Quantum Annealing Assisted Deep Learning for Lung Cancer Detection

Andrew Ward
Stanford University
450 Serra Mall, Stanford, CA
atward@stanford.edu

Nicholas Bambos[1]
Stanford University
450 Serra Mall, Stanford, CA
bambos@stanford.edu

Abstract

Lung cancer is a leading cause of death worldwide, and life-threatening misdiagnoses are prevalent. In order to assist in diagnosis and prognosis, promising machine learning and deep learning techniques have been proposed, but these are computationally intensive and inaccurate. The goal of my research is to use a new technology, quantum annealing, to efficiently and accurately determine a diagnosis and prognosis from scans of potentially cancerous lungs. I seek to leverage the recent success of deep learning image recognition along with the exponential computational improvements shown in quantum annealing to develop an efficient classification pipeline for potentially cancerous lung tissue images.

1. Introduction

Lung cancer is the deadliest form of cancer worldwide, accounting for almost one fifth of all cancer deaths [1] and a predicted 115,870 deaths in the US in 2017 [2]. Furthermore, lung cancer is often misdiagnosed, leading to a rapid decrease in life expectancy or expensive and harmful treatments. Promising work has recently attempted to assist and improve lung cancer diagnosis and prognosis using deep learning and other machine learning techniques [3] [4]. However, these methods are constrained by computational resources and poor scalability. For example, Yu et al. recently determined lung cancer prognosis by using machine learning algorithms on histopathology images from cancer patients, but features were only extracted from 3% of each image [4]. New techniques are needed to look more holistically at the input data and improve prediction accuracy.

Quantum computing promises exponential improvements for many optimization problems. D-Wave Systems has developed a novel processor that performs a process known as quantum annealing, which has achieved a $\sim 10^8$ processing time improvement over classical systems when solving problems related to quantum behavior [5]. Additionally, the D-Wave architecture has been used to demonstrate simple deep learning applications [6] [7] [8]. However, although deep learning has been demonstrated on the D-Wave, and deep learning has been shown to be promising for lung cancer research, no biological studies have been done with D-Wave. Achieving a similar performance improvement in lung cancer research could revolutionize the diagnosis and prognosis of lung cancer, as well as make these methods readily accessible to health facilities worldwide.

The goal of this work is to determine the potential improvements that quantum annealing, used jointly with deep learning, can offer to lung cancer image classification and diagnosis. In this paper, I empirically investigate how quantum annealing can be applied to aid with a simple image classification problem. Specifically, I consider deep belief network architectures, because the variable updates for these architectures have previously been computed using the D-Wave quantum annealer [7]. I compare network training using the classical algorithm of contrastive divergence, and training that uses a simulated result comparable to results using quantum annealing.

The remainder of the paper is organized as follows. In Section 2, I discuss some work related to this study. In Section 3, I provide some background on Deep Belief Networks and quantum annealing. In Section 4, I present the methods used in this project. Section 5 discusses the data sources used in this project. Section 6 presents a numerical performance evaluation study that compares simulated quantum and classical heuristics. Finally, I conclude and discuss ongoing and future work in Section 6.

2. Related Work

A number of studies have previously used deep learning and machine learning to analyze lung cancer images. In order to properly diagnose lung cancer, two stages of image classification need to take place. First, a patient is given a computed tomography (CT) scan, and the resulting images are then viewed to see if any cists or tumors are present in the patient. If a nodule is identified as potentially cancerous, a biopsy of the nodule is surgically taken from the patient and stained with hematoxylin and eosin (H&E). A radiologist then examines the cells from the biopsy under a microscope, and makes either a cancerous or non-cancerous diagnosis.

[1]Not affiliated with CS 231N

Studies have attempted to improve both stages of this pipeline using deep learning. In 2016, Li et al. [3] used a convolutional neural network to identify lung nodules from CT scans, and achieved an accuracy of 86.4% with a sensitivity of 89.0%. Also in 2016, Yu et al. [4] created a fully automated pipeline to extract image features from H&E-stained pathology images and run an ensemble of machine learning algorithms (Naïve Bayes, Random Forests, SVMs) to distinguish between two types of lung cancers: adenocarcinoma and squamous cell carcinoma. They achieved an AUC of 0.85.

