CMST 27(3) 99–107 (2021) DOI:10.12921/cmst.2021.0000011

Applying a Quantum Annealing Based Restricted Boltzmann Machine for MNIST Handwritten Digit Classification

K. Kurowski[1], **M. Slysz**[1], **M. Subocz**[1], **R. Różycki**[2]

[1] *Poznan Supercomputing and Networking Center*
ul. Jana Pawła II 10
61-139 Poznan, Poland
E-mail: krzysztof.kurowski@man.poznan.pl, mslysz@man.poznan.pl, marek.subocz@man.poznan.pl

[2] *Poznan University of Technology*
Institute of Computing Science
ul. Piotrowo 2, 60-965 Poznan, Poland
E-mail: rafal.rozycki@cs.put.poznan.pl

Received: 30 March 2021; revised: 14 June 2021; accepted: 28 June 2021; published online: 2 July 2021

This paper was guest edited by Dr. Cezary Mazurek

Abstract: As indicated in various recent research, there may still be challenges in achieving acceptable performance using quantum computers for solving practical problems. Nevertheless, we demonstrate promising results by using the recent advent of the D-Wave Advantage quantum annealer to train and test a Restricted Boltzmann Machine for the well studied MNIST dataset. We compare our new model with some tests executed on the previous D-Wave 2000Q system and show an improved image classification process with a better overall quality. In this paper we discuss how to enhance often time-consuming RBM training processes based on the commonly used Gibbs sampling using an improved version of quantum sampling. In order to prevent overfitting we propose some solutions which help to acquire less probable samples from the distribution by adjusting D-wave control and embedding parameters. Finally, we present various limitations of the existing quantum computing hardware and expected changes on the quantum hardware and software sides which can be adopted for further improvements in the field of machine learning.

Key words: machine learning, RBM training, quantum annealing, D-Wave quantum computer, MNIST dataset

I. INTRODUCTION

Promising theoretical and experimental research indicates using quantum annealing as an alternative approach and a powerful technique for several combinatorial and discrete optimization problems. Quantum annealers are using different quantum phenomena, including tunneling and entanglement. However, in contrast to circuit-based quantum systems, quantum annealers have been successfully designed and implemented to efficiently take advantage of the adiabatic theorem to find a physically realizable Hamiltonian ground state. A few generations of quantum annealer devices that exist today offer physical implementations of a non-trivial size up to 5000 qubits provided by a D-Wave Advantage device. In practice, adiabatic quantum optimization implemented even in the older generation of quantum annealer device, namely D-Wave 2000Q, may provide benefits in solving classically-hard problems using appropriate embed-

ding and heuristic techniques, e.g. for a job-shop scheduling problem as we demonstrated recently in [1]. Also, many machine learning algorithms are trained via solving optimisation problems, in particular minimisation of a cost function. We have observed interesting studies in the area of applying quantum annealing for machine learning, discussed for instance in [2–5]. In this context, a new approach using simulated quantum annealing (SQA) to numerically simulate quantum sampling in a deep Boltzmann machine (DBM) was presented in [6]. The authors proposed a framework for training the network as a quantum Boltzmann machine (QBM) in the presence of a significant transverse field for reinforcement learning. However, they demonstrated that the process of embedding Boltzmann machines in larger quantum annealer architectures is problematic when huge weights and biases are needed to emulate the Boltzmann machine's logical nodes using chains and clusters of physical qubits. On the other hand, quantum annealing has the potential to speed up the sampling process exponentially. Using a quantum annealer to draw representative samples from a Boltzmann distribution could potentially provide an alternative to classical machine learning techniques. Nevertheless, the remaining question is how far we still are from practical use cases and applications as well as what are the current constraints we should consider in particular.

The rest of this paper is organized as follows. Sec. 2 describes our primary motivations to design and implement a new quantum-based sampling for Restricted Boltzmann Machines. Sec. 3 presents the model characteristics and all the parameters, whereas Sec. 4 discusses various extensions and improvements thanks to the access to the latest quantum D-Wave device. Sec. 5 shows a set of experiments and achieved results, and our conclusions are presented in Sec. 6.

II. MOTIVATIONS

Motivated by the recent improvements in quantum annealers offered by D-Wave, we wanted to test its efficiency for a machine learning algorithm using a comprehensive image processing benchmark. Due to the limitation of the existing quantum annealers and environmental effects we focused on Restricted Boltzmann Machines (RBM) as a particular class of unsupervised deep learning commonly used for classification, regression or feature learning [7]. RBM is a well-known probabilistic unsupervised learning model which is learned by an algorithm called Contrastive Divergence. An important step of this algorithm is called Gibbs sampling – a method that returns random samples from a given probability distribution. We decided to conduct our experiments on the popular MNIST dataset considered a standard benchmark in many of the machine learning and image recognition subfields [8]. The resolution of the MNIST images is also perfect for the state-of-the art quantum annealers, as the number of qubits is still limited. An interesting approach was proposed for unsupervised learning based on the implementation of a hybrid classical-quantum architecture in [9]. The authors demonstrated how a D-Wave 2000Q device can be used for the generation of artificial images. For this task, they used a sub-sampled 16×16 pixels version of the standard handwritten digit dataset MNIST.

