Quantum Unsupervised Domain Adaptation: Does Entanglement Help?

Nanqing Dong[1]
nanqing.dong@cs.ox.ac.uk
Michael Kampffmeyer[2]
michael.c.kampffmeyer@uit.no
Irina Voiculescu[1]
irina.voiculescu@cs.ox.ac.uk

[1] Department of Computer Science
University of Oxford
Oxford, UK

[2] Department of Physics and Technology
UiT The Arctic University of Norway
Tromsø, Norway

Abstract

Fueled by the recent breakthroughs in quantum theory and hardware, it has been demonstrated that computer vision (CV) tasks can be solved in quantum computers via the emerging quantum machine learning (QML). In contrast to classical computing, quantum computing relies on the entanglement between qubits to communicate. While many conventional machine learning (ML) tasks have been well-studied, there are still ML problems to be solved on quantum data. One of this challenges is the domain shift on quantum data. In this work, we aim to understand the role of entanglement in mitigating the domain shift in a quantum domain. We formulate quantum unsupervised domain adaptation (QUDA) for the first time and, to address the domain shift existing in quantum data, propose quantum adversarial domain adaptation (QADA). Limited by the capacity of the current quantum devices, we lay the groundwork for a computation-efficient quantum system that implements QADA in a simple manner. QADA integrates two quantum neural networks (QNNs), including a quantum classifier and a quantum discriminator. Two QNNs are trained in a hybrid classical-quantum platform with an adversarial strategy. We evaluate QADA on unsupervised domain adaptation tasks between the MNIST and SVHN image datasets under a quantum setting. Our simulated experiments show that QADA can be used to mitigate the domain shift in a quantum device, which further validates that the entanglement in a quantum circuit model can be used to achieve QUDA.

1 Introduction

Albert Einstein described *entanglement* [38] as a *spooky* phenomenon in quantum physics. The entanglement between two quantum bits, or *qubits*, cannot be implemented by any classical computers, which makes the entanglement a characteristic feature of quantum computing. In the era of near-term noisy intermediate-scale quantum (NISQ) computing [36], quantum machine learning (QML) [6, 39] has gained increasing attention. As an emerging research area in its early stage, the goal of QML is to understand or develop machine learning (ML) models on quantum hardware [13]. So far, there have been exploratory studies

using quantum neural networks (QNNs) to solve simple ML tasks, such as image classification [8, 10, 15, 22, 32]. QNNs are quantum circuit models consisting of parametric quantum gates. A QNN can be viewed as a variational quantum circuit (VQC) [37] in the quantum physics literature. Although the quantum gates in VQC are *unitary i.e.* linear transformations, it has been proven that VQC can approximate nonlinear functions [7, 33]. The input to QNNs are qubits, where semantic information is encoded as quantum states. Similar to classical NNs, a forward pass in a QNN is equivalent to manipulate the qubits via sequential quantum gates. Meanwhile, the loss can be back-propagated [24] by approximating the gradients for the parametric quantum gates. In the near term, the optimization of VQC is implemented via a hybrid quantum-classical (HQC) framework [37], where the computing tasks can be decomposed into two sub-tasks. The first sub-task is to apply quantum gates to manipulate qubits in a quantum computer and the second sub-task optimizes the parameters of quantum gates in a classical computer.

Previous quantum studies assume that the training and test set come from the same underlying distribution, which does not consider the scenarios that two sets are collected from different data distributions, namely the *source domain* and the *target domain*. In practice, the source and the target domain may have significant distributional divergence, which is referred to as the *domain shift*. The domain shift is a practical challenge in CV as it causes misalignment in feature space when the model learned from the source domain is directly applied to the target domain, thus leading to a performance drop in the target domain. In classical ML, this problem has been investigated under the term unsupervised domain adaptation (UDA) [9], for scenarios where the labels of the target domain are inaccessible during the training. Although large progress has been made in UDA [16, 20, 27, 28, 29, 40, 41, 42, 44], how to address domain shift in the quantum domain remains an open question. We define this new problem as *quantum unsupervised domain adaptation* (QUDA). Note