Separately, D-Wave Systems has developed a machine that leverages the quantum properties of superconductors to perform quantum annealing. In 2011, D-Wave Systems made quantum annealing technology commercially available, and from 2011 to 2017, D-Wave has scaled the number of qubits (quantum bits) contained on their processor from 128 to over 2000. However, there remains some controversy over the extent to which the D-Wave's success can be attributed to quantum effects [9]. Nevertheless, D-Wave's quantum processing unit (QPU) has demonstrated a ~10^8 processing time improvement over classical systems when solving problems related to quantum behavior [5]. However, these problems were specifically chosen to be easy to solve using quantum annealing, and very difficult to solve using comparable classical methods. More generally, the D-Wave quantum annealer has been shown to offer an improvement of up to a factor of ~10^3 on a more useful class of real-world problems when compared to classical heuristic solvers [10].

A number of studies have used quantum annealing in conjunction with image processing. O'Malley et al. [8] compared quantum annealing to classical methods in a study that attempted to reconstruct faces using pre-learned facial features. They were able to analyze large datasets (2,400 facial images) using quantum methods, although the low connectivity of the D-Wave QPU limited the number of pre-learned features that could be extracted and considered during reconstruction.

In 2015, Adachi and Henderson [7] used the D-Wave architecture to identify handwritten numbers. This study used the well-understood MNIST dataset [11], but scaled the 28-by-28 images down to blurry 6-by-6 grayscale images so that they would completely fit on the quantum annealer. The study used deep belief networks to identify these 6x6 images, and compared classical algorithms with quantum annealing in the generative training phase of the DBN. My paper attempts to recreate this study, with the intention of eventually increasing both the types of architectures on which quantum annealing can offer an improvement, and the size of the input images (to be able to consider lung cancer images, for which a similar scale-down approach would not be feasible).

3. Background

In this section, I will discuss Deep Belief Networks and quantum annealing, both of which are integral to the project.

3.1. Deep Belief Network

A Deep Belief Network consists of multiple stacked Restricted Boltzmann Machines (RBMs), so that the output of one RBM is the input of the next. Each RBM has a set of visible nodes v_i, a set of hidden nodes h_j, and a set of weights w_{ij} that connects all visible nodes to hidden nodes. In this way, an RBM is similar to a fully connected layer of a neural network. However, in the most commonly used Bernoulli-Bernoulli RBM, each of v_i and h_j can only take on binary values. An example DBN is shown in Figure 1. The weights connecting the nodes, as well as the biases on all of the visible and hidden nodes, though, are continuous. So, in order to obtain values for hidden nodes given values for the visible nodes, values are sampled from a joint probability distribution determined by the visible nodes, weights, and biases.

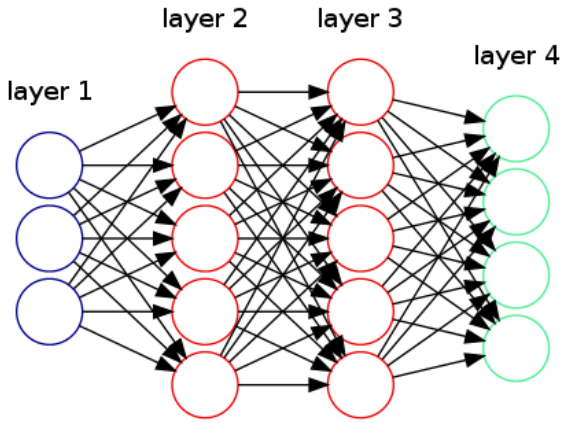

Figure 1: Deep Belief Network architecture. The input layer and hidden layers all have 36 nodes, while the output layer has 10 nodes.

Training of Deep Belief Networks consists of two stages: the generative training phase, which is unsupervised, and the discriminative training phase, which is supervised. The unsupervised training is done for each RBM separately; that is, the weights and biases are first learned in an unsupervised way for the first RBM in the network only. Once these have converged, or a specified number of "pre-training" iterations have elapsed, the second RBM's weights and biases are trained. This unsupervised process is the stage for which quantum annealing has been shown to offer a potential advantage.

The true update to the weights in the model is given by

$$w_{ij}^{t+1} = \alpha w_{ij}^{t} + \varepsilon\left(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model}\right), \quad (1)$$

where α is the momentum, and ε is the learning rate, and $\langle v_i h_j \rangle$ denotes the expected value of the product of v_i and h_j. The bias update formula is similar. The first term is easy to estimate, since it is calculated with a specific input data vector. However, the second term is the expected value over the entire dataset, and is very computationally costly to calculate.