Our first tests were based on a reference RBM implementation for D-Wave available at [10]. However, it turned out that the original 28×28 resolution was too high for our initial tests. The proposed method only allowed a small number of input values, due to the fact that the whole problem was encoded on the quantum computer. Therefore, we could only test this approach on a scaled-down version of the MNIST data set with the 14×14 resolution which was hardly an acceptable limitation for practical purposes. We redesigned the proposed approach and implemented a new model from scratch based on highly optimised QUBO formulation available at [11]. Consequently, the compression was no longer needed and we achieve much better results as we demonstrate in the next sections.

III. PROBLEM DESCRIPTION

Restricted Boltzmann Machine is a variant of a Boltzman Machine – a stochastic, generative machine learning model inspired by statistical physics. It can model the underlying probability distribution of a training dataset. An RBM is a bipartite graph with two groups of nodes called *visible* and *hidden*. Learning an RBM corresponds to fitting its parameters such that the distribution represented by the RBM models the distribution underlying the training dataset.

There is a weighted connection between each pair of vertices from different layers. A bias value is also associated with each vertex from both layers, see a schematic RBM architecture in Fig. 1.

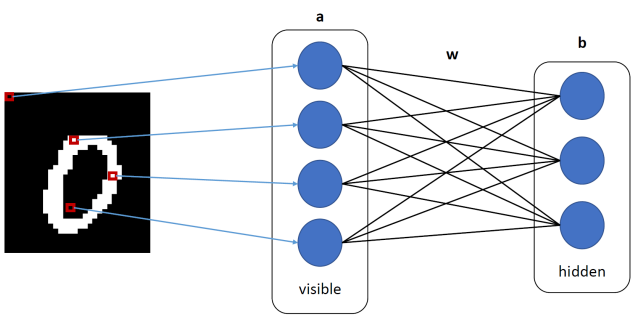

Fig. 1. The schematic representation of a Restircted Boltzman Machine

An RBM is an energy-based, probabilistic model, which means that there is a scalar value assigned to each possible state. The probability of observing a given state depends

on the energy function. The energy function for an RBM is given by the following Eq. (1)

$$E(v,h) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_{i,j} v_i h_j w_{ij} \ . \quad (1)$$

Probability value for a given state (v,h) is described as:

$$p(v,h) = \frac{1}{Z} e^{-E(v,h)} \ , \quad (2)$$

where Z is a partition function which serves as a normalization factor and is generally difficult to compute

$$Z = \sum_{v,h} e^{-E(v,h)} \ . \quad (3)$$

It is hard to compute every possible combination of v and h, as it would require 2^{m+n} operations, where n and m are the lengths of v and h vectors, respectively. Classically, this limitation has been evaded by assuming independence of variables [12, 13]. Under this assumption, given the values from the visible layer (e.g. the training data), much simpler formulas for the conditional probabilities of the hidden layer can be derived:

$$p(h_j = 1|v) = \sigma(b_j + \sum_i w_{ij} v_i) \ , \quad (4)$$

where σ denotes the sigmoid function. Respectively, it can be shown for the visible layer:

$$p(v_i = 1|h) = \sigma(a_i + \sum_j w_{ij} h_j) \ . \quad (5)$$

The weights w as well as the biases a and b are parameters subject to learning. The negative log likelihood function seems to be the natural choice for the cost function, and the goal of maximum likelihood is to find the parameter values that give the distribution that maximise the probability of observing the data. Thus, in each iteration t we want to maximize the probability that each pixel in the generated vector v_t is equal to the training example x

$$L(x) = \frac{1}{T} \sum_t^T - \log p(v_t = x) \ . \quad (6)$$

If learning is using a gradient-based method such as Stochastic Gradient Descent (SGD), it is necessary to calculate the gradient of the loss function with respect to the parameters. It can be shown that the gradient consists of two terms called positive and negative phases of the gradient

$$\frac{\partial(-\log p(v_t))}{\partial\theta} = \mathbb{E}_h \left[\frac{\partial E(v_t,h)}{\partial\theta} \middle| v_t \right] - \mathbb{E}_{v,h} \left[\frac{\partial E(v,h)}{\partial\theta} \right] \ . \quad (7)$$