In this work, we make the first attempt to address QUDA by proposing *quantum adversarial domain adaptation* (QADA). In classical adversarial domain adaptation (ADA) [16, 20, 29, 41, 42, 44], there are two complementary NNs. Take image classification as an example, ADA consists of a classifier and a discriminator. The two NNs play a minimax game [17], where the discriminator is trained to discriminate the difference between the source and the target domain and the classifier is trained to learn domain-invariant representations for the classification task. In the quantum domain, simply introducing an auxiliary QNN analogous to ADA, is not possible due to the following three challenges: 1) we can not physically access the intermediate quantum states within a QNN as only the measurement outcomes of the qubits are observable; 2) the common feature vectors of classical NNs are ill-defined in the *tensor product Hilbert space*; and 3) due to the *No-Cloning Theorem* [43], quantum states can not be copied. The first two challenges require that the system can easily manipulate the quantum representations. The third challenge encourages the efficient use of the quantum data, as the cost of quantum state preparation (*i.e.* preparing quantum data) could be easily ignored in simulated QML studies [2]. To bridge the aforementioned challenges and facilitate QUDA, we present our solution QADA. The overall workflow is illustrated in Fig. 1. Note that, instead of having two separate NNs as in ADA, two QNNs in QADA are integrated into one system. This novel design not only mitigates the above drawbacks but also gains system efficiency for practical implementations. Considering that QADA relies on entangling gates to connect multiple qubits, equivalently, the question of interest in this work can be rephrased as: *does entanglement help?*

To empirically validate this theoretical concept, we leverage a CPU-supported simulated HQC platform. We adapt a classical ADA task to the quantum setting to evaluate QADA,

where the source and target datasets are MNIST [24] and SVHN [35] respectively. We utilize a classical-to-quantum embedding network to encode qubits with classical semantic information. The experimental results not only show that QADA can be used to mitigate domain shift in the quantum domain, but also validate that the entanglement, as the fundamental element of QNNs, can play an important role in QUDA.

Note, the main purpose of this work is to solve UDA on quantum data, instead of using quantum computing to solve UDA. Our contributions can be summarised as follows. 1) To the best of our knowledge, we are the first to formulate the problem of QUDA. 2) We propose QADA, the first method for QUDA. 3) We design and develop the proposed quantum circuit model in a simulated HQC platform. 4) We simulate the quantum domain shift environment with classical datasets and empirically evaluate QADA.

2 Related Work

2.1 Adversarial Domain Adaptation

Inspired by generative adversarial networks (GANs) [17], ADA utilizes a discriminator to learn domain-invariant representations for the classifier. Among these seminal ADA methods [16, 20, 29, 41, 42, 44], DANN [16] and ADDA [41] are the most related to this work. DANN proposes a gradient reverse layer where the classifier is trained to classify the categories and confuse the discriminator at the same time. ADDA instead, consists of three stages. In the first stage, a source feature extractor and a feature classifier are trained. Then, a target feature extractor and the discriminator are trained jointly to make the discriminator unable to distinguish the representations extracted from the source and the target domain. Finally, the feature classifier is placed after the target feature extractor to produce the complete image classifier for the target domain. Compared to DANN, ADDA requires additional memory footprint, which might not be an economic solution for QUDA. Note, quantum data is different from classical data, which makes QUDA different from UDA fundamentally. Although the state-of-the-art UDA performance has been achieved by many classical methods, these methods can not be applied to quantum data. Thus, we do not compare with state-of-the-art UDA methods in Sec. 5.

2.2 Quantum Generative Adversarial Networks

Normally, a quantum system can generate data with statistics beyond the capacity of a classical system [13, 25]. This phenomenon is also known as *quantum supremacy* [1, 36] or *quantum advantage* [25]. It has been shown that quantum generative adversarial networks (QGANs) [8, 12, 37, 45] can exhibit quantum advantages for deep generative modeling. We want to highlight that QADA is not an extension of QGANs. QGANs usually have two quantum systems (or one quantum system and one classical system), which can not be adapted for QUDA. It it worth mentioning that, for QGANs with two separate quantum systems, the output of one QNN cannot be transferred or copied to the other QNN as the input. This is determined by the nature of quantum measurement. As a comparison, QADA only has one quantum system where two QNNs are linked and circumvent this problem.