Typically, the update is instead performed using contrastive divergence, which calculates a probability distribution for all of the hidden nodes, and then samples binary values for the hidden nodes based on that distribution. Then, with these values for the hidden variables, a probability distribution of the visible nodes is calculated, and then binary values are sampled from the probability distribution of the visible nodes. The update formula then becomes

$$w_{ij}^{t+1} = \alpha w_{ij}^{t} + \varepsilon\left(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{CD-n}\right), \quad (2)$$

where *CD-n* represents *n* iterations of contrastive divergence (sampling the hidden nodes, locking them to that value, and sampling the visible nodes). In practice, *CD-1* is often used because it is fast and works surprisingly well. However, this method has many cases where it converges slowly, or gets stuck in local optima.

This unsupervised learning provides a way for the network to potentially learn features from the input data without any labels. Once each RBM layer of the DBN has finished with the generative training phase, the supervised training begins. For this, the hidden variables of the last RBM are locked to the value of the output labels, and then traditional backpropagation is used to get gradient updates for all of the weights and biases in the entire network. This continues for a specified number of "post-training" iterations.

3.2. Quantum Annealing

Now I will discuss quantum annealing, the optimization heuristic that runs on the D-Wave QPU.

Problems on the D-Wave machine are represented using an Ising model formulation. This model, first used to simulate magnetism in statistical mechanics, consists of discrete "spin" variables $s_i \in \{-1,1\}$.

These variables determine the energy of the system,

$$E(\boldsymbol{s}) = \boldsymbol{s}^T \boldsymbol{J} \boldsymbol{s} - \mu \boldsymbol{h}^T \boldsymbol{s} \quad (3)$$

where $\boldsymbol{h}_i$ is the weight associated with site *i*, and $\boldsymbol{J}_{ij}$ is the coupling strength between sites *i* and *j*. The D-Wave processor then seeks to minimize the energy of this system, heuristically solving the problem

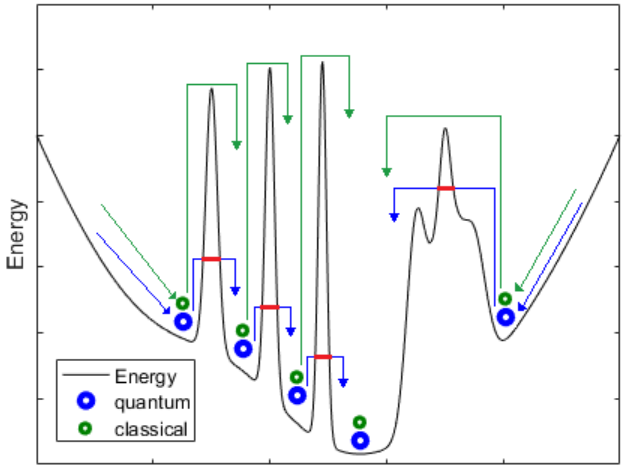

Figure 2: Typical paths of classical and quantum annealing through the search space. The problems shown are attempting to find the minimum energy (objective function) value. In order to pass over an energy peak, a classical algorithm such as simulated annealing must have an energy value higher than the peak in order to explore further. However, quantum annealing can tunnel through tall, narrow energy peaks with a low amount of energy. Tunneling is shown here with a red line. So, if approaching this energy landscape from the left, the quantum algorithm would be much more likely to reach the minimum than the classical algorithm would, but if the problems approached this landscape from the right, quantum annealing would not offer a noteworthy advantage.

$$\boldsymbol{min}_{\boldsymbol{s}}\ \boldsymbol{s}^T \boldsymbol{J} \boldsymbol{s} - \mu \boldsymbol{h}^T \boldsymbol{s} \quad (4)$$
$$\text{subject to } \boldsymbol{s} \in \{-1,1\}^n$$

This problem can be trivially transformed to have $\boldsymbol{s} \in \{0,1\}^n$, so that the variables more closely align with RBMs.

To solve (4), the D-Wave QPU uses quantum annealing, an algorithm that takes advantage of the quantum nature of the processor hardware. Quantum annealing utilizes *quantum tunneling* to more efficiently search through highly non-convex solution spaces. If the energy landscape (defined by (3)), has tall, narrow peaks, then the variables of the problem (represented by qubits) will be able to tunnel through these peaks to continue the search for the global minimum energy. We know from quantum mechanics that a quantum particle, such as an electron, will be able to tunnel through finite-energy barriers with probability inversely proportional to an exponential function of both the height of the barrier and the width of the barrier. So, if *h* is the energy barrier height and *w* is its width, then

$$P(\text{tunnel}) \propto e^{-w\sqrt{h}}. \quad (5)$$

This finite probability is extremely helpful for exploring very rugged search spaces. In classical algorithms, there is