The first part of the gradient can be derived as $-h \cdot v^T$. However, the second part of the gradient is again difficult to compute. To deal with this issue we use the Gibbs Sampling method in order to estimate it. From input v_t we can sample values h_t from the hidden layer as in Eq. (4). Then, given the vector h_t we can again sample a new vector v'_t on the visible layer according to Eq. (5). After repeating this process k times we get a random sample from the given distribution. It can be shown that a good estimate of the negative part of the gradient can be denoted as $-h_t^{(k)} \cdot v_t^{(k)^T}$. In practice good results can be achieved for just $k = 1$. Finally, the update rule for the algorithm can be written as shown in Eq. (8):

$$w_{t+1} = w_t + \alpha \left(h_t \cdot v_t^T - h'_t \cdot v_t'^T \right) \ , \quad (8)$$

where α is the learning rate. Analogously the update rules for biases can be derived as denoted in Eqs. (9) and (10)

$$a_{t+1} = a_t + \alpha \left(v_t^T - v_t'^T \right) \ , \quad (9)$$

$$b_{t+1} = b_t + \alpha \left(h_t^T - h_t'^T \right) \ . \quad (10)$$

This learning algorithm is called Contrastive Divergence (CD) and is commonly used to train an RBM model. The process is visualised in Fig. 2. It can be further enhanced by adding solutions typical of some other popular learning methods, such as learning rate decay or gradient momentum, which might have a positive impact on the learning process.

IV. QUANTUM EXTENSIONS TO RBM

RBM is an energy based model with an energy function described in Eq. (1). The D-wave's Quantum Annealing algorithm provides an efficient way to solve such a function. In order to do that, a QUBO (Quadratic Unconstrained Binary Optimization) Eq. (11) needs to be constructed:

$$f(x) = \sum_i Q_{i,i} x_i + \sum_{i<j} Q_{i,j} x_i x_j \ . \quad (11)$$

In the next step, the QUBO representation is transformed into a *Binary Quadratic Model* which can be used by D-wave's quantum sampler in order to obtain samples from the model's actual probability distribution.

With an energy-based sampler it is possible to obtain accurate samples from the original distribution, which is computationally expensive for a classical computer. However, according to the latest experimental studies there are still some technical challenges related to the constriction of the D-Wave annealer which affects the quality of sampling from a Boltzmann distribution as indicated in [14].

It is worth noting that it can be done without making additional assumptions about the independence of variables, which in general does not have to be true. Instead of classical sampling as described in Eqs. (4) and (5), we get the samples as a result of a quantum annealing process. The only input we need to encode for the quantum annealer are the