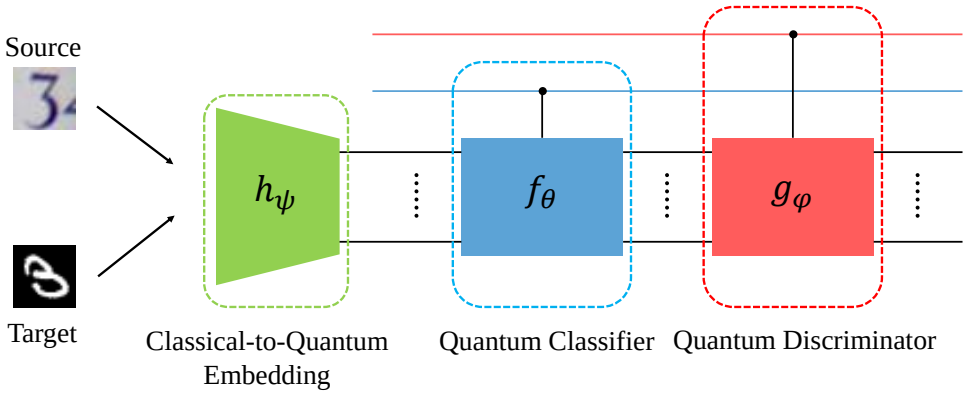

Figure 1: The quantum embedding network h_ψ (green) projects a classical image $\boldsymbol{x}$ into a quantum state vector $|\boldsymbol{x}\rangle$, which is described in Sec. 4.1. The quantum classifier f_θ (blue) takes $|\boldsymbol{x}\rangle$ as input and predicts the class of $|\boldsymbol{x}\rangle$ via the readout qubit (horizontal blue line), which is described in Sec. 4.2. The quantum discriminator g_ϕ (red) acts on the output quantum state vector of f_θ and classify whether $\boldsymbol{x}$ is from the source domain via the readout qubit (horizontal red line), which is described in Sec. 4.3.

3 Problem Formulation

Following previous discussions [3, 13, 14, 26, 32] on QNNs, we constrain the problem to be a binary image classification task and we constrain the input to be a quantum state vector. We consider a $(N+2)$-qubit quantum system as the composite of two systems, namely a N-qubit *input register* and a 2-qubit *output register*. Let $\mathcal{D}_S = \{(|\boldsymbol{x}_j\rangle, y_j)\}_{j=1}^{n_S}$ denote the labeled dataset sampled from the source domain $\mathcal{S}$ and $\mathcal{D}_T = \{|\boldsymbol{x}_j\rangle\}_{j=1}^{n_T}$ denote the unlabeled dataset sampled from the target domain $\mathcal{T}$. n denotes the number of examples for a given dataset. $|\boldsymbol{x}\rangle = |x_1\rangle \otimes |x_2\rangle \cdots |x_N\rangle$ is a N-qubit quantum state for the input register, where $|x\rangle = \alpha|0\rangle + \beta|1\rangle, \alpha, \beta \in \mathbb{C}, |\alpha|^2 + |\beta|^2 = 1$. $y_j \in \{-1, 1\}$ is a binary label for the corresponding $|\boldsymbol{x}\rangle$. In contrast to previous works, there are two qubits for the output register in this work. In addition to y, we assume that there is an auxiliary binary label $d \in \{-1, 1\}$ indicates whether the corresponding $|\boldsymbol{x}\rangle$ is sampled from $\mathcal{S}$ or not. Following classical UDA [4, 5, 16], the goal of the learning algorithm is to build a classifier $f : |\boldsymbol{x}\rangle \mapsto y$ with a low *target risk*

$$\mathcal{R}_{\mathcal{D}_T}(f) = \mathbb{P}_{(|\boldsymbol{x}\rangle, |y\rangle)\sim\mathcal{T}}[f(|\boldsymbol{x}\rangle) \neq y], \tag{1}$$

with only label information of $\mathcal{D}_S$. Note, with the definition of Eq. 1, the generalization bound [4, 44] for classical UDA still holds for QUDA.

4 Method

For consistency, we use for QADA the same notational conventions as in ADA. The quantum classifier is denoted as f_θ with parameters θ and the quantum discriminator is denoted as g_ϕ with parameters ϕ. In addition to the classifier and the discriminator, there is another quantum embedding network h_ψ with parameters ψ, which projects classical images

to quantum state vectors. Note, we simulate quantum data via $h_\psi$as we do not have access to real quantum data at the moment. The overall framework is shown in Fig. 1.

4.1 Quantum Embedding

Ideally, the real quantum state vector $|\boldsymbol{x}\rangle$ for a QNN should contain the same amount of information for the image of interest as its classical counterpart $\boldsymbol{x}$ for a classical NN. However, current quantum computers can not store the pixel-level information of a classical image $\boldsymbol{x}$ due to hardware limitations and we do have access to real quantum data in this study. Therefore, we adopt a common practice in QML, which is to assume that $|\boldsymbol{x}\rangle$ is a quantum embedding of the semantic information of the image of interest $\boldsymbol{x}$ [21, 31]. Let h_ψ be a quantum embedding network, which consists of two core parts. We use h_ψ to simulate quantum data based on classical data. The first part is a classical feature extractor that takes a classical image as input and outputs a feature vector. The second part is a linear classical-to-quantum projection network [31], which projects a high-dimensional feature vector to a N-qubit quantum state vector. We use h_ψ to encode classical semantic information into a quantum state vector. Given a well-trained h_ψ, we can have the simulated quantum data $|\boldsymbol{x}\rangle$ formulated in Sec. 3. Although, we can simulate quantum data in this way, the simulated quantum data may not fully reflect the characteristics of real quantum data.