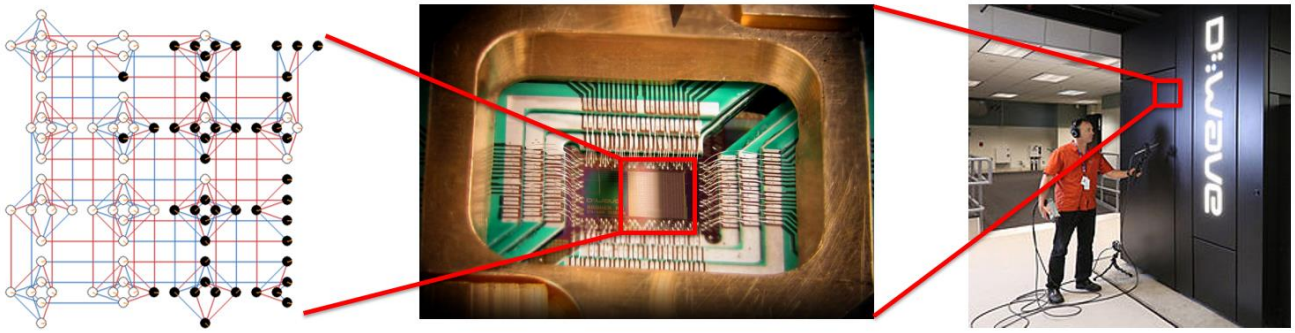

Figure 3: Left: Architectural layout of the D-Wave processor. Circles represent qubit sites, with black and white representing spins of -1 and +1. A line between two sites *i* and *j* represents a potential nonzero interaction $\boldsymbol{J}_{ij}$ term between the sites. Each subgraph should have 8 qubits, but due to fabrication defects, some qubit sites are unusable. Each qubit is coupled to a maximum of 6 neighboring qubits. Middle: D-Wave quantum processing unit (QPU). The chip operates at 10-15 millikelvin, a regime where superconducting effects can be utilized to perform quantum computation. Also pictured are wires which read out values of the qubits after each anneal. Right: exterior shielding for D-Wave machine. The shielding serves to isolate the sensitive qubits from environmental noise and to house all of the equipment needed to cool the QPU to 10-15 millikelvin.

zero probability of going through a place of higher energy to get to a different place of lower energy; the only way to achieve this is to give the entire system more energy, and if that is done, the path through the search space has a high probability of going away from the global minimum.

Figure 2 illustrates this point. If approaching this one-dimensional landscape from the left, quantum annealing would have a much higher probability of reaching the global minimum than a classical algorithm would. However, if approaching from the right, the energy barrier is wide enough that there is a negligible chance of quantum tunneling through the base, so quantum annealing would perform comparably to, for example, simulated annealing.

3.3. D-Wave Hardware

In order to understand how quantum annealing runs on the D-Wave QPU in detail, we need to understand the D-Wave hardware.

The current generation of D-Wave QPU, the D-Wave 2X, contains 1152 qubit sites, each of which can represent one s_i variable. Not every two qubits are connected; each qubit is connected to at most 6 of its neighbors, resulting in 3360 total couplers. The D-Wave architecture is laid out in a graph structure known as a Chimera graph, and is shown in Figure 3 (left). However, the quantum processor is extremely sensitive, and due to fabrication defects, the processor used in this study only had 1135 working qubits and 3265 working couplers.

Equation (4) represents the energy minimization of the multi-qubit quantum system on the D-Wave processor. The processor runs quantum annealing in order to heuristically find $\boldsymbol{s}$ such that the energy is minimized. The quantum annealing steps the processor runs are as follows:

1. The coupling strengths and qubit weights are loaded onto the processor.
2. The system is prepared is a quantum superposition of all possible answers. If an answer was read out at this stage, it would be equivalent to selecting an answer out of all possible solutions uniformly at random.
3. The system gradually evolves from this initial superposition state (with no problem-specific constraints) to the final state specified by the $\boldsymbol{h}$ and $\boldsymbol{J}$ loaded into the system. As the coupling strengths and qubit weights gradually increase/decrease to the desired levels, the probability of reading out each state also changes, so that it becomes more likely to read out lower-energy states and less likely to read out high-energy states. This step is analogous to gradually lowering the temperature when running a simulated annealing algorithm.
4. An answer is read out of the system, with the qubits representing the variables $\boldsymbol{s}$. The longer the time allowed for the previous step, the more likely the global minimum is reached.

