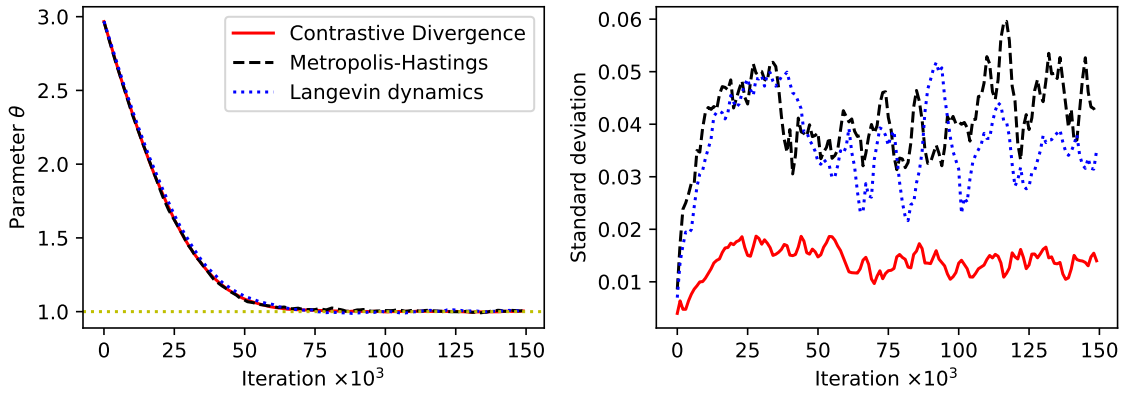

FIG. 2. Convergence of estimation of standard deviation parameter θ using EBM: CD learning (solid line), free energy minimization using Metropolis–Hastings algorithm (dashed line) and Langevin dynamics (dotted line). Target value of the parameter is 1, initial value is 3. Mean value of the parameter estimate (left panel) and standard deviation of (right panel) obtained by averaging over 20 runs and 1000 consecutive samples.

The divergence has only one minimum at $\theta = \tilde{\theta}$ as expected. The derivative of the KL divergence will be used below to make optimization algorithms:

$$\frac{\partial D_{KL}}{\partial \theta} = \frac{1}{2}\left(\frac{1}{\tilde{\theta}} - \frac{1}{\theta}\right). \tag{6}$$

2.2. Dissipative learning

Above we considered the parameter θ to be fixed. Here we suppose that θ a slow DoF of the system. Then relaxation dynamics will push θ in the direction opposite to

$$\frac{\partial E}{\partial \theta} = -\frac{x^2}{2}.$$

Assuming x is a fast variable, the derivative can averaged over x:

$$\mathbb{E}_{x\sim p}\left[\frac{\partial E(x;\theta)}{\partial \theta}\right] = -\sqrt{\frac{\theta}{2\pi}}\int_{-\infty}^{\infty}\frac{x^2}{2}\exp\left(-\frac{\theta x^2}{2}\right)\mathrm{d}x = -\frac{1}{2\theta}.$$

It is worth noting that the derivative of the mean energy with respect to θ is zero according to (5), that is the expectation value and the differentiation do not commute. It turns out that the dissipation generates the same dynamics as the second addendum in (6). The first addendum here gives a reference point and is essential to obtain the correct stationary state. The first addendum can also be interpreted as mean energy, but with respect to the target distribution $\tilde{p}$:

$$\mathbb{E}_{\tilde{x}\sim\tilde{p}}\left[\frac{\partial E(\tilde{x};\theta)}{\partial \theta}\right] = -\frac{1}{2\tilde{\theta}}.$$

Then optimization direction will coincide with mean relaxation force if we expand the system with the second part depending on $\tilde{x}$ and assuming the second part has opposite energy to the first one:

$$\frac{\partial D_{KL}}{\partial \theta} = \mathbb{E}_{x\sim p,\tilde{x}\sim\tilde{p}}\left[\frac{\partial E^T}{\partial \theta}\right], \quad E^T(x,\tilde{x};\theta) = E(x;\theta) - E(\tilde{x};\theta).$$

In the real system, the expectation values can be obtained either by averaging over ensembles or over time. The distribution $\tilde{p}$ can be enforced to the variable $\tilde{x}$ by an external force demonstrating the training set to the device. Obtaining opposite signs of energy for two parts of the system for the same value of the parameter θ is the most challenging task in designing real devices. Consider one example of such an implementation. Energy of liquid crystal interaction with an external electric field has the following form [18]:

$$E = -\frac{\Delta\epsilon(\mathbf{n}\cdot\mathbf{E})^2}{8\pi},$$

where $\mathbf{E}$ is electric field, $\mathbf{n}$ is the director vector, and $\Delta\epsilon$ is anisotropy of permittivity. Let the electric field have fixed direction $\mathbf{z}$ and be proportional to the parameter θ. Let x be projection of the director $\mathbf{n}$ to $\mathbf{z}$. Then the energy coincides with the considered above up to a constant. Sign of the constant is determined by $\Delta\epsilon$ which can be both positive and negative depending on the exact choice of the liquid crystal. It means that by choosing $\Delta\epsilon$ of opposite signs for two lobes x and $\tilde{x}$ we will obtain opposite energies for the same value of the external field θ.