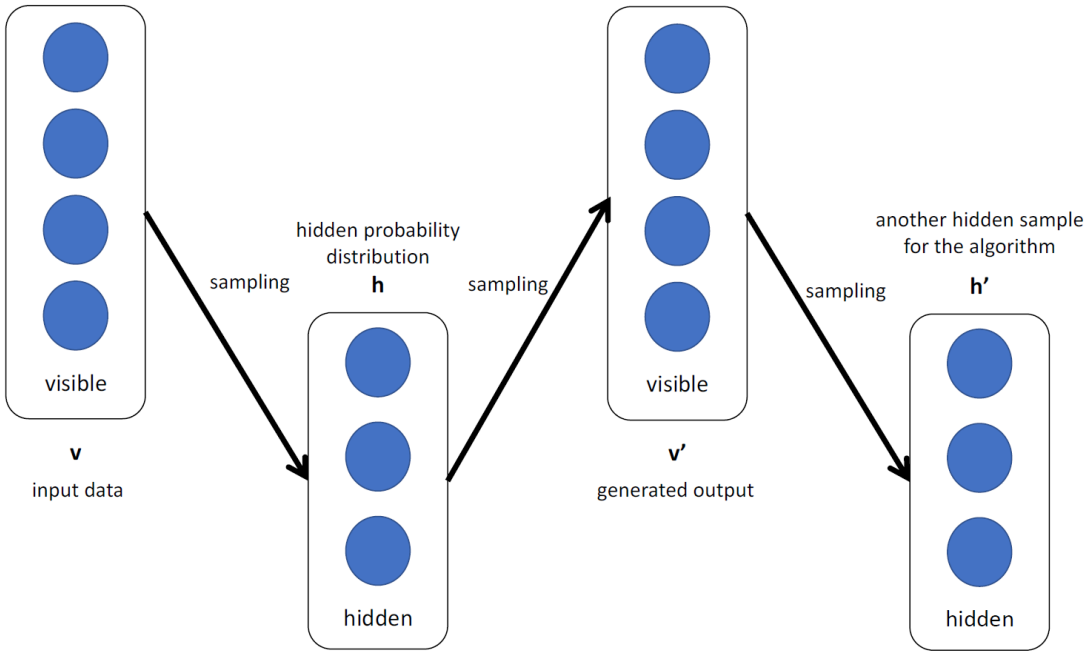

Fig. 2. The schematic representation of the Contrastive Divergence algorithm

model parameters. They are given as a Hamiltonian equation for the problem. Sampling from the opposite layer requires a specific definition of a Hamiltonian, corresponding to the model parameters and input variables. For sampling the hidden layer, given an input vector v consisting of binary variables $v_{1...n}$, we define the problem as follows. For each $v_i = 1$ a term is added to the equation as in Eq. (12), with $h_{1...m}$ set as solvable parameters

$$\mathcal{H} = -\sum_{j}^{m}\left(\left(\sum_{i}^{n}(w_{i,j}\cdot v_i)+b_j\right)\cdot h_j\right) . \qquad (12)$$

Analogously, given a binary input vector h of length m, for each $h_j = 1$ we add a term to the Hamiltonian as in Eq. (13), setting $v_{1...n}$ as solvable parameters

$$\mathcal{H} = -\sum_{i}^{n}\left(\left(\sum_{j}^{m}(w_{i,j}\cdot h_j)+a_i\right)\cdot v_i\right) . \qquad (13)$$

V. EXPERIMENTS

V. 1. Basic Experiments

We created a new RBM model in Python and replaced the classical sampling steps by the output of Dwave's quantum sampling function. We tested our model on the well-known MNIST dataset [8]. MNIST is a dataset of handwritten digit images with 60 000 training samples and 10 000 testing samples. It is a popular machine learning benchmark dataset which consists of images of size 28×28 pixels each.

Before feeding the data to the model, we had to perform some pre-processing steps due to the fact that an RBM can only process binary data. We transformed the original MNIST images to binary representation using a threshold of 100 pixel brightness out of $0 \div 255$ greyscale as shown in Fig. 3. Next, we had to flatten the images so they could fit onto the 1-dimensional RBM visible layer of length $28 \times 28 = 784$.

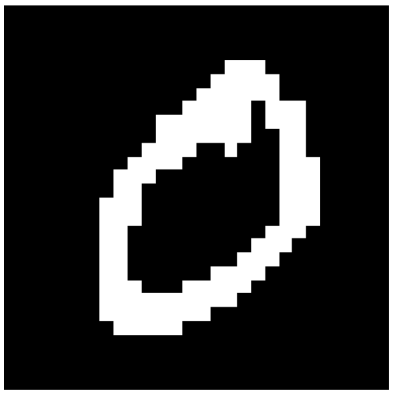

Fig. 3. The example image of digit '0' after the basic pre-processing phase

To monitor the quality of training processes we used a popular MSE (Mean Squared Error) measure between the input image which activates the sampling process and the generated image. The two images were compared pixel by pixel and the MSE value was then normalized by dividing it by the image size to fit between 0 and 1. For testing purposes