4.2 Quantum Classifier

For the image classification task, the readout qubit for f_θ is prepared as $|1\rangle$. So the input quantum state vector to f_θ is $|1,\boldsymbol{x}\rangle = |1\rangle \otimes |\boldsymbol{x}\rangle$. Here, we require that f_θ has *full entanglement* [13] on $(N+1)$ qubits, *i.e.* each data qubit in $|\boldsymbol{x}\rangle$ and the readout qubit are entangled via entangling gates. With full entanglement, we only have to measure the readout qubit to infer the learning outcome, which is a computation-efficient design in quantum computing. Mathematically, each quantum gate can be represented as a unitary matrix. Thus, f_θ can be viewed as a large unitary matrix which maps $|1,\boldsymbol{x}\rangle$ from $\mathbb{C}^{2^{(N+1)}}$ to $\mathbb{C}^{2^{(N+1)}}$ in the Hilbert space. Under this definition, we can also represent f_θ as a parametric unitary matrix U_θ. The forward pass of the input state $|1,\boldsymbol{x}\rangle$ via f_θ is equivalent to applying an affine transformation U_θ on $|1,\boldsymbol{x}\rangle$. The output state is $U_\theta\,|1,\boldsymbol{x}\rangle$. As we are only interested in the readout qubit, the readout qubit is measured by a Pauli operator[1] in this work, which is a Hermitian matrix. As the measurement outcome on the readout qubit will be binary (either -1 or 1) with uncertainty. The prediction is defined as the expectation of the observed measurement outcomes if the output state $U_\theta\,|\boldsymbol{x},1\rangle$ is prepared and measured for multiple times. We have

$$f_\theta^0(\boldsymbol{x}) = \langle 1,\boldsymbol{x}|\,U_\theta^\dagger |Z\otimes \underbrace{I\otimes I\otimes\cdots\otimes I}_{N}|U_\theta\,|1,\boldsymbol{x}\rangle\,, \tag{2}$$

where $-1 \le f_\theta^0(\boldsymbol{x}) \le 1$[2]. We use the superscript 0 in $f_\theta^0(\boldsymbol{x})$ to denote the measurement on the readout qubit and we have $f_\theta^0(\boldsymbol{x}) : \mathbb{C}^{2^N} \mapsto [-1,1]$. Given the logit $f_\theta^0(\boldsymbol{x})$ and the binary label y, we can now define the hinge loss as

$$\mathcal{L}(f_\theta^0(\boldsymbol{x}),y) = \max(0, 1 - y\cdot f_\theta^0(\boldsymbol{x})). \tag{3}$$

[1] The common Pauli operators: $Z = \begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}, X = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}, I = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$.

[2] $\dagger$ denotes the adjoint operation, *i.e.* the complex conjugate of the transpose.

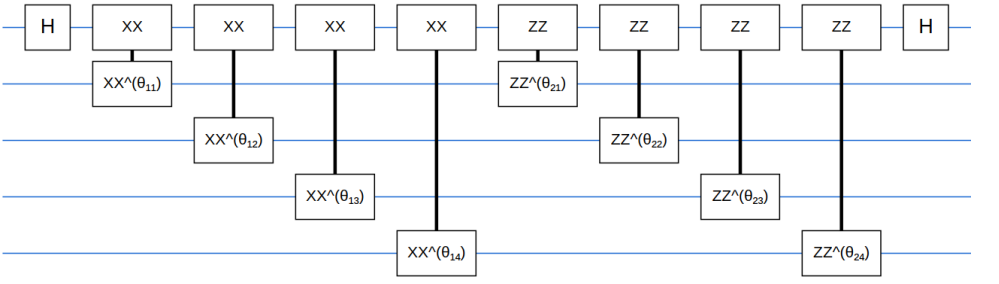

Figure 2: Demonstration of the network architecture of a 2-layer QNN with 4 data qubits and 1 readout qubit. The top line represents the readout qubit. The following four lines represent 4 data qubits respectively. The vertical bold line represents the entanglement between two qubits. The white boxes are quantum gates. θ_{jk} stands for the parameter of the quantum gate operated between the readout qubit and the k^{th} data qubit at the j^{th} layer.

Similar to the logit defined in Eq. 2, we also want to define the representations learned by f_θ. Unlike classical NNs, it is impractical to extract features from any hidden layers of QNNs due to the physical implementation. To access the representations of QNNs, we project the features in the Hilbert space onto the Euclidean space via measurement on the data qubits transformed by f_θ ($U_\theta |1,\boldsymbol{x}\rangle$) and have

$$f_\theta^{1\cdots N}(\boldsymbol{x}) = \langle 1,\boldsymbol{x}| U_\theta^\dagger |I \otimes \underbrace{Z \otimes Z \otimes \cdots \otimes Z}_{N} | U_\theta |1,\boldsymbol{x}\rangle , \tag{4}$$

where the subscript $1\cdots N$ denotes the data qubits.