2.3. Contrastive divergence training

Contrastive divergence (CD) is probably the most popular method used for training EBMs. The method belongs to the family of gradient descent methods sharing the update rule:

$$\theta \mapsto \theta + \eta f, \quad f \approx -\frac{\partial D_{KL}}{\partial \theta},$$

where $\eta > 0$ is the learning rate constant and f is an approximation to the steepest descent direction on the loss surface. The optimization algorithm here is a variant of stochastic gradient descent:

(1) Get a sample $\tilde{x}$ from the training set.
(2) Sample x from the distribution p given for the current θ.
(3) Compute dissipation force $f = \partial E(\tilde{x};\theta)/\partial\theta - \partial E(x;\theta)/\partial\theta$.
(4) Update weights $\theta \mapsto \theta + \eta f$.
(5) Repeat.

To estimate the force f more accurately, one can sample values in batches and then average the force over all batches. Nevertheless, if the learning rate η is sufficiently small, the averaging of the generated force in time gives a good enough estimate of the expectation value to obtain convergence.

2.4. Metropolis–Hastings algorithm

For complex systems sampling values x from the PD p is generally difficult since the partition function Z is not known. It is common in this case to sample values using Metropolis–Hastings (MH) algorithm. In this case, values of x are not independent between steps, but rather form a Markov chain. To define the chain we need an auxiliary probability density $g(y|x)$ called proposal density, which gives us a new candidate y given a previous value x. We will assume g symmetric: $g(x|y) = g(y|x)$. The generation of x according to MH algorithm is done as follows:

(1) Sample y according to the distribution $g(y|x)$ for the given previous value of x.
(2) Calculate the acceptance rate $\alpha = Z(y;\theta)/Z(x;\theta)$.
(3) Generate a uniform random number $u \in [0, 1]$.
(4) If $u \leq \alpha$, accept y as a new value of x.
(5) Otherwise let x preserve its value.

The optimization algorithm with MH sampling is the same as in the previous section, but the previous value of x is used to generate its new value. The emerging correlations between adjacent samples can increase spread of the optimization direction estimates f, which will require a decrease of the training rate constant to maintain convergence.

For the benchmark below we took proposal density $g(y|x)$ to be normal with the mean value y and standard deviation 0.5.

2.5. Langevin dynamics

Dynamics arising from the Monte-Carlo method described in the previous section is suitable to obtain correct estimates of expectation values, however it does not correspond to the real dynamics. In many cases, the dynamics is described by a variant Langevin equation, such as the stochastic Landau–Lifschitz–Gilbert (LLG) equation for magnetic systems. In the discretized form the Langevin equation for one variable can be written as follows:

$$x^{t+1} = x^t - \eta' \frac{\partial E(x^t;\theta^t)}{\partial x} + W^t,$$

where W^t has a normal distribution with zero mean and standard deviation $\sqrt{2/\eta'}$ and W^t for different t are independent. The distribution of x^t is the same Boltzmann distribution as above for the inverse temperature β. Dynamics of the slow variable θ are described by relaxation in the same way as above:

$$\theta^{t+1} = \theta^t - \eta f^t, \quad f^t = \frac{\partial E(\tilde{x}^t;\theta^t)}{\partial\theta} - \frac{\partial E(x^t;\theta^t)}{\partial\theta}.$$

For the benchmark below we generated training samples $\tilde{x}$ by Langevin equation with parameter θ set to the target value $\tilde{\theta}$. Time scale for the fast variable was defined by the ratio: $\eta'/\eta = 500$.

2.6. Benchmark

We compared performance of CD, dissipative learning with MH sampling and Langevin dynamics on the learning parameter θ of the normal distribution introduced in Section 2.1. The initial value of parameter was set to 3, while the target value equals 1. We set learning rate to $2 \cdot 10^{-4}$ and made $1.6 \cdot 10^5$ steps by all algorithm repeating the procedure 20 times. For each of the algorithms we averaged estimates of the parameter θ over 1000 consecutive samples, and also estimated standard deviation of the estimates on the same intervals. The results are presented in Fig. 2.2. All the methods perform equally well on average having the same convergence rate. Correlations between adjacent samples in Monte-Carlo based method and in Langevin dynamics produce $3 - 4$ times large variation of the estimate as expected. We can conclude that real-world physical dynamics is as suitable for training the ML model as a generally accepted CD method.

3. Hidden neurons

In Section 2, we described a general approach for turning arbitrary dissipative systems into a self-training machine. The class of generated PDs is defined by energy functional E. Although we have not imposed any restrictions on the energy (except of being bounded from below) and arbitrary PD can be generated under appropriate choice of the energy, in practice we have rather restricted choice of energies available for physical implementations. Standard BM is based on the Ising model, having the quadratic energy functional:

$$E[x] = \frac{1}{2}\sum_{jk} w_{jk}x_j x_k + \sum_j b_j x_j, \tag{7}$$

with x_j representing the state of a spin j (playing the role of a neuron) taking values $x_j = \pm 1$. The parameters $\theta = (w, b)$ consist of connection strengths w_{jk} between neurons j and k and biases b_j for every neuron j. The functional E allows to tune mean values and covariance of components of x, but it is impossible to obtain a nontrivial joint distributions of three and more neurons. More complex inter-dependencies between neurons can be obtained using the same quadratic energy functional, if we include new hidden DoF.

Suppose the variables x describe visible neurons, whose state is described by the training set, and which are output of the model. Introduce new DoF denoted y playing the role of hidden variables. Now the energy functional depends on x, y and θ. We do not impose any constraints on the energy functional, but for clarity of presentation we recall an example of such functional used in restricted BM (RBM):

$$E[x] = \frac{1}{2}\sum_{jk} w_{jk}x_j y_k + \sum_j a_j x_j + \sum_k b_k y_k, \tag{8}$$

where a and b are biases for visible and hidden neurons, respectively, and the interaction is nonzero only between units of different classes (visible, hidden, weight). The energy functional for RBM coincides with (7) with the vector x extended by y with additional constraints forbidding interaction between neurons of the same type.