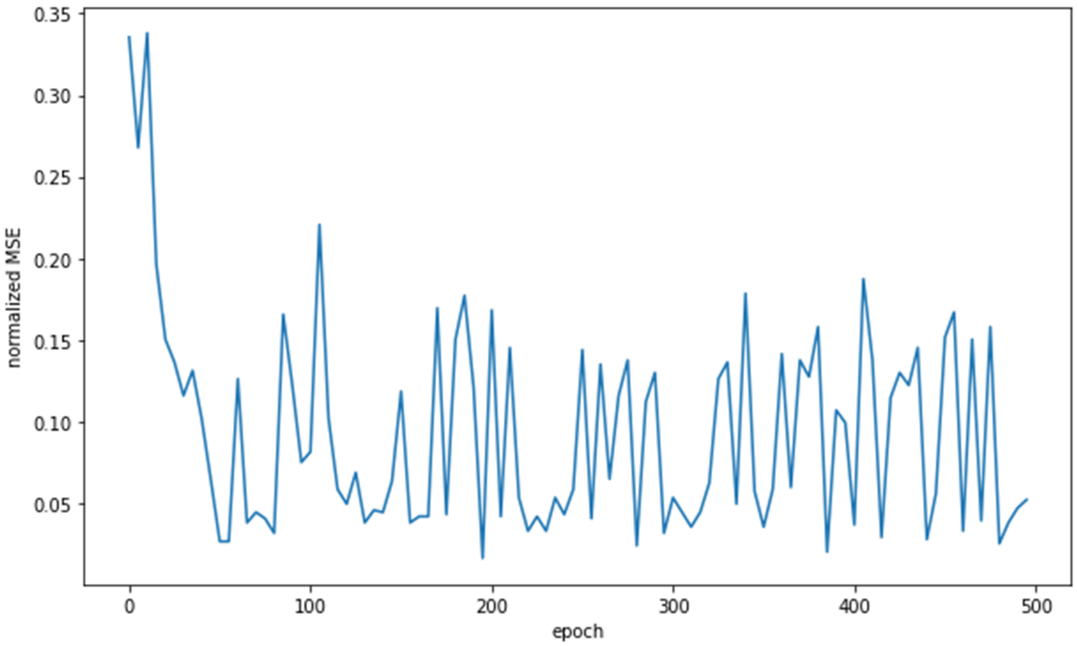

Fig. 4. Normalised MSE values changing with an increasing number of epochs for a two digit problem

we compared the output image with an image of the digit constructed by averaging over the whole dataset. As the measure does not exactly represent the quality of the output, which may be subjective, it tends to go lower with a decreasing loss function during the learning process. An exemplary plot of this value can be seen in Fig 4. The plot has an L-shaped curve, typical for machine learning problems, which drops rapidly at the beginning of the learning process and then gradually decreases in the remaining epochs. However, there are some fluctuations because the MSE is not an exact measure of the quality of the model.

We started to train our model starting from a single digit case and processed only images of digit '0'. After training the model for 500 epochs the model was able to generate fairly accurate recreation of the digit. An exemplary result can be seen in Fig. 5.

Then, we gradually added more digits to the training data to make the problem harder for the model.

V. 2. Chain Strength

The chain strength is a crucial control parameter available in D-Wave architecture responsible for ensuring that results follow given restrictions. When a chain connects two qubits, they are supposed to have the same binary value. If the opposite is true, the chain is broken, which may lead to suboptimal results. However, all QUBO weights are autoscaled to values between -1 and 1 together with the chain strength value. As the chain strength gets larger, the QUBO weights representing the original problem may shrink to near-zero values, decreasing their importance. This means that increasing the chain strength may result in a more distributed range of solutions, as original punishments become less precise and more prone to thermal noise. Usually, this effect would be undesirable, leading to a lower probability of obtaining an optimal solution. However, the RBM uses quantum annealer for sampling from a range of possible solutions. When more generalization is needed (e.g. while learning to produce images of more than one digit from the same architecture), the broader distribution of solutions is a positive phenomenon.

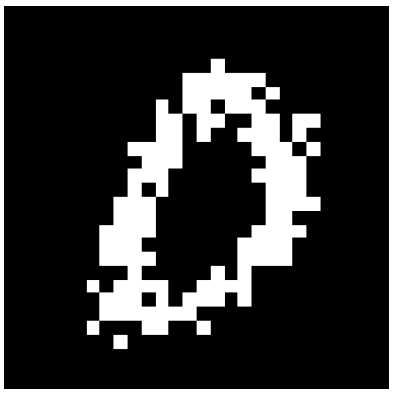

Fig. 5. The example image of digit '0' generated by the model with normalized MSE $= 0.067$

The phenomenon described in Sec. 5.2 is observable in Fig. 6. When trained on a low number of digits, bqm yields the best results with a low chain strength value. When the number of digits goes up it turned out that the higher chain strength value is much better.