However, it is not possible to run arbitrarily large problems in this manner. As mentioned above, each qubit is only connected to a maximum of 6 other qubits. This means that only 6 off-diagonal elements of $\boldsymbol{J}$ can be nonzero per row, regardless of the dimension of $\boldsymbol{J}$.

In an image classification problem, this would mean that fewer than 6 pixels can be considered at once on the D-Wave when using a DBN model. Thus, an additional step is necessary. An RBM of larger size can be graphically embedded onto the D-Wave architecture. Embedding a problem provides a mapping from physical qubits on the D-Wave machine to logical variables in the problem of interest; often, multiple physical qubits are strongly coupled together and collectively represent one logical variable.

Finding an embedding can be done heuristically, and, for fully connected graphs, it only has to be done once per

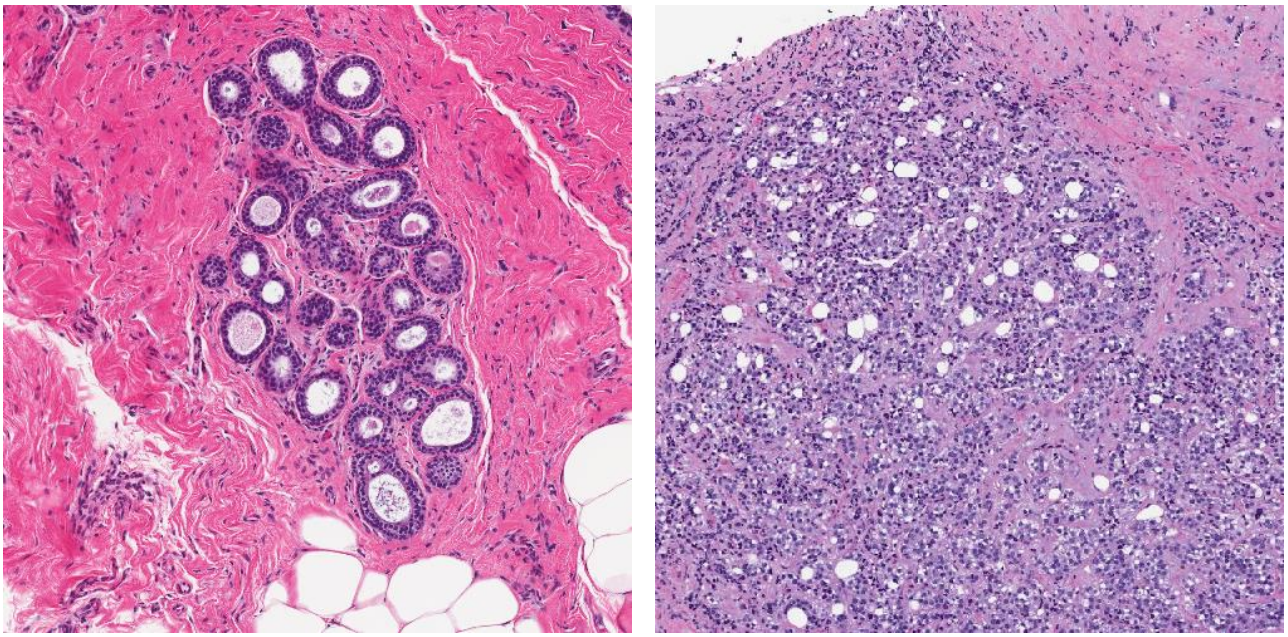

Figure 4: Example hematoxylin and eosin stained slides that show non-cancerous lung tissue (left) and cancerous lung tissue (right)

problem size. In other words, once we have an embedding for, for example, a 36x36 bipartite graph, we can solve any 36x36 bipartite graph problems using that same embedding. Furthermore, since steps 1-4 above utilize the entire architecture, it is possible for multiple copies of smaller problems to be embedded on the architecture at the same time, effectively solving the problem multiple times every iteration of the quantum annealing algorithm.

If the system is annealed slowly enough from the initial quantum state to the final problem state, then the system is guaranteed to find the optimal solution. However, this time might be exponentially slow. An interesting result shown while doing experiments on the D-Wave stated that results from the quantum annealing follow a joint Boltzmann distribution, with all of the qubits as variables. This is exactly the information we need to gather to make an update of the parameter weights in equation (1). In fact, we can modify the update equation to the form that includes sampling from a quantum annealer:

$$w_{ij}^{t+1} = \alpha w_{ij}^{t} + \varepsilon\left(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{Q-sample}\right) \quad (6)$$

So, with the quantum annealer, we can estimate the final term in the weight update much more accurately in a reproducibly fast way. This insight can be used to more rapidly perform the unsupervised learning stage in DBN training.