As in Section 2, we assume θ to be slow variables, then for given θ we have joint PD

$$p(x, y; \theta) = Z^{-1}Z(x, y), \quad Z(x, y) = \exp(-\beta E[x, y; \theta]), \quad Z = \sum_{x,y} Z(x, y). \tag{9}$$

The visible neurons state is described by the marginal PD:

$$p(x) = \sum_y p(x, y) = Z^{-1}Z(x), \quad Z(x) = \sum_y Z(x, y). \tag{10}$$

For example, the marginal distribution for RBM is notably easy to compute, since for a fixed x the components of y are independent. Easy algebra leads us to the result:

$$p(x|y) = Z^{-1}\prod_j e^{-\beta a_j x_j}\prod_k \left(\sum_j e^{-\beta\Delta_j} + e^{\beta\Delta_j}\right)^{-1}, \quad \Delta_j = \sum_k w_{jk}y_k. \tag{11}$$

Introducing sufficiently many hidden neurons, an arbitrary PD $p(x)$ can be attained.

The training problem is to find parameters θ such that the marginal distribution $p(x; \theta)$ minimizes distinction with a target PD $\tilde{p}(x)$. The loss function can be defined as cross-entropy, which as we have seen above gives the same minimum as KL divergence, since entropy of $\tilde{p}$ does not depend on θ and does not affect the result:

$$H(\tilde{p}; p) = \mathbb{E}_{x\sim\tilde{p}}[-\log p(x)] = -\sum_x \tilde{p}(x)\log p(x).$$

The only change here compared to Section 2 is that $p(x)$ is a marginal distribution. In virtue of (10),

$$H(\tilde{p}; p) = F - \sum_x \tilde{p}(x)\log Z(x), \quad F = \sum_x \tilde{p}(x)\log Z, \tag{12}$$

where F is the Helmholtz free energy.

Now we compute the gradient of the loss with respect to parameters, which is used for the loss minimization. The derivative of the free energy is the same as in Equation (3):

$$\frac{\partial F}{\partial\theta} = -\beta\sum_{x,y} p(x, y)\frac{\partial E(x, y)}{\partial\theta} = -\beta\mathbb{E}_{x,y\sim p}\left[\frac{\partial E(x, y)}{\partial\theta}\right].$$

For a fixed x the second addendum in (12) is a conditional expectation of the relaxation force:

$$\frac{\partial}{\partial\theta}\log Z(x) = Z^{-1}\sum_y \frac{\partial Z(x, y)}{\partial\theta} = -\beta\sum_y p(y|x)\frac{\partial E(x, y)}{\partial\theta} = -\beta\mathbb{E}_{x,y\sim p}\left[\frac{\partial E(x, y)}{\partial\theta}\middle| x\right], \tag{13}$$

where the conditional probability is given by:

$$p(y|x) = \frac{p(x,y)}{p(x)} = \frac{Z(x,y)}{Z(x)}.$$

Taking mean over the target PD we obtain the derivative over the second addendum:

$$\frac{\partial}{\partial\theta}\sum_x \tilde{p}(x)\log Z(x) = -\beta\sum_{x,y}\tilde{p}(x)p(y|x)\frac{\partial E(x,y)}{\partial\theta} = -\beta\mathbb{E}_{x\sim\tilde{p}}\left[\mathbb{E}_{y\sim p}\left[\frac{\partial E(x,y)}{\partial\theta}\middle|x\right]\right].$$

Finally, the loss derivative is mean difference between energy gradients averaged on the target and model distributions:

$$\frac{\partial L}{\partial\theta} = \beta\left(\mathbb{E}_{x\sim\tilde{p}}\left[\mathbb{E}_{y\sim p}\left[\frac{\partial E(x,y)}{\partial\theta}\middle|x\right]\right] - \mathbb{E}_{x,y\sim p}\left[\frac{\partial E(x,y)}{\partial\theta}\right]\right). \tag{14}$$

The averaging over hidden variables y both times happens with respect to the model distribution p, hence we obtain the same result as in Section 2, but all values should be considered as values averaged over hidden DoF.

The system whose relaxation coincides with minimization of the loss can be constructed applying the same principles as in Section 2.

(1) The average over the ensemble is changed to the time average.
(2) Every addendum in (14) is generated by one of two copies of the system, which share weights θ and exchanges, but has its own examples of x and y. The energies of the copies are chosen opposite.
(3) The variables $\tilde{x}$ are controlled by external forces, which recreate distribution $\tilde{p}$.
(4) The dynamics of the weights θ are dominated by relaxation. The weights must be slow variables, x and y are fast variables.

A schematic of the proposed device with hidden variables is shown in Fig. 2.b. The exact dynamics of the device is not important as soon as it guaranties the condition (4). For example, Monte-Carlo simulation results in the contrastive divergence method widely used for training RBM in ML. Langevin dynamics can be a more adequate method for real physical system simulation.

3.1. Monte-Carlo simulation

Correct distribution of variables x, y and $\tilde{y}$ for a fixed θ can be generated using Metropolis–Hastings algorithm. The approach does not take into account exact dynamics, but generates correct mean values of energy and energy gradients, which are the only values affecting the dynamics of slow variables θ. This method allows us to cover wide range of physical implementations of BM including magnetic in the approximation of the Ising model.

