Quantum deep transfer learning

Longhan Wang, Yifan Sun and Xiangdong Zhang

Key Laboratory of Advanced Optoelectronic Quantum Architecture and Measurements of Ministry of Education, Beijing Key Laboratory of Nanophotonics & Ultrafine Optoelectronic Systems, School of Physics, Beijing Institute of Technology, 100081, Beijing, People’s Republic of China

* Authors to whom any correspondence should be addressed.

E-mail: zhangxd@bit.edu.cn and yfsun@bit.edu.cn

Keywords: quantum transfer learning, quantum machine learning, quantum computation

Abstract
Quantum machine learning (QML) has aroused great interest because it has the potential to speed up the established classical machine learning processes. However, the present QML models can merely be trained on the dataset of single domain of interest. This severely limits the application of the QML to the scenario where only small datasets are available. In this work, we have proposed a QML model that allows the transfer of the knowledge from one domain encoded by quantum states to another, which is called quantum transfer learning. Using such a model, we demonstrate that the classification accuracy can be greatly improved for the training process on small datasets, comparing with the results obtained by former QML algorithm. Last but not least, we have proved that the complexity of our algorithm is basically logarithmic, which can be considered an exponential speedup over the related classical algorithms.

1. Introduction
The idea of machine learning (ML) was originated from the model of brain cell interactions several decades ago, which has led to various outstanding achievements during the recent years. The algorithms of it are employed in lots of field, such as speech recognition, visual object recognition, object detection, etc [1, 2]. Usually, the ML algorithms accomplish the tasks by training models on their own datasets [3–13]. The typical ones are, for example, logistic regression [8], neural network [5, 6, 9, 10], support vector machine [11, 12], Bayesian classifier [13] and so on.

Other than above ML algorithms, whose model are trained on the datasets drawn from the same feature space and the same distribution, there is another type of ML algorithm called transfer learning (TL). The main strategy of TL is to utilize the knowledge of other task domains to reduce the effort to train the model of interest. In many cases, such a strategy is found to be truly beneficial [14–19]. Generally, the TL can be performed by four steps, shown by figure 1(a). (1) For a given task with the dataset, find a source domain dataset for knowledge transfer. (2) Train a model on source domain dataset. (3) Build a criteria for the transfer process. Such a criteria is set to be the filter for obtaining the helpful information contained in the output of the source domain model, or in the model parameters generated by previous training process, etc, depending on the specific task. In other words, introduce a proper learning model. (4) Train the target task model on the target domain dataset using the learning information obtained by (3).

Finding the knowledge transfer structure is the key to TL, which is enabled by steps (2) and (3). For different learning tasks, these steps are executed using different learning models. For example, in a feature-based TL task [20], the feature dataset is obtained from the source dataset and characterized by a kernel matrix. Then, the alignments of feature and target data, given by the kernel matrices of them respectively, are employed as an evaluation of their similarity. Thus, by comparing the alignments, one can pick out the relevant feature datasets as a supply for the training of the task model. Obviously, such a strategy is especially supportive when the size of target domain data is insufficient for training an ML...
model. In fact, broad applications of TL can be found in plenty of daily issues. It has already shown the potential to solve practical challenges in the image recognition, social network analysis, and document classification, etc [14, 15].

The exploration on quantum computing indicates that a speed up can be achieved using quantum algorithms when solving certain problems. Proven by series of recent works [21–27], tasks of ML are also on the list and the corresponding solutions are termed quantum machine learning (QML) algorithms. By far, several QML algorithms have been proposed and demonstrated, including quantum support vector machine [28, 29], quantum deep learning [30–36], quantum Boltzmann machine (QBM) [37, 38], quantum generative adversarial learning [39–41] and so on. Also, in references [42, 43], the authors have discussed three kinds of TL schemes: quantum to classical, quantum to quantum and classical to quantum. They employ quantum variational circuit for the learning of the feature for the classification task. The quantum variational circuit requires a classical computer to obtain the updates of the parameters of the quantum circuit. In essence, they are hybrid classical–quantum TL schemes although they involve quantum to quantum TL. Recently, TL for scalability of neural-network quantum states [44] uses the classical learning model to express the quantum state, and reuses the previously trained parameters obtained by TL, thereby improving the effectiveness of the ground state ansatz. However, the ML strategies beyond the dataset-based training, such as the knowledge transfer enabled by classical TL algorithms, has not been found in the quantum framework for computing (for fully quantum–quantum computing). Those strategies would naturally combine the advantage of TL and quantum computation, removing crucial requirements on dataset, parameter updating, etc for QML and providing a speed-up in all stages of the processing.