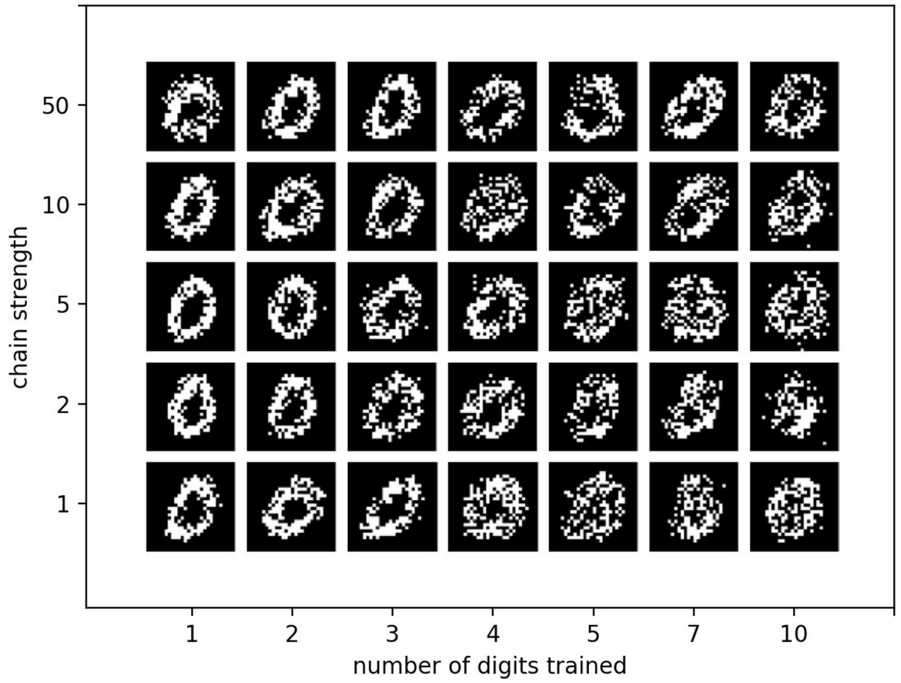

Fig. 6. The influence of chain strength and number of digits on the overall pictures quality generated by the RBM

We also wanted to discover the performance of the quantum sampling process. We run the algorithm on both available quantum annealers (Advantage and 2000Q lower-noise) and received similar execution times. QPU annealing on average takes 30 μs; sampling takes avg. 200 μs. The post-processing takes avg. 350 μs, and readout operations take avg. 150 μs. The overall connectivity cost to D-wave's API was also minimized in the code. Consequently, our implementation provides a significant speedup over a classical approach and can also be used as an alternative benchmark for the RBM supported by quantum annealing devices. Nevertheless, in our opinion there are some additional improvements possible but have not been investigated yet, in particular concerning coupling and biases scaling.

V. 3. Hidden Layer Size

Another model hyperparameter that we tested was the size of the hidden layer. We tested how the generated images change for hidden layer sizes from 10 to 100 units increasing by 10 for a two digit problem (digits '0' and '1'). A plot consisting visualizations of generated images of the same '0' digit from the test set with corresponding MSE score is shown in Fig. 7.

For most of the experiments we set the hidden layer size to 60 units, which was enough for the network to learn efficiently and the training was still fast enough.

V. 4. Additional Extensions

In order to further enhance the model we implemented some additional improvements to the learning algorithm. Firstly we added learning rate decay, which is a mechanism that decreases the learning rate α every couple of iterations. It is used to further optimize the learning process after getting stuck close to the optimum.

Then we added Momentum, which is a popular improvement of gradient based algorithms, such as SGD or Adam. The main idea behind the method is to calculate the gradient, not only based on the current data sample, but also the previous gradient direction. The new parameter update rule is denoted as follows:

$$w_{t+1} = w_t + \alpha \cdot V_t \ , \tag{14}$$

where Vt is the velocity calculated from the momentum algorithm

$$V_t = \gamma \cdot V_{t-1} + \left(h_t \cdot v_t^T - h_t' \cdot v_t'^T\right) \ . \tag{15}$$

One can observe that for $\gamma \in [0; 1]$ each gradient term from previous iterations is added with a smaller weight. A typical value for γ is usually around 0.9. This method should help the learning algorithm to get out of saddle points easily and prevent overfitting.

Another improvement that was added is a different variant of the learning algorithm called persistent CD. The main difference is that in most iterations, instead of taking the input data as vector v, we use the v' vector from the previous iteration to calculate the negative phase of the gradient. The reasoning behind this is that in a small number of iterations the model changes only slightly so we could effectively reuse the previous samples to simulate executing more