Here we focus on RBM, which energy is defined by (8). By definition visible and hidden neurons in RBM do not interact, therefore for a fixed hidden neurons state y the individual components of the visible neurons state x are independent and vice versa. In particular, conditional distribution of x given y is given by (11). This allows us to efficiently generate in the parallel manner samples of y given x and x given y. For example, the following algorithm is used to generate samples of x.

(1) Compute difference of energies of states that differs by single bit j:

$$\Delta_j = 2\sum_{jk} E_{jk}y_k + 2a_j.$$

(2) Compute conditional probability of $x_j = 1$ given y:

$$p_j = p(x_j = 1|y) = \frac{1}{1+e^{\Delta_j}}.$$

(3) Generate vector ξ of uniformly distributed values $\xi_j \in [0,1]$.
(4) Set component j of the generated vector x_j to 1 if $\xi_j <= p_j$ and to -1 otherwise.

The simulation is done step by step. We put the upper index t over all values for the time step t. The initial value of x^0 is arbitrary. On each iteration all the values are updated as follows:

(1) $\tilde{x}^{t+1}$ is sampled from the training set.
(2) Random $\tilde{y}^{t+1}$ is generated by the above algorithm for the given $\tilde{x}^{t+1}$ and θ^t.
(3) Random y^t is generated by the algorithm above for the given x^t and θ^t.
(4) Random x^{t+1} is generated by the algorithm above for given y^t and θ^t.
(5) The relaxation force is computed:

$$f^{t+1} = \frac{\partial}{\partial\theta}E(x^{t+1},y^{t+1};\theta) - \frac{\partial}{\partial\theta}E(\tilde{x}^{t+1},\tilde{y}^{t+1};\theta).$$

(6) Update the parameters:

$$\theta^{t+1} = \theta^t + \eta f^{t+1} + W^t.$$

(7) Repeat.

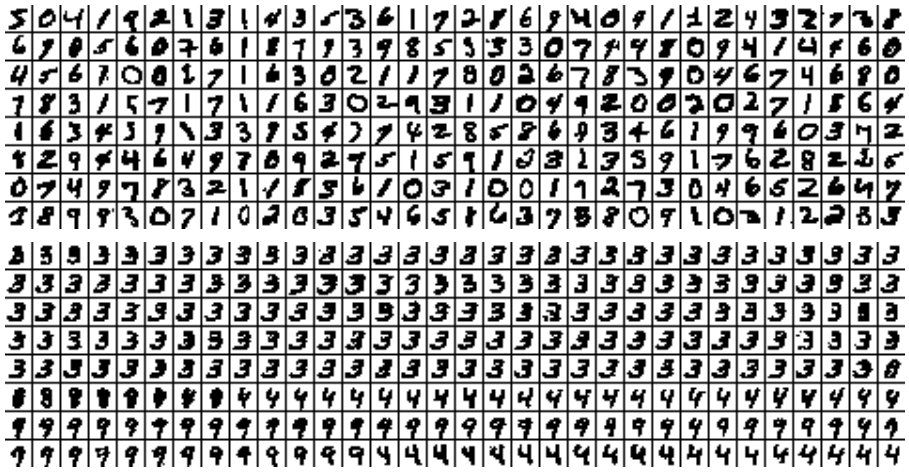

FIG. 3. (Upper panel) First 256 images of MNIST dataset scaled to 14×14 and converted to black and white. (Lower panel) State of visible neurons of the RBM after 800 epoch of training shown on every 16-th simulation step. Images are listed from left to right and from top to down.

The parameter update on step (6) is done simulating Langevin dynamics for slow variables θ. The "learning rate" parameter η should be small enough to ensure convergence, which in physical terms means that dynamics of θ must be much slower than of fast variables x and y. The addendum W is a random noise, whose amplitude is proportional to the temperature T. Since the smallest value of the error is reached at minimum of the energy, the temperature should not be large. On the other hand, small variations of θ due to the noise allows one to avoid overfitting and can be beneficial.

It is worth noting that the presented algorithm is called contrastive divergence in ML, if no noise is appended. The algorithm with the noise was earlier proposed in [15], where it was shown that it demonstrates better convergence than the CD. In most implementations of the CD the value x^{t+1} is sampled using conditional distribution $p(x|\tilde{y}^{t+1})$. In our approach, dynamics of $\tilde{x}$ and x are not directly connected, which allows us to estimate positive and negative parts of the loss gradient independently in two copies of the system.

The most difficult part of the physical implementation of the training algorithm is to obtain both positive and negative parts of the gradient as relaxation forces. In Section 2, the problem is almost non-existent, since the state of $\tilde{x}$ is externally driven and does not depend on θ, therefore, to obtain negative gradient, one can change ferromagnetic exchange to antiferromagnetic leading to negation of the energy. The trick however does not work, if there are hidden variables. Indeed, change of the exchanges energy landscape and therefore, PD of the hidden variables, which invalidates mean values of the gradient.

Pure mathematically the positive and negative terms can be considered an adversarial learning procedure, where discriminator time-flow is reversed [19]. Effective reversed time can appear in quantum mechanics [20], but it is hard to use in practice. Fortunately, the same negative gradient can be achieved by formal change of sign of β. Negative temperatures of spin systems were reported in a number of works [21–23]. Although the statistical physics for negative temperatures is studied for a long time [24, 25], a practical realization of local negative temperatures is extremely challenging.

Another possible strategy for generation of negative gradients is to shield hidden variables from the thermostat allowing its interaction only with weights and visible neurons. In the case dynamics of the hidden state y is determined exclusively by x and θ. Therefore the change of sign of all interaction constants negates energy and its gradient, but does not affect PD of y. Following this approach the self-trained system should contain two copies of x and y, with ferromagnetic exchange in one copy and antiferromagnetic in another, both interacting with the same weights θ.