In this work, we propose a complete scheme of quantum deep transfer learning (QDTL) based on energy model. As an analogy of classical TL process, our scheme takes advantage of the knowledge from other domain and is completely described by quantum states. We not only propose a generative model for the quantum data to be transferred, but also define the generalized kernel matrices of it, which is applicable for mixed quantum states. Also, the quantum alignment method is introduced to judge the similarity of two kernel matrices. As an example of practical illustration, we apply the scheme to the classification task of Iris dataset, and demonstrate that the classification accuracy can be improved greatly in comparison with the traditional QML algorithm.
2. Scheme of quantum deep transfer learning

The schematic diagram of the scheme is shown in figure 1(b). It also includes four steps which have a one-to-one correspondence to the classical scheme shown in figure 1(a). All datasets in the whole task are encoded by quantum states. Because the feature dataset here is encoded by mixed states, we use density matrix to describe the corresponding theory. The extract of knowledge transfer structure is also the heart of the scheme, which includes three parts: (1) generate quantum feature datasets using a generative model so that a series of quantum datasets sharing similar features of source domain can be obtained; (2) design a quantum subroutine for the propose using the network model of QBM. A QBM is a quantum machine starting with an initial state, a QBM can approximately reach the demanded Boltzmann distribution by annealing process [37, 45]. However, not all optimization problems can be solved using QBM model with equal number of visible and hidden nodes, depicted by blue and red circles respectively. Each node represents a two-level subsystem, or a qubit, whose basis states are denoted by $|\pm\rangle$. Figure 2 shows an example of a QBM, where the number of visible (blue) nodes are equal to the hidden (red) nodes. The edges between the nodes represents the weighted connections.

The edges between the nodes represents the weighted connections.

![Network model based on QBM. In the case, the number of visible (blue) nodes are equal to the hidden (red) nodes. The edges between the nodes represents the weighted connections.](image)

2.1. Quantum generative model for feature data

The purpose of obtaining a feature dataset in QDTL scheme can be reached in various ways. Here we design the subroutine for the propose using the network model of QBM. A QBM is a quantum machine described by Ising-type Hamiltonians whose eigenstates probability distribution is a Boltzmann distribution. Starting with an initial state, a QBM can approximately reach the demanded Boltzmann distribution by annealing process [37, 45]. However, not all optimization problems can be solved using QBM as an example and other models in QDTL protocols are similar. Specifically, for a given source dataset $\{\rho^i_s\}_{i=1..M_s}$, we train a QBM such that the marginal Boltzmann probability of model $\rho^i_s$ is close to probability of source data. Then, the quantum feature dataset can be obtained by measuring QBM according to the labeled target dataset $\{\rho^j_t, \|y\rangle\}_{j=1..M_t}$. Figure 2 shows an example of a QBM model with equal number of visible and hidden nodes, depicted by blue and red circles respectively. Each node represents a two-level subsystem, or a qubit, whose basis states are denoted by $|1\rangle$ and $|-1\rangle$. The QBM can be described by the transverse Ising model [28] so that the Hamiltonian is given by

$$
H = -\sum_a \Gamma_a \sigma^x_a - \sum_a b_a \sigma^z_a - \sum_{a,b} w_{ab} \sigma^x_a \sigma^z_b
$$

(1)

where $\Gamma_a$ is the parameter of transverse field, $b_a$ is the threshold of the network, and $w_{ab}$ is the weight between network nodes. $\sigma^x_a$ and $\sigma^z_a$ are the Pauli operators of $x$ and $z$ given by $\sigma^x_a = I \otimes \cdots \otimes I \otimes \sigma^x \otimes I \otimes \cdots \otimes I$ and $\sigma^z_a = I \otimes \cdots \otimes I \otimes \sigma^z \otimes I \otimes \cdots \otimes I$, with Pauli matrixes $\sigma^x$ and $\sigma^z$. $N$ is the total qubits number. With the Hamiltonian given by equation (1), the partition function and density matrix of QBM are $Z = \text{Tr} [e^{-H}]$ and $\rho = Z^{-1} e^{-H} [35]$.