4.3 Quantum Discriminator

Similar to the image classification task, we can formulate the domain classification task in the same way. Without loss of generality, g_ϕ can be represented as U_ϕ in the matrix form. As the goal is to learn domain-invariant representations, the input state to g_ϕ should be the representations from the image classification task. As mentioned above, the quantum process applies on the whole system. So, for simplicity, we use $U_\theta^{\{1,\cdots,N\}} |\boldsymbol{x}\rangle$ to denote the transformation on the N data qubits alone. The readout qubit for g_ϕ is prepared as $|1\rangle$. So the input state of g_ϕ is $|1\rangle \otimes U_\theta^{\{1,\cdots,N\}} |\boldsymbol{x}\rangle$ and the output state is $U_\phi |1\rangle \otimes U_\theta^{\{1,\cdots,N\}} |\boldsymbol{x}\rangle$. Again, we only measure on the readout qubit

$$g_\phi^0(f_\theta,\boldsymbol{x}) = \langle 1| \otimes \langle \boldsymbol{x}| U_\theta^{\{1,\cdots,N\}\dagger} |U_\phi^\dagger| Z \otimes \underbrace{I \otimes I \otimes \cdots \otimes I}_{N} | U_\phi |1\rangle \otimes U_\theta^{\{1,\cdots,N\}} |\boldsymbol{x}\rangle , \tag{5}$$

where $-1 \leq g_\phi^0(f_\theta,\boldsymbol{x}) \leq 1$. Similar to Eq. 3, we can define the hinge loss for the domain classification as

$$\mathcal{L}(g_\phi^0(f_\theta,\boldsymbol{x}),d) = \max(0, 1 - d \cdot g_\phi^0(f_\theta,\boldsymbol{x})). \tag{6}$$

4.4 Network Architecture

Two QNNs are set to have the same network architecture in this work as a regularization [34]. We limit our choices of the building elements of QNNs in the range of single-qubit quantum gates and two-qubit entangling gates. A two-qubit entangling gate builds entanglement between two qubits but the two qubits are not necessarily neighboring. A single-qubit quantum gate acts on a single qubit. Without entangling gates, there is no interaction between qubits *i.e.* entanglement is the core of QNNs. Here, we adopt a seminal QNN architecture proposed by [14], which only utilizes Z-parity gates ($ZZ^{\theta} = e^{-i\theta Z\otimes Z}$), X-parity gates ($XX^{\theta} = e^{-i\theta X\otimes X}$), and Hadamard gates ($H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$). According to *ZX-calculus* [9], any nonlinear function can be ε-approximated by combining these three quantum gates. It has been shown that this architecture represents a generic family of QNNs and can be used to illustrate the effect of entanglement [13]. The network architecture is illustrated in Fig. 2. Analogous to classical NNs, each block of parity gates can be considered as a layer in QNNs. There are an X-parity layer and a Z-parity layer in Fig. 2.

4.5 Optimization

With all components defined above, we introduce the optimization procedure of QADA. Although the capacity of current quantum computers limits the number of qubits in real quantum applications, we choose a gradient-based optimization method for QNN which can scale up to a larger number of parameters in the long term. Following [13, 21], we use the *parameter shift rule* [11] to approximate the quantum gradient. Formally, let θ_j and ϕ_k be the j^{th} and k^{th} parameters of the parameter sets θ and ϕ. The gradient of θ_j in Eq. 3 is

$$\nabla_{\theta_j}\mathcal{L}(f^0_{\theta}(\boldsymbol{x}),y) = \frac{\mathcal{L}(f^0_{\{\theta|\theta_j\to\theta_j+\frac{\pi}{2}\}}(\boldsymbol{x}),y) - \mathcal{L}(f^0_{\{\theta|\theta_j\to\theta_j-\frac{\pi}{2}\}}(\boldsymbol{x}),y)}{2}, \tag{7}$$

where $\{\theta|\theta_j\to\theta_j+\frac{\pi}{2}\}$ denotes that the parameter θ_j is updated to $\theta_j+\frac{\pi}{2}$ and other parameters in θ are unchanged. Similarly, for the gradient of ϕ_k in Eq. 6, we have

$$\nabla_{\phi_k}\mathcal{L}(g^0_{\phi}(f_{\theta},\boldsymbol{x}),d) = \frac{\mathcal{L}(g^0_{\{\phi|\phi_k\to\phi_k+\frac{\pi}{2}\}}(f_{\theta},\boldsymbol{x}),d) - \mathcal{L}(g^0_{\{\phi|\phi_k\to\phi_k-\frac{\pi}{2}\}}(f_{\theta},\boldsymbol{x}),d)}{2}. \tag{8}$$