One more strategy is based on reformulation of the optimization problem, abandoning the dynamics defined by (14). Considering the loss minimization method as an iterative procedure with the fixed point $p = \tilde{p}$, other energy functionals can be proposed, whose relaxation procedure has the same property. For example, let the hidden variables be shared by two parts of the system $y = \tilde{y}$. Define the total energy

$$E = -(x - \tilde{x}) \cdot wy - (x - \tilde{x}) \cdot a.$$

Here $\tilde{x}$ is defined by external impulses and x and y are subjected to Langevin dynamics. If the distributions of x and $\tilde{x}$ coincide then the system is in thermal equilibrium. The relaxation with respect to w and a can be used to adjust the PD of x, but the error measure is no longer KL divergence, and in practice leads to much worse results than the CD method. A practical choice for implementation of the negative gradient in presence of hidden variables is still an open question.

3.2. MNIST digits generation

We benchmarked performance of the proposed approach for implementation of RBM on generation of images of hand-written digits trained in MNIST dataset. MNIST dataset [26] is split to the training set of $60\,000$ and the test set of $10\,000$ gray scale images of ten different digits together with labels, which digit the image represents. We downsampled the images to size 14×14 and converted to black and white, considering the pixel black, if its luminosity is smaller than 90% of the maximum. All $60\,000$ images were used for the training, setting a new image to $\tilde{x}$ on each iteration. The beginning of the dataset is shown in the upper panel in Fig. 3.2.

The images were flattened to vectors of 196 visible neurons states, and 1000 more hidden neurons were introduced. Looping over all $60\,000$ images is considered one epoch, the training took 800 epoch in total. The training step was set $\eta = 10^{-4}$. To track convergence we calculated mean $-\log Z(x;\theta)$ over samples x taken from the test set for θ obtained on the corresponding epoch. This value is an estimation of $\mathbb{E}_{x\sim\tilde{p}}[-\log p(x;\theta)]$, which is different from the cross entropy by the absence of the free energy term $\log Z$. The applied training method does not compute free energy directly, which makes it fast, but this also prevents us from accurate estimation of the loss function. The obtained metrics are not reliable, if a lot of noise is introduced in the parameters θ on each iteration, showing rapidly decreasing loss, while real cross entropy can increase. However, in our experience the free energy does not vary significantly close to the minimum, and the metric can be used for comparison of different methods. The success of the convergence was checked by inspection of the generated images, which indeed showed close resemblance with real hand-written digits.

We run two numerical experiments: one without noise and another with normally distributed W with amplitude 0.001. The generated PD in both cases converges to the target distribution. In the presence of the noise the convergence was smooth, while without noise the loss decrease happens in step-like manner stagnating between the steps. Overall convergence rate is higher without noise. While trained BM in both cases produces correctly looking digits, the BM obtained by training without loss demonstrates faster switching between digits classes, that is smaller correlation between adjacent states.

The dynamics of the dissipative BM trained without noise is shown in Fig. 3.2 (bottom panel). The BM started from a random state, then after a burnout period of 16 steps. We recorded 256 states of the visible neurons skipping every 15 intermediate steps. The generated images reconstruct distribution of pixel colors matching MNIST dataset, but adjacent images are correlated resulting in smooth change of the digit shape. Visual inspection confirms that the dissipative BM correctly generalizes the training set producing images indistinguishable from those produced by humans.

4. BM solving classification problem

In previous sections, we considered generative models producing random values w with a learned distribution $p(x)$. For classification problems, it is often convenient to have a discriminative model, which produce random values of labels y distributed according to a learned conditional probability $p(y|x)$ given an observation of features x. This discriminative model can be implemented on top of BM with some modifications required.

The BM energy in the case is a function of features x, labels y and weights θ. The joint PD of x and y is given by Boltzmann distribution (9). In contrast to Section 3 both x and y are states of visible neurons. As shown above, addition of hidden neurons allows complex PDs, but in all formulas they are averaged out, therefore we do not list the hidden neurons explicitly in this section.

Given target distribution of labels $\tilde{p}(y|x)$ for fixed features values x, the model distribution can be compared with it using cross-entropy:

$$H(\tilde{p}, p|x) = -\sum_y \tilde{p}(y|x) \log p(y|x) = \mathbb{E}_{y\sim\tilde{p}}[-\log p(y|x)|x].$$

The loss function is commonly defined as mean of the cross-entropy over all x:

$$L = \sum_x \tilde{p}(x) H(\tilde{p}, p|x) = -\sum_{x,y} \tilde{p}(x,y) \log p(y|x) = \mathbb{E}_{x,y\sim\tilde{p}}[-\log p(y|x)].$$

Noticing (see Eq. (9)) that

$$p(y|x) = Z(x)^{-1} Z(x,y),$$

the loss function can be written in the following form:

$$L = \beta \sum_{x,y} \tilde{p}(x,y) E(x,y) + \beta \sum_{x,y} \tilde{p}(x,y) \log Z(x).$$

The first term here is the mean energy with respect to the target distribution. The second addendum is the Helmholtz free energy for a fixed x averaged over x. Consider the gradient of the loss over parameters θ. Since $\tilde{p}$ does not depend on θ, the averaging over $\tilde{p}$ can be swapped with differentiation over θ, which gives us the derivative of the first term. Using