According to the training process in [35], we obtain the workable QBM model, then measure the qubits of the visible layer of the QBM. The measurement operators are set to be the projection operators of the
sample states of the target domain. For example, the measurement operator defined by sample state $\rho_j^f$ can be given by

$$|v_j^f\rangle\langle v_j^f| = \prod_j^{1+\sqrt{\delta_j^f}} \frac{1}{2},$$

where $|v_j^f\rangle = |v_1^f\rangle|v_2^f\rangle\ldots|v_{N/2}^f\rangle$ denotes the state of the visible layer with $N/2$ qubits, $v_j^f$ is determined by the $l$th qubit of $\rho_j^f$ and equals to 1 $(-1)$ when the qubit is $|1\rangle$ ($|-1\rangle$). After the projective measurement of the visible qubits, we obtain the clamped Hamiltonian $H_{v_j^f} = \langle v_j^f|H|v_j^f\rangle$. Then, the partition function and density matrix of the qubits of the hidden layer are $Z_{v_j^f} = \text{Tr}[e^{-H_{v_j^f}}]$ and $\rho_{v_j^f} = Z_{v_j^f}^{-1}e^{-H_{v_j^f}}$, respectively. We call $\rho_{v_j^f}$ the feature state of sample $\rho_j^f$. It should be noted here that $\rho_{v_j^f}$ is actually a mixed state. Together with label state $|y_j\rangle$ of $\rho_j^f$, a feature sample pair $(\rho_{v_j^f}, |y_j\rangle)$ is obtained. After measuring the visible layer of the QBM with the projection operators defined by all the samples in $\{(\rho_j^f, |y_j\rangle)\}$, one can obtain a feature dataset $f_T = \{(\rho_j^f, |y_j\rangle)\}_{j=1..M_T}$, whose number of sample states is the same with target dataset.

### 2.2. Kernel matrix

For the aim of describing the characters of the datasets properly, we design a subroutine for representing the kernel matrix of density matrix of the feature dataset. The computation of kernel matrix for pure states has been proposed in reference [28]. Here, we provide a more generalized form for mixed states. In the scheme, a series of reference qubit systems $R = \{R_j\}_{j=1..M_T}$ are introduced. $R_j$ has the same state space with $\rho_j^f$ and the orthonormal basis states are denoted by $|n^{R_j}\rangle$, where $n^{R_j}$ is an integer ranging from 1 to $d$. Following the example of the last part, we set $d = 2^{N/2}$. By such reference systems, one can generate a series of pure states whose subsystem states are represented by $\{\rho_j^r\}$ [46]. A precise purification procedure has been given in reference [47]. We also apply a quantum random access memory (QRAM) [48] that can create a superposition of feature data in a data register, correlated with the address register. The state of address register of QRAM is denoted by $\rho = \frac{1}{M}\sum_{k=1}^{M_T} |k\rangle\langle r|$, where $k$ and $r$ represent the $k$th and the $r$th memory cells. In the QRAM, a superposition state can be obtained by the oracle operations on all the sample states $\rho_j^r$ of $f_T$, expressed by

$$\rho_1 = \frac{1}{M_T}\sum_{k=1}^{M_T} \sum_{n=1}^{d} \sqrt{p_k^r|\psi_n^m\rangle} \langle \psi_n^m| \otimes |n^{R_k}\rangle \langle m^{R_r}|,$$

where $|\psi_n^m\rangle$ $(|\psi_n^m\rangle)$ represents the $r$th (mth) basis state of $\rho_j^r$ ($\rho_j^m$) and $p_k^r$ ($p_m^r$) is the probability of it. After the operation of partial trace on reference system $R$, a mixed state $\widehat{K}$ can be obtained

$$\widehat{K}_1 = \operatorname{Tr}_R(\rho_1) = \frac{1}{M_T}\sum_{k=1}^{M_T} \sum_{n=1}^{d} \sqrt{p_k^r|\psi_n^m\rangle} \langle \psi_n^m| \langle r| \otimes |n^{R_k}\rangle \langle m^{R_r}|,$$

Then, after the partial trace on the feature states register $\{\rho_j^f\}$ for the mixed state $\widehat{K}_1$, a mixed state $\widehat{K}$ can be obtained

$$\widehat{K} = \operatorname{Tr}_f(\widehat{K}_1) = \frac{1}{M_T}\sum_{k=1}^{M_T} \sum_{n=1}^{d} \sqrt{p_k^r|\psi_n^m\rangle} \langle \psi_n^m| |k\rangle \langle r|. $$

In fact, the density matrix of $\widehat{K}$ given by equation (5) has a well correspondence with the classical kernel matrix [49].

In order to describe the characters of the labeled datasets, we also need to generalize a state $\widehat{K}_\text{ideal}$ to encode the normalized kernel matrix of the label state. Because, in our case, the label states are pure quantum states, we do not have to introduce the reference system. The state of kernel matrix about label data $|y_j\rangle$ can be directly obtained

$$\widehat{K}_\text{ideal} = \frac{1}{M_T}\sum_{j=1}^{M_T} \langle y_j | y_j \rangle |\bar{j}\rangle \langle \bar{u}|.$$ 

One can also address the similar results using the algorithm in reference [28].