4. Methods

The first goal of this project was to identify handwritten digits using a Deep Belief Network, as was done in the Adachi and Henderson study [7]. For this, I built off of the source code from a MATLAB toolbox implementation of Deep Neural Networks [12]. I modified this code to be able to perform *CD-n* updatets, as well as iterate over different learning rates and momentum parameters.

When exploring hyperparameter space, I first iterated over a logarithmic scale of values, then took the best result of that sweep (averaged over 3 trials) and did a more fine-tuned exploration around the best result. I continued exploring until I found a local optimum, i.e. the values on either side of that hyperparameter value resulted in lower performance.

I also attempted to modify this toolbox to make calls to the D-Wave to sample from the quantum annealing solutions in order to get an estimate of the model expectation, as in equation 6. However, due to D-Wave system outages and other complications, I was unable to get results from the quantum annealer for this project.

5. Data

5.1. Cancer Images

For the cancer image analysis portion of this project, I have obtained 144 hematoxylin and eosin stained lung tissue slides, which radiologists use to diagnose lung

cancer. This dataset was used by Levensen et al. in a cancer image classification study, which used pigeons to identify cancerous images [13]. This dataset contains 72 cancerous and 72 non-cancerous lung cancer images. The images are further broken down by magnification: 48 images magnified 4x, 50 images magnified 10x, and 46 images magnified 20x. All images were 500x500 pixels.

At each magnification, I divided up the images into ½ training, ¼ validation, and ¼ testing images. For the training images, I rotated them 0, 90, 180, and 270 degrees, as well as rotated the mirror images 0, 90, 180, and 270 degrees, for a total of 576 training images. The test images and validation images I did not augment. Examples of both cancerous and non-cancerous images are shown in Figure 4.

5.2. MNIST

For recreating the image analysis done using the quantum annealer, I am using a scaled-down version of the MNIST dataset, which contains 60,000 training images and 10,000 test images. For each image, I have created a coarse version of the 28x28 image by removing a 2-pixel wide strip around the edges, and then taking the average of each 4x4 square of pixels to create a single pixel, scaling each image down to 6x6. This is shown in Figure 5.

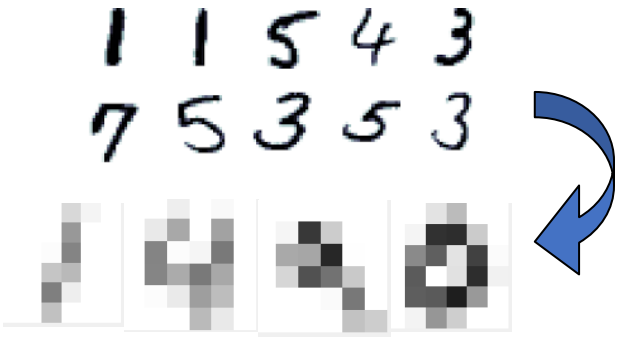

Figure 5: Example images from the MNIST dataset [11], original (top) and scaled down to 6x6 images, in order to fit on a D-Wave QPU (bottom).

6. Results

The experiments I ran are shown in Figure 6. Since I was unable to use the quantum annealer to utilize equation 6, I compared equations 1 (right column) and 2 (left column). For the model expectation in equation 1, I ran *CD-1000*, which, due to the many iterations of contrastive divergence, calculated a near-exact value for the expectation, and allowed a near-optimal directional step for the weight updates. Additionally, running *CD-2000* did not offer any noteworthy improvement in performance on a small number of test cases, which indicates that *CD-1000* does indeed perform a near-optimal update.

The optimal hyperparameters are shown in Table 1. In order to determine the hyperparameters to use, I considered the maximum accuracy on 100 pre-training iterations and 200 post-training iterations, averaged over 3 trials. The momentum for *CD-1* during pre-training is much larger than the momentum for *CD-1000*. This makes sense, as we expect *CD-1000* to have a much more accurate estimate of the gradient at each time step, so much less of the previous steps should be used for the current update.

	CD-1	***CD-1000***
α_{pre}	0.82	0.14
ε_{pre}	0.094	0.092
α_{post}	0.91	0.90
ε_{post}	0.000015	0.000011

Table 1: Optimal hyperparameters (momentum, learning rate) for both unsupervised (pre) training and supervised (post) training, for the commonly used stochastic update *CD-1*, and *CD-1000*, which serves as an estimate for the true weight update.