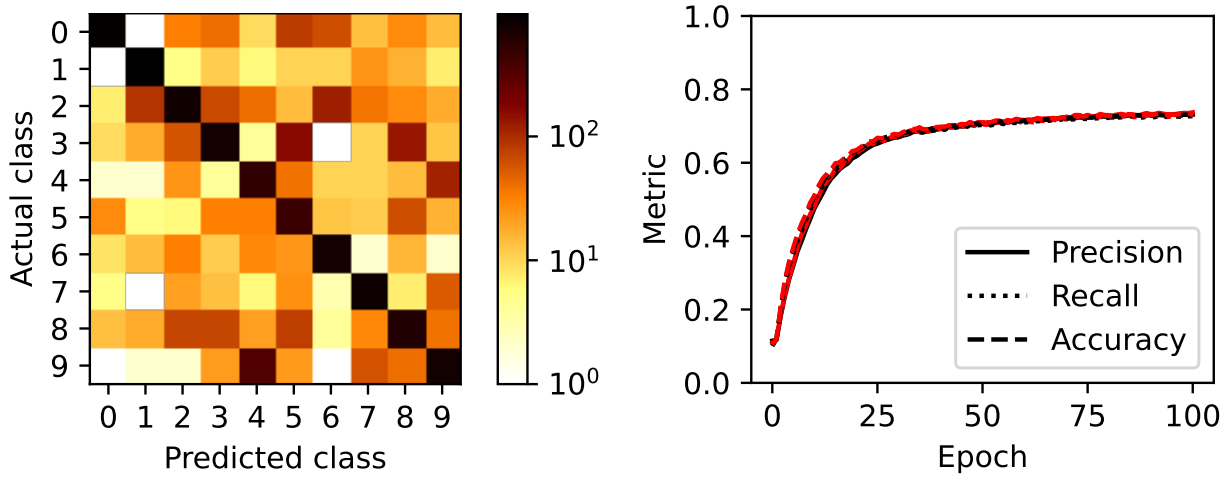

FIG. 4. Results of classification of hand-written digits from MNIST dataset. (Left panel) Confusion matrix on the test data after 100 epoch of training. (Right panel) Convergence history for accuracy and averaged over all classes precision and recall, demonstrating good agreement of all metrics. Metrics on the test set (red lines) and the training set (black lines) are almost identical.

earlier computed in (13) derivative of $\log Z(x)$, we obtain the loss derivative:

$$\frac{\partial L}{\partial \theta} = \beta \left(\sum_{x,y} \tilde{p}(x,y) \frac{\partial E(x,y)}{\partial \theta} - \sum_{x,y} \tilde{p}(x,y) \sum_{y'} p(y'|x) \frac{\partial E(x,y')}{\partial \theta} \right)$$
$$= \beta \mathbb{E}_{x\sim\tilde{p}} \left[\mathbb{E}_{y\sim\tilde{p}} \left[\frac{\partial E(x,y)}{\partial \theta} \middle| x \right] - \mathbb{E}_{y\sim p} \left[\frac{\partial E(x,y)}{\partial \theta} \middle| x \right] \right].$$

The argument of the outermost expectation value is exactly the derivative over θ of the cross-entropy $H(\tilde{p}, p)$ obtained in Section 2 assuming x fixed. Therefore, the discriminative model differs from the generative model in the way we treat the features state x: for the discriminative model x is always distributed according to the target distribution $\tilde{p}$.

The physical system whose dynamics coincides with the minimization of loss procedure, is designed using the principles stated in Section 2.

(1) The parameters θ are treated as slow DoF of the system.
(2) While the state $\tilde{x} = x$ is always driven by external forces defining target PD, a copy $\tilde{y}$ of y is introduced in such a way that interactions between x, y, θ are the same as for $x, \tilde{y}, \theta$, except for the opposite sign.
(3) Assuming ergodicity the expectation values are approximated by the time averages.
(4) The energy derivatives are treated as relaxation forces for the weight θ.

The proposed schematics of the self-training device is shown in Fig. 2.c. Its part involving labels variables y, $\tilde{y}$ is essentially the same as the schematic of the generative dissipative BM shown in Fig. 2.a, but now we have additional externally driven DoF encoding target distribution for the features $\tilde{x}$.

4.1. Digits classification benchmark

To validate the capability of the proposed method to solve complex problems, we do classification of hand-written digits from MNIST dataset using the dissipative BM. For the test, we used full size 28×28 black and white images, where the color was encoded in z projection of the spin: 1 for pure white and -1 for pure black pixel. 60 000 samples were used for training and 10 000 for testing. One loop over all training samples is considered an epoch. The training procedure took 100 epoch.

The state of the used BM consists of the features vector x, the labels vector y and the hidden units vector z. The size of the feature vector equals 784 and matches the number of pixels of the image. The labels vector y has 10 components, each component encodes the probability of the image to belong to the corresponding class, that represents one of ten digits. Both x and y compose the vector of visible neurons. For the benchmark, we set the number of hidden neurons to 200.