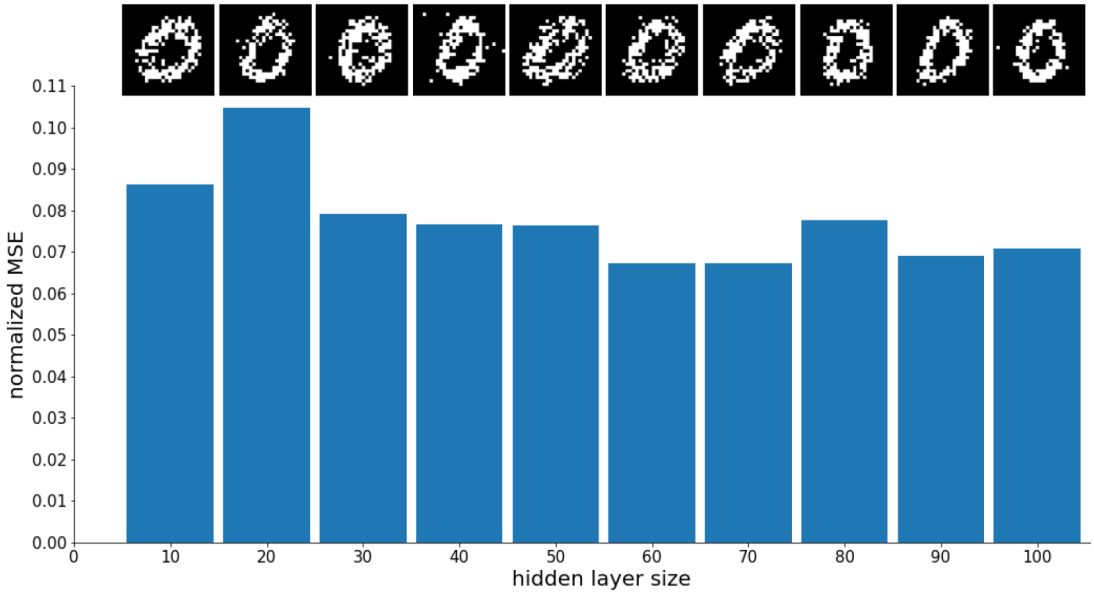

Fig. 7. The example dependence between the hidden layer size and the quality of the generated images (normalised MSE)

Gibbs sampling steps, as for $k > 1$. We had to use the v vector from the training every bs iterations to run the algorithm. For classical case bs it is recommended to be the batch size; however, in our implementation without the mini-batch approach it can be set to any value. The persistent CD is used to be able to escape from local optima, as for $k = 1$ the sampling result for the negative gradient phase is usually very close to the original input data, henceforth prevent overfitting.

Additionally, to compare our results with a classical version of RBM we used the RBM algorithm implemented in *scikit-learn* library [15] with the same training dataset and hyperparameters, see the example digit generated in Fig. 8.

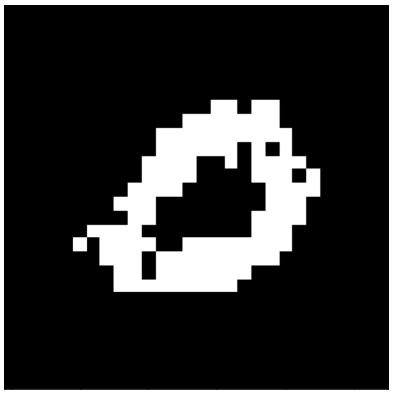

Fig. 8. The example result of digit '0' generated by a classical RBM from sklearn library with normalised MSE $= 0.039$

VI. CONCLUSIONS

We investigated and compared the approach for training the RBM that uses quantum sampling from two generations of D-Wave quantum annealers. We demonstrated experimentally that the latest D-Wave Advantage QPU architecture consisting of more than 5000 qubits and 35 000 couplers was good enough to compensate for limited qubit connectivity and a noisy previous generation environment. Our original contribution was to implement and run our experiments to successfully train the RBM using a quantum annealer in the D-Wave Advantage for the original MNIST dataset with 28×28 pixels resolution. Moreover, we estimated quantum sampling runtime to show the efficiency of QPU versus classical systems.

We also tested many algorithm hyperparameters and indicated that one of the most important aspects to consider is the size of the RBM. The visible layer size was set to 784, as it must be compatible with the output, but the hidden layer size can be changed and, according to our experiments, may impact the quality of the model. However, the time of learning the model grows significantly with the hidden layer's size because of the increasing number of variables in each sampling step. Thus, we empirically set 50 as a reasonable hidden layer size for the remaining experiments.

The model has also been tested on a different number of digits in the training dataset. For a single digit it was easy to efficiently train the model as it even tended to overfit. This was more of a problem for a larger number of digits. To prevent overfitting we proposed a number of solutions. One of them was tuning the chain strength parameter, which helped to acquire less probable samples from the distribution because the quantum annealer tended to find the global optimum (e.g. the most probable sample) too often. We also added some classical extensions to the learning algorithm

such as learning rate decay, momentum and persistent CD variant of the learning algorithm.

Acknowledgment

This research has been partially supported by the statutory funds of Poznan University of Technology and Poznan Supercomputing Networking Center affiliated to the Institute of Bioorganic Chemistry, Polish Academy of Sciences as well as the grant PRACE-LAB2 POIR.04.02.00-00-C003/19.

References

[1] K. Kurowski, J. Weglarz, M. Subocz, R. Rozycki, G. Waligóra, *Hybrid Quantum Annealing Heuristic Method for Solving Job Shop Scheduling Problem*, [In:] *Computational Science – ICCS 2020. Lecture Notes in Computer Science* **12142**, Eds. V.V. Krzhizhanovskaya, G. Závodszky, M.H. Lees, J.J. Dongarra, P.M.A. Sloot, S. Brissos, J. Teixeira, Springer, Cham (2020).

[2] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, S. Lloyd, *Quantum machine learning*, Nature **549**, 195–202 (2017).

[3] S.H. Adachi, P. Maxwell, *Henderson Application of quantum annealing to training of deep neural networks*, arXiv: 1510.06356 (2015).