The graphs in Figure 6 show that, even with the optimal learning parameters (for 100 pre-training and 200 post-training iterations), *CD-1* cannot achieve any improvements beyond random guessing in as many as 50 pre-training and 100 post-training iterations. As an aside, when *CD-1* was run for 10,000 post-training iterations with the parameters shown in column 1 of Table 1, the model achieved a training accuracy of 0.86 and a test accuracy of 0.81. However, when using *CD-1000*, there was a significant increase in prediction accuracy as a function of pre-training iterations, even when only running 10 iterations of post-training. This is promising for the quantum annealer, which is expected to follow the *CD-1000* curves more closely, while taking a fraction of the time to train.

7. Future Work

In this paper, I have shown how quantum annealing can be used in the unsupervised learning of Deep Belief Networks. I have given simulated results that show the potential improvement that can be gained from using quantum annealing in this way. Work is ongoing to get results from the D-Wave 2X quantum annealer to concretely demonstrate an improvement.

One important area of future research is to explore opportunities to analyze larger images with the D-Wave quantum annealer. A possible study that would address this is the analysis of Convolutional Deep Belief Networks (CDBNs). There, the constraining factor is not the size of the pixel input, it is the size of the filters used.

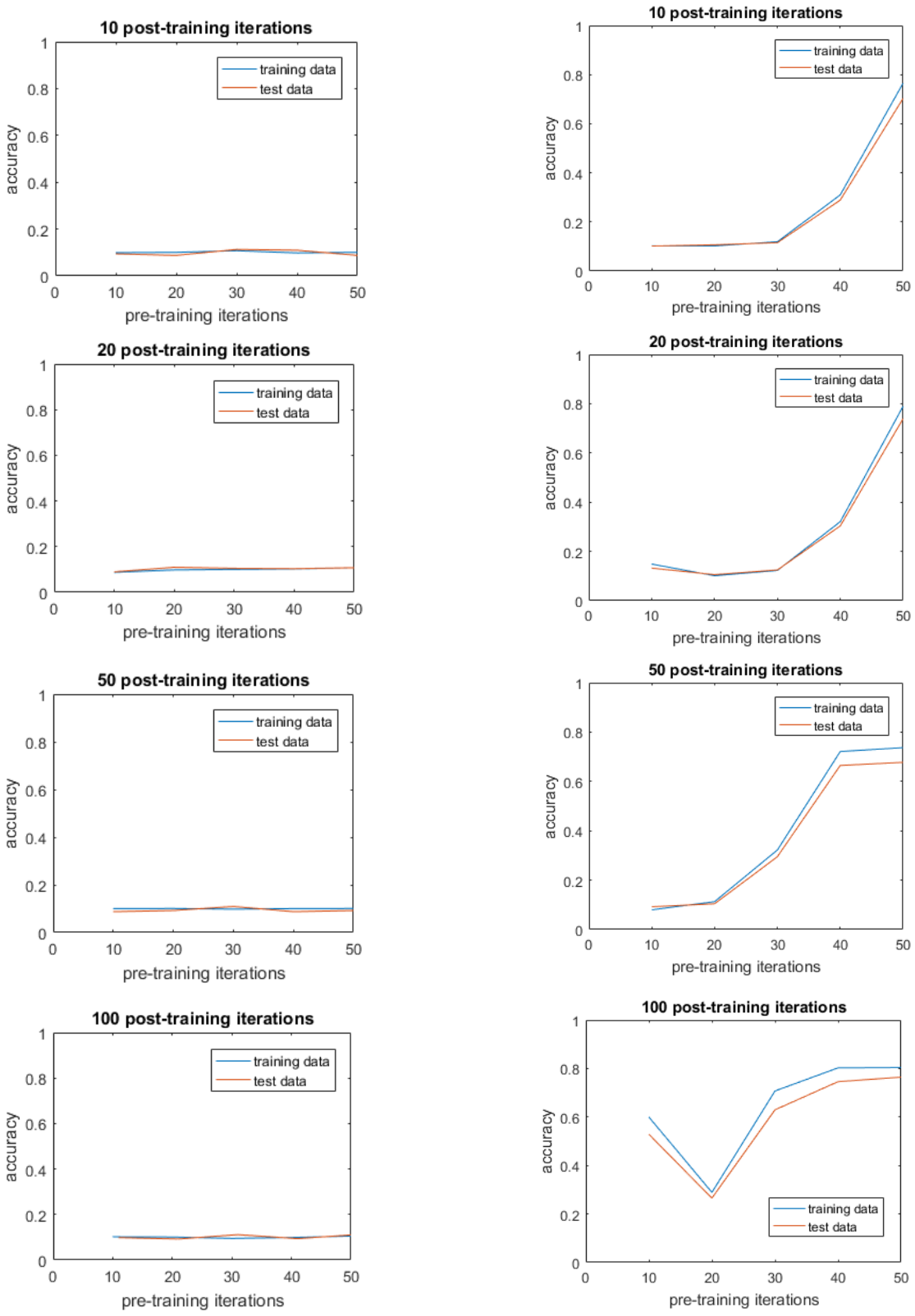

Figure 6: Results showing prediction accuracy of running *CD-1* (left) and *CD-1000* on the DBN to identify coarse-grained MNIST images. With parameters tuned for optimal accuracy for 200 post-training iterations and 100 pre-training iterations, *CD-1* never showed any improvement over random guessing. However, *CD-1000*, which is meant to serve as a near-optimal update rule, was able to achieve significant accuracies with as few as 10-post training iterations, and there was a noticeable improvement with increased pre-training iterations.

Acknowledgments

I gratefully acknowledge support from the Canary Foundation for Early Cancer Detection for this research. I am also thankful to D-Wave Systems for access to a D-Wave 2X quantum annealing processor, as well as many helpful discussions with D-Wave researchers and engineers.