We adopt a similar alternating optimization strategy as in [30]. The training procedure for f_{θ} and g_{ϕ} is described in Algo. 1. With f_{θ} fixed, we first train g_{ϕ} to discriminate the domain of the input state $|\boldsymbol{x}\rangle$. Then, with g_{ϕ} fixed, we train f_{θ} to classify the input state $|\boldsymbol{x}\rangle$ while making the output state $U_{\theta}^{\{1,\cdots,N\}}|\boldsymbol{x}\rangle$ indiscriminate by g_{ϕ}, *i.e.* encouraging f_{θ} to learn domain-invariant representations. The quantum embedding h_{ψ} can be acquired either in a supervised fashion (*e.g.* h_{ψ} can be jointly optimized with f_{θ} in a supervised classification task to learn such an embedding) or an unsupervised fashion [26]. We assume h_{ψ} is given in Algo. 1, so $|\boldsymbol{x}\rangle$ is given as stated in Sec. 3.

Algorithm 1: Minibatch stochastic gradient descent training of QADA. The number of steps to apply to g_ϕ, k is a hyperparameter. The hyperparameter λ controls the weight of the adversarial loss.

- Initialize f_θ, g_ϕ.

for number of training iterations **do**

  for k steps **do**

    - Sample minibatch of $N/2$ images from $\mathcal{D}_S$ and $N/2$ images from $\mathcal{D}_T$.
    - Update ϕ by minimize $\mathcal{L}(g_\phi^0(f_\theta, \boldsymbol{x}), d)$.

  end

  - Sample minibatch of $N/2$ images from $\mathcal{D}_S$ and $N/2$ images from $\mathcal{D}_T$.
  - Update θ by minimize $\mathcal{L}(f_\theta^0(\boldsymbol{x}), y) - \lambda \mathcal{L}(g_\phi^0(f_\theta, \boldsymbol{x}), d)$.

end

5 Experiments

5.1 Experimental Setup

5.1.1 Environment

Limited by the hardware, the simulated experiments were conducted on a classical computer with an Ubuntu 18.04 LTS system. The CPU is an Intel® Xeon® Processor E5-2686 v4 @ 2.30 GHz with 45 MB cache with 64GB RAM.

5.1.2 Datasets

We used two public datasets for UDA in the classical domain. The MNIST dataset [24][3] contains grayscale images of ten handwritten digits (0 to 9). Each image has a fixed resolution of 28×28. The SVHN dataset [35][4] contains character-level digits cropped from the RGB house number images collected in the street. Each digit belongs to one of ten classes (0 to 9) and each image has a fixed resolution of 32×32. Following [3, 14, 15], we only choose digit 3 and digit 6 as the classes of interests. We used MNIST and SVHN as the source and target datasets and investigated UDA performance both ways, *e.g.* $M \rightarrow S$ denotes that the source dataset is the training set of MNIST and the target dataset is the test set of SVHN.

5.1.3 Implementation

We use a ResNet18 [18] pre-trained on ImageNet [23] as the fixed feature extractor. The linear classical-to-quantum projection network in Sec. 4.1 consists of two linear projections. The first linear projection is to project a 512-dimensional feature vector into a N-dimensional feature vector. This dimension reduction step can be achieved via an autoencoder [19] in an unsupervised fashion. The second linear projection is to transform a classical feature vector into a N-qubit quantum state vector. Each element of the classical feature vector is transformed into an angle ω (radian) for the corresponding single-qubit Y gate $R_y(\omega) = \begin{bmatrix} \cos(\frac{\omega}{2}) & -\sin(\frac{\omega}{2}) \\ \sin(\frac{\omega}{2}) & \cos(\frac{\omega}{2}) \end{bmatrix}$, Note, quantum embedding in this step is equivalent to rotate a qubit

[3] http://yann.lecun.com/exdb/mnist

[4] http://ufldl.stanford.edu/housenumbers

Method	$M \to M$	$M \to S$	$S \to S$	$S \to M$
SL	0.9106	-	0.6987	-
TL	-	0.4686	-	0.5005
QADA	-	0.5733	-	0.6768

Table 1: Binary accuracy for QADA where the quantum classifier is a 2-layer QNN.

$|0\rangle$ in an initial state along with Y axis by ω (radian). Given a real number r_j in the classical feature vector, we have the corresponding $\omega_j = \frac{e^{r_j} - e^{-r_j}}{e^{r_j} + e^{-r_j}} \cdot \frac{\pi}{2}$. Two QNNs are implemented as in Sec. 4.4. We use the 2-layer QNN (XX-ZZ) as the baseline model. The batch size is 4. For simplicity, we use a fixed learning rate of 10^{-4} for both QNNs and λ is set to be 0.001.