[4] S. Lloyd, M. Mohseni, P. Rebentrost, *Quantum algorithms for supervised and unsupervised machine learning*, arXiv: 1307.0411 (2013).

[5] N. Wiebe, A. Kapoor, K.M. Svore, *Quantum Deep Learning*, arXiv: 1412.3489 (2015).

[6] D. Crawford, A. Levit, N. Ghadermarzy, J.S. Oberoi, P. Ronagh, *Reinforcement Learning Using Quantum Boltzmann Machines*, arXiv: 1612.05695 (2016).

[7] P. Smolensky, *Chapter 6: Information Processing in Dynamical Systems: Foundations of Harmony Theory*, [In:] *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* **1**: Foundations, Eds. D.E. Rumelhart, J.L. McLelland, MIT Press, 194–281 (1986).

[8] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, *Gradient-Based Learning Applied to Document Recognition*, Proceedings of the IEEE **86**(11), 2278–2324 (1998).

[9] M. Benedetti, J. Realpe-Gomez, A. Perdomo-Ortiz, *Quantum-assisted helmholtz machines: a quantum-classical deep learning framework for industrial datasets in near-term devices*, arXiv: 1708.09784 (2017).

[10] S. Ni, S. Nagayama, *Performance comparison on cfrbm between gpu and quantum annealing*, Technical report, Mercari (2018).

[11] *Quantum annealing based RBM*, https://github.com/marek subocz/QRBM.

[12] G.E. Hinton, S. Osindero, Y.W. Teh, *A fast learning algorithm for deep belief nets*, Neural Comput. **18**(7), 1527–1554 (2006).

[13] T. Tieleman, *Training Restricted Boltzmann Machines using Approximations to the Likelihood Gradient*, Proceedings of the 25$^{\text{th}}$ international conference on Machine learning, 1064–1071 (2008).

[14] J. Brugger, Ch. Seidel, M. Streif, F. Wudarski, Ch. Dittel, A. Buchleitner, *Output statistics of quantum annealers with disorder*, arXiv: 1808.06817 (2021).

[15] *Scikit-learn documentation*, https://scikit-learn.org/stable/modules/generated/sklearn.neuralnetwork.BernoulliRBM.html.

Krzysztof Kurowski is the Technical Director of Poznan Supercomputing and Networking Center. He graduated from Poznan University of Technology and holds the PhD degree with habilitation in Computer Science. Since 2008 he has been leading the Applications Department at Poznan Supercomputing and Networking Center, Poland. He has been actively involved in many R&D projects related to Information and Communication technologies at the national, European and international levels. He was a research visitor at University of Queensland, Argonne National Lab and University of Southern California. His research activities have been focused on the advanced parallel simulations, scheduling and resource management in networked and distributed computing environments. The results of his research have been successfully published in highly ranked scientific and journal papers. Recently, he has also been active in the following research domains: power-efficient application modelling, multi-scale simulations, advanced visualization with Human-Machine interactions, and massive e-learning technologies.

Mateusz Slysz is a final-year student of Computer Science at Poznan University of Technology. Previously, he graduated with an engineering degree in this field and also obtained a bachelor's degree in Physics from the Adam Mickiewicz University in Poznan. Since 2020 he has been a member of a research group at Poznan Supercomputing and Networking Center, where he is working on machine learning algorithms for quantum computers, which allows him to combine his two scientific passions. During his work and studies he conducted research about quantum models such as qRBM, qGAN and quantum-based portfolio optimisation. In his private life he is a big football fan and board game enthusiast.

Marek Subocz graduated from Poznan University of Technology with a Bachelor of Engineering title in 2021. Since 2019 he has been working as a research engineer at Poznan Supercomputing and Networking Center, where he focuses on practical applications for quantum annealing, particularly regarding the job shop scheduling problem, as well as quantum-gates-based implementations of various neural network structures. Apart from computer science he is interested in volleyball and psychology.

Rafał Różycki PhD, DSc, graduated from Poznan University of Technology in Computing Science in July 1994. In 2000 he received his PhD degree in Computing Science from Poznan University of Technology. He published his habilitation thesis on scheduling computational jobs with respect to energy constraints in 2014. Since 2020 he has been heading the Laboratory of Operational Research and Artificial Intelligence. Professor at the Institute of Computing Science, Poznan University of Technology. His main areas of interest include project and machine scheduling, discrete-continuous scheduling, energy-aware scheduling, combinatorial optimization, metaheuristic algorithms. He has authored or co-authored over 90 scientific publications in international journals, monographs and conference proceedings, and presented his results during over 50 domestic and international scientific conferences and workshops.