References

[1] D. Forman, F. Bray, D. Brewster, C. Gombe Mbalawa, B. Kohler, M. Piñeros, E. Steliarova-Foucher, R. Swaminathan, and J. Ferlay, editors, Cancer Incidence in Five Continents, Vol. X. Lyon: International Agency for Research on Cancer, 2013.

[2] American Cancer Society. Cancer Facts & Figures 2017. Atlanta, Ga: American Cancer Society; 2017.

[3] W. Li et al. "Pulmonary Nodule Classification with Deep Convolutional Neural Networks on Computed Tomography Images." Computational and Mathematical Methods in Medicine 2016 (2016): 6215085. PMC.

[4] K. Yu, C. Zhang, G. Berry, R. Altman, C. Re, D. Rubin, and M. Snyder, "Predicting non-small cell lung cancer prognosis by fully automated microscopic pathology image features," Nature Communications, Vol. 7, pp. 12474, 2016.

[5] V. Denchev, S. Boixo, S. Isakov, N. Ding, R. Babbush, V. Smelyanskiy, J.Martinis, and H. Neven, "What is the Computational Value of Finite Range Tunneling?," Phys. Rev. X, Vol. 6, No. 3, pp. 031015, 2016.

[6] D. Korenkevych, Y. Xue, Z. Bian, F. Chudak, W. G. Macready, J. Rolfe, and E. Andriyash, "Benchmarking Quantum Hardware for Training of Fully Visible Boltzmann Machines," arXiv preprint, arXiv:1611.04528, 2016.

[7] S. H. Adachi and M. P. Henderson, "Application of quantum annealing to training of deep neural networks," arXiv preprint, arXiv:1510.06356, 2015.

[8] D. O'Malley, V. Vesselinov, B. S. Alexandrov, and L. B. Alexandrov. "Nonnegative/binary matrix factorization with a D-Wave quantum annealer," arXiv preprint arXiv:1704.01605, 2017.

[9] S. W. Shin, G. Smith, J. A. Smolin, and U. Vazirani, "How 'Quantum' is the D-Wave Machine?," arXiv preprint arXiv:1401.7087, 2014.

[10] J. King, S. Yarkoni, J. Raymond, I. Ozfidan, A. D. King, M. M. Nevisi, J. P. Hilton, and C. C. McGeoch, "Quantum annealing amid local ruggedness and global frustration," arXiv preprint arXiv:1701.04579, 2017.

[11] Y. LeCun et al. "The MNIST database of handwritten digits." 1998. http://yann.lecun.com/exdb/mnist/

[12] M. Tanaka (2013). Deep Neural Network Toolbox, MATLAB File Exchange. http://www.mathworks.com/matlabcentral/fileexchange/42853-deep-neural-network

[13] Levenson RM, Krupinski EA, Navarro VM, Wasserman EA. "Pigeons (Columba livia) as Trainable Observers of Pathology and Radiology Breast Cancer Images." *PLOS ONE* 10(11): e0141357. 2015.