The energy of BM is defined by the functional:

$$E(x,y,z;\theta) = \sum_{ij} w^{xz} x_i z_k + \sum_{ij} w^{yz} y_j z_k + \sum_i b_i^x x_i + \sum_i b_i^y y_i + \sum_i b_i^z z_i,$$

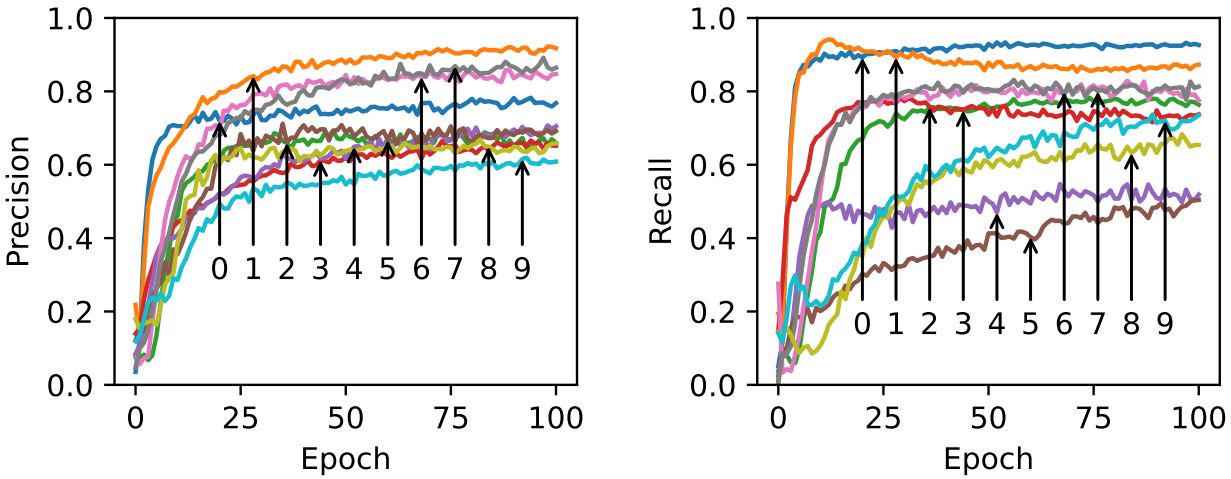

FIG. 5. Convergence history for precision (left panel) and recall (right panel) for individual classes computed on the test set.

where interaction matrices w and biases b form parameters vector $\theta = (w^{xz}, w^{yz}, b^x, b^y, b^z)$. The proposed dissipative BM contains two copies of x, y and z; we mark the second copy by tilde above the letter. The total energy of spin BM consists of energies of the two copies sharing the same parameters:

$$E^T = E(x, y, z; \theta) - E(\tilde{x}, \tilde{y}, \tilde{z}; \theta).$$

The possible strategies to negate energy of the second contribution are discussed in Section 3. During the training $\tilde{x}$ and $\tilde{y}$ are controlled by the external forces specifying training samples. The variables x, y, z and $\tilde{z}$ are distributed according to Boltzmann distribution. The dynamics was approximated by Monte-Carlo sampling of fast variables and relaxation dynamics of slow variables θ according to the algorithm:

(1) Sample new $\tilde{x}$, $\tilde{y}$ from the training set.
(2) Sample $\tilde{z}$ according to (11).
(3) Update $\theta \mapsto \theta - \eta \dfrac{\partial E^T}{\partial \theta}$.
(4) Repeat.

To reduce jitter we sample all variables in batches of 10 elements, that correspond to the physical system with 10 independent copies of each subsystem sharing the same weights.

The simulation was run on hand-written Python code using Numpy and JAX for GPU acceleration. The relaxation rate η was set to 10^{-4}. Initial weights θ were randomly sampled from the normal distribution with zero mean and standard deviation 0.1. Results of the training are shown in Fig. 4.1 and Fig. 4.1.

After 100 epoch we reached $\sim 73\%$ accuracy and very close values of precision and recall averaged over all classes. Although the metrics are much smaller than attainable by other methods, the result is comparable with the result of RBM with 200 hidden units. Increasing the number of hidden units and changing architecture of the machine including more hidden layers can significantly improve the result [15]. In the article, our main concern was comparison of the performance classical RBM and the proposed adaptation suitable for nanomagnetic implementation.

Confusion matrix shown in the left panel in Fig. 4.1 demonstrates that the most errors arise in misclassification of 9 as 4, 3 as 5 or 8, which is not much different from the errors of other methods. The same errors are confirmed by the convergence history plots shown in Fig. 4.1, pointing out that digits 4 and 5 are most confused with other digits. The metrics on the training set and on the test set are very close with precision and recall of individual digits continue to grow with every epoch, indicating the prolongation of training can lead to even better results.

Since in the proposed dissipative BM subsystem generating positive and negative update steps for the weights are generated by independent subsystems, the time required for averaging the contributions is larger than in CD method. In the BM, we are forced to use smaller training steps than in CD, therefore, convergence of simulated SBM is slower. In real magnetic nanosystems, the natural time scale defined by Larmor precession is orders of magnitude smaller than obtainable in simulation. Despite a slower convergence rate in simulation, the physical implementation of the BM is expected to be drastically faster than the ML model of BM used currently.

5. Discussion

Above, we proposed several approaches to implement energy-based ML models as physical devices. In particular, nanomagentic-devices are a natural candidate for the implementation of BM. Two problems are not completely solved and remain challenging for experimental implementation of the devices. The first problem is the necessity of three spin interactions between two neurons and the connection weight. Multispin connections in some cases give significant contribution to the total energy [27], but at the moment, it is not clear how to design magnetics with the desired multi-spin interactions. Probably more promising is creation of an effective multispin exchange by introducing auxiliary spins and averaging over them in the spirit of the work [28].

The second challenging problem is learning the weights of the hidden spins by relaxation. In this case, the introduction of lobes with opposite energies also affects the PD of the hidden neurons. The solution here can lay in avoidance of the hidden neurons interaction with the thermostat, and considering stochastic LLG dynamics of the spins, instead of Fokker–Planck equation. At least a similar approach allowed one to create a simple self-learning device experimentally [17].

References

[1] Min B., Ross H., Sulem E., et al. Recent Advances in Natural Language Processing via Large Pre-trained Language Models: A Survey. *ACM Computing Surveys*, 2023, **56**(2), P. 1–40.