5.2 Empirical Results

5.2.1 Main Finding

As one of the contributions, we demonstrate that QUDA is necessary when the domain shift exists in the quantum data. Here, we denote the standard supervised learning as SL, where the training and test sets come from the same distribution. We denote transfer learning as TL, which directly applies the model trained on the source domain to the target domain. As shown in Table 1, there is a severe performance drop when the training and test sets come from two different distributions. As a comparison, QADA can efficiently alleviate the downside of domain shift. Based on the fact that the qubits are connected only via entangling gates, we conclude that the entanglement does play an important role in QUDA. A physical interpretation of a quantum circuit model is that quantum gates rotate qubits in a XYZ axis system. Let the readout qubit be a ball with a directional pole and the binary decision boundary is on the sphere. The optimization process is expected to rotate the readout qubit to ensure that the pole points in the right region of the sphere. Thus, the process of QADA can be understood as using the quantum discriminator to regularize the rotation of the readout qubit of the quantum classifier.

5.2.2 Ablation Studies

We examine the impact of the number of data qubits N and the depth (the number of layers L) of QNNs on QADA in Tables 2 and 3 respectively. For the first experiment, we use the same 2-layer QNN architecture above but with a different number of qubits in the system. As shown in Table 2, the increase in N does not necessarily improve the performance of QUDA. Intuitively, more qubits should embed more information of classical features. However, the number of data qubits N in QNNs is not equivalent to the number of features in classical NNs and QNNs do not rely on features for binary classification. Instead, QNNs utilize the entanglement (multi-qubit interaction) to propagate information. More qubits might increase the difficulty of the entanglement-based information exchange across qubits. Another reasonable hypothesis is that some information is redundant for quantum classification. This hypothesis can be linked with learning domain-invariant representations by maximizing the cross-domain mutual information [27, 40] in classical UDA. Similarly, for the second experiment, we fix $N = 4$ but include multiple XX-ZZ blocks (each block has 2 layers) in the QNNs. The results in Table 3 imply that increasing the depth of QNN may not always help

N	$M \to S$	$S \to M$
4	0.5733	0.6768
8	0.5797	0.6982
16	0.5682	0.6402

Table 2: Impact of N on QADA.

L	$M \to S$	$S \to M$
2	0.5733	0.6768
4	0.5655	0.6499
6	0.5516	0.6331

Table 3: Impact of the depth of QNNs on QADA.

in QADA. This finding contrasts with the observations in classical ML, where classical NNs rely on deep layers to extract semantics features. The impact of N and the depth of QNNs are still inconclusive. Unfortunately, the current mathematical tools can not support a formal analysis on QNNs as we often do in classical NNs, *i.e*. QNNs are still black-boxes at the current stage. Here, we just want to share the empirical findings and provide the intuitive insights. Both findings suggest that QUDA is different from UDA, and QADA could be an interesting research direction.

5.2.3 Limitations

Since only simulated experiments are possible at this stage, any results on a real quantum device could be slightly different from those presented here. When simulating the quantum process with large N and L, the memory limit of classical computers can be easily exceeded; the training time is also long. Thus, the behavior of QADA for advanced experimental setups, such as (1) more complex QML tasks, (2) large N and L, or (3) large-scale datasets, is inconclusive and requires more than just further investigation: it awaits further development. This work serves merely as exploratory work as we can only adopt a simple setting.

6 Conclusions

In this work, we formulate and discuss the problem of QUDA, which addresses the domain shift problem on quantum data for the first time. We propose QADA, a computation-efficient solution leveraging QNNs and adversarial training. The empirical results not only demonstrate that QADA could be applied to practical QML tasks but also validate that the entanglement can be used for QUDA. In the future, QADA could be facilitated beyond the current hardware limitations by a joint effort in quantum theory and engineering.

Acknowledgement

We would like to thank Bob Coecke and Aleks Kissinger from the Department of Computer Science, University of Oxford, for knowledge sharing.

References

[1] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando GSL Brandao, David A Buell, et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574 (7779):505–510, 2019.

[2] Jeongho Bang, Arijit Dutta, Seung-Woo Lee, and Jaewan Kim. Optimal usage of quantum random access memory in quantum machine learning. *Physical Review A*, 99(1): 012326, 2019.

[3] Johannes Bausch. Recurrent quantum neural networks. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1368–1379. Curran Associates, Inc., 2020.

[4] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems*, pages 137–144, 2007.

[5] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175, 2010.

[6] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. Quantum machine learning. *Nature*, 549(7671):195–202, 2017.

[7] Yudong Cao, Gian Giacomo Guerreschi, and Alán Aspuru-Guzik. Quantum neuron: an elementary building block for machine learning on quantum computers. *arXiv preprint arXiv:1711.11240*, 2017.

[8] Shouvanik Chakrabarti, Huang Yiming, Tongyang Li, Soheil Feizi, and Xiaodi Wu. Quantum wasserstein generative adversarial networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[9] Bob Coecke and Aleks Kissinger. *Picturing Quantum Processes*. Cambridge University Press, 2017.

[10] Iris Cong, Soonwon Choi, and Mikhail D Lukin. Quantum convolutional neural networks. *Nature Physics*, 15(12):1273–1278, 2019.

[11] Gavin E. Crooks. Gradients of parameterized quantum gates using the parameter-shift rule and gate decomposition, 2019.

[12] Pierre-Luc Dallaire-Demers and Nathan Killoran. Quantum generative adversarial networks. *Physical Review A*, 98(1):012324, 2018.

[13] Nanqing Dong, Michael Kampffmeyer, Irina Voiculescu, and Eric Xing. Negational symmetry of quantum neural networks for binary pattern classification. *arXiv preprint arXiv:2105.09580*, 2021.

[14] Edward Farhi and Hartmut Neven. Classification with quantum neural networks on near term processors. *arXiv preprint arXiv:1802.06002*, 2018.

[15] Edward Farhi and Hartmut Neven. Classification with quantum neural networks on near term processors. *Quantum Review Letters*, 1(2 (2020)):10–37686, 2020.

[16] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

[17] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[19] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[20] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International Conference on Machine Learning*, pages 1989–1998, 2018.

[21] Ben Jaderberg, Lewis W Anderson, Weidi Xie, Samuel Albanie, Martin Kiffner, and Dieter Jaksch. Quantum self-supervised learning. *arXiv preprint arXiv:2103.14653*, 2021.

[22] Iordanis Kerenidis, Jonas Landman, and Anupam Prakash. Quantum algorithms for deep convolutional neural networks. In *International Conference on Learning Representations*, 2020.

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[24] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[25] Seth Lloyd and Christian Weedbrook. Quantum generative adversarial learning. *Physical Review Letters*, 121(4):040502, 2018.

[26] Seth Lloyd, Maria Schuld, Aroosa Ijaz, Josh Izaac, and Nathan Killoran. Quantum embeddings for machine learning. *arXiv preprint arXiv:2001.03622*, 2020.

[27] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning*, pages 97–105, 2015.

[28] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *International Conference on Machine Learning*, pages 2208–2217, 2017.

[29] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, 2018.

[30] Pauline Luc, Camille Couprie, Soumith Chintala, and Jakob Verbeek. Semantic segmentation using adversarial networks. In *NIPS Workshop on Adversarial Training*, 2016.

[31] Andrea Mari, Thomas R Bromley, Josh Izaac, Maria Schuld, and Nathan Killoran. Transfer learning in hybrid classical-quantum neural networks. *Quantum*, 4:340, 2020.

[32] Jarrod R McClean, Sergio Boixo, Vadim N Smelyanskiy, Ryan Babbush, and Hartmut Neven. Barren plateaus in quantum neural network training landscapes. *Nature communications*, 9(1):1–6, 2018.

[33] Kosuke Mitarai, Makoto Negoro, Masahiro Kitagawa, and Keisuke Fujii. Quantum circuit learning. *Physical Review A*, 98(3):032309, 2018.

[34] Saeid Motiian, Quinn Jones, Seyed Mehdi Iranmanesh, and Gianfranco Doretto. Few-shot adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, 2017.

[35] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[36] John Preskill. Quantum computing in the nisq era and beyond. *Quantum*, 2:79, 2018.

[37] Jonathan Romero and Alan Aspuru-Guzik. Variational quantum generators: Generative adversarial quantum machine learning for continuous distributions. *arXiv preprint arXiv:1901.00848*, 2019.

[38] Erwin Schrödinger. Discussion of probability relations between separated systems. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 31, pages 555–563. Cambridge University Press, 1935.

[39] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. An introduction to quantum machine learning. *Contemporary Physics*, 56(2):172–185, 2015.

[40] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.

[41] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017.

[42] Zeya Wang, Baoyu Jing, Yang Ni, Nanqing Dong, Pengtao Xie, and Eric P Xing. Adversarial domain adaptation being aware of class relationships. In *European Conference on Artificial Intelligence*, 2020.

[43] William K Wootters and Wojciech H Zurek. A single quantum cannot be cloned. *Nature*, 299(5886):802–803, 1982.

[44] Yuchen Zhang, Tianle Liu, Mingsheng Long, and Michael Jordan. Bridging theory and algorithm for domain adaptation. In *International Conference on Machine Learning*, pages 7404–7413, 2019.

[45] Christa Zoufal, Aurélien Lucchi, and Stefan Woerner. Quantum generative adversarial networks for learning and loading random distributions. *npj Quantum Information*, 5 (1):1–9, 2019.