[2] Chen Y., Nazhamaiti M., Xu H., et al. All-analog photoelectronic chip for high-speed vision tasks. *Nature*, 2023, **623**(7985), P. 48–57.

[3] Grollier J., Querlioz D., Camsari K.Y., et al. Neuromorphic spintronics. *Nature Electronics*, 2020, **3**(7), P. 360–370.

[4] Li S., Kang W., Zhang X., et al. Magnetic skyrmions for unconventional computing. *Materials Horizons*, 2021, **8**(3), P. 854–868.

[5] Yokouchi T., Sugimoto S., Rana B., et al. Pattern recognition with neuromorphic computing using magnetic fieldinduced dynamics of skyrmions. *Science Advances*, 2022, **8**(39), P. eabq5652.

[6] Gu K., Guan Y., Hazra B.K., et al. Three-dimensional racetrack memory devices designed from free- standing magnetic heterostructures. *Nature Nanotechnology*, 2022, 17(10), P. 1065–1071.

[7] Blasing R., Khan A.A., Filippou P.C., et al. Magnetic Racetrack Memory: From Physics to the Cusp of Applications Within a Decade. *Proceedings of the IEEE*, 2020, **108**(8), P. 1303–1321.

[8] Marullo C., Agliari E. Boltzmann Machines as Generalized Hopfield Networks: A Review of Recent Results and Outlooks. *Entropy*, 2020, **23**(1), 34 pp.

[9] Khater A., Abou Ghantous M. Magnetic properties of 2D nano-islands I: Ising spin model. *Journal of Magnetism and Magnetic Materials*, 2011, 323(22), P. 2717–2726.

[10] Yang S., Shin J., Kim T., et al. Integrated neuromorphic computing networks by artificial spin synapses and spin neurons. *NPG Asia Materials*, 2021, **13**(1), P. 1–10.

[11] Dixit V., Selvarajan R., Alam M.A., et al. Training Restricted Boltzmann Machines With a D-Wave Quantum Annealer. *Frontiers in Physics*, 2021, **9**, 589626, 10 pp.

[12] Lecun Y., Chopra S., Hadsell R., et al. A tutorial on energy-based learning. In G. Bakir, T. Hofman, B. Scholkopt, A. Smola, B. Taskar, editors, Predicting structured data. MIT Press. 2006.

[13] Dayan P., Hinton G.E., Neal R.M., et al. The Helmholtz Machine. 1999. *Neural Comput.*, 1995, **7**(5), P. 889–904.

[14] Teh Y.W., Welling M., Osindero S., et al. Energy-Based Models for Sparse Overcomplete Representa- tions. *Journal of Machine Learning Research*, 2003, **4**, P. 1235–1260.

[15] Du Y., Mordatch I. Implicit Generation and Modeling with Energy Based Models. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

[16] Lobanov I. Spin Boltzmann machine. *Nanosystems: Physics, Chemistry, Mathematics*, 2022, **13**(6), P. 593–607.

[17] Kiraly B., Knol E.J., Van Weerdenburg W.M.J., et al. An atomic Boltzmann machine capable of self- adaption. *Nature Nanotechnology*, 2021, **16**(4), P. 414–420.

[18] Aksenova E., Dobrun L., Kovshik A., et al. Magnetic Field-Induced Macroscopic Alignment of Liquid- Crystalline Lanthanide Complexes. *Crystals*, 2019, **9**(10), P. 499.

[19] Yair O., Michaeli T. Contrastive Divergence Learning is a Time Reversal Adversarial Game. URL/arXiv: https://doi.org/10.48550/arXiv.2012.03295.

[20] Colombo S., Pedrozo-Peafiel E., Adiyatullin A.F., et al. Time-reversal-based quantum metrology with many-body entangled states. *Nature Physics*, 2022, **18**(8), P. 925–930.

[21] Purcell E.M., Pound R.V. A Nuclear Spin System at Negative Temperature. *Physical Review*, 1951, **81**(2), P. 279–280.

[22] Hakonen P., Lounasmaa O.V. Negative Absolute Temperatures: "Hot" Spins in Spontaneous Magnetic Order. *Science*, 1994, **265**(5180), P. 1821–1825.

[23] Medley P., Weld D.M., Miyake H., et al. Spin Gradient Demagnetization Cooling of Ultracold Atoms. *Physical Review Letters*, 2011, **106**(19), 195301, 4 pp.

[24] Ramsey N.F. Thermodynamics and Statistical Mechanics at Negative Absolute Temperatures. *Physical Review*, 1956, **103**(1), P. 20–28.

[25] Baldovin M., Iubini S., Livi R., et al. Statistical mechanics of systems with negative temperature. *Physics Reports*, 2021, 923, P. 1–50.

[26] Deng L. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 2012, **29**(6), P. 141–142.

[27] Hoffmann M., Blgel S. Systematic derivation of realistic spin models for beyond-Heisenberg solids. *Physical Review B*, 2020, **101**(2), P. 024418.

[28] Yoshioka N., Akagi Y., Katsura H. Transforming generalized Ising models into Boltzmann machines. *Physical Review E*, 2019, **99**(3), P. 032113.

Submitted 10 October 2023; revised 11 November 2023; accepted 7 December 2023

Information about the authors:

Igor S. Lobanov – Faculty of Physics, ITMO University, Lomonosova Str. 9, Saint Petersburg, 191002 Russia; ORCID 0000-0001-8789-3267; igor.lobanov@metalab.ifmo.ru; lobanov.igor@gmail.com