### 2.3. Quantum alignment

To compare the similarity of the feature dataset and target dataset, we introduce the quantum alignment. It is defined by the trace of two kernel matrices of the quantum datasets. Our scheme is a more generalized
alignment is expressed as matrices. Following the spirit of the classical TL, the quantum alignment of a labeled dataset can be
the trick in the kernel matrix part, the purified quantum states of
(a list of each step for our scheme in table 1. Next, we show the advance by the example of the classification
than the original target dataset and is truly beneficial in practical use. In order to have a clear view, we show
three subroutines again.
visible layer from the last run (the MPD of the first run is from source dataset), then, executing the above
operations, the output quantum state
ancilla qubit being the control qubit. Finally, another Hadamard operation is performed on the ancilla
quantum alignment can be obtained.
In fact, equation (9) provides a method to estimate the trace of the product of two arbitrary density
matrices. Applying equation (9), each trace in equation (10) can be estimated so that the

\[ \text{Align}(\hat{K}) = \frac{\text{Tr}(\hat{K}\hat{K}^{\text{ideal}})}{\sqrt{\text{Tr}(\hat{K}) \sqrt{\text{Tr}(\hat{K}^{\text{ideal}})}}}. \]  

The detailed demonstration is presented in the section S1 of supporting materials (https://stacks.iop.org/NJP/23/103010/mmedia). Applying equation (9), each trace in equation (10) can be estimated so that the quantum alignment can be obtained.

Employing the above three subroutines, a complete QDTL can be realized by obtaining feature datasets multiple times. We compare the quantum alignment of each \( f_T \), denoted by \( K_f \), with that of
\[ \left\{ \left( \rho_{j1}^{f_T}, \langle y_j \rangle \right) \right\}_{j=1,M_T}, \]  

denoted by \( K_c \). Here \( f_T \) represents the \( t \)-th \( f_T \), which is given by training QBM based on the marginal probability distribution (MPD) of the visible layer from the last run. If \( K_f^{\text{ideal}} > K_c \) \cite{39} for some \( z_m \)s, dataset \( f_{T_m}^{\text{ideal}} \) is pick out as the supplement to the target dataset. Training QBM based on the MPD of the visible layer from the last run (the MPD of the first run is from source dataset), then, executing the above three subroutines again.

| Subroutine                             | Function                                      |
|----------------------------------------|-----------------------------------------------|
| Quantum generative model for feature data | Obtaining a feature dataset by QBM model       |
| Kernel matrix                          | Describing the characters of the feature dataset and the target dataset |
| Quantum alignment                      | Comparing the similarity of the feature dataset and target dataset |

Table 1. The subroutines of the QDTL scheme and their functions.
3. Examples

For displaying the effectiveness of the QDTL, the numerical simulation of its performance on the classification of Iris dataset is presented. The Iris dataset is a frequently-used dataset for the demonstration of classification task. It contains 150 samples of the Iris information that describe three types of the flowers, setosa, versicolour and virginica. Each sample contains four attributes: sepal length (SL), sepal width (SW), petal length (PL) and petal width (PW). Based on such a dataset, we discuss the binary classification task, that is, classification for two types of Iris. We firstly specify the datasets for our binary tasks. Suppose that the Iris dataset is denoted by $I = I_1 \cup I_2 \cup I_3$, where $I_1$, $I_2$ and $I_3$ are the datasets of setosa, versicolour and virginica samples respectively. The target domain dataset $T$ is set to be $I_p \cup I_q$, with $p, q = 1, 2, 3$ and $p \neq q$. The corresponding source domain dataset $S$ is set to be $I_p \cup I_s$, with $s = 1, 2, 3$ and $s \neq p, q$. This means that one sub-dataset of $T$ and $S$ contains the same type of Iris data samples. Here we only present the results when $T = I_2 \cup I_3$ in the main text, the source domain dataset is chosen to be $S_1 = I_1 \cup I_2$ and $S_2 = I_1 \cup I_3$.

More numerical examples are illustrated in S2 and S3 of supporting materials.

Before applying the above QDTL algorithm, we preprocess these datasets. The attribute of each sample is changed from a real number to the binary number. In other words, four thresholds are defined for the four attributes of samples. When these attributes are greater than the thresholds, they are marked as $1$. Otherwise, they are marked as $-1$. For such a preprocessing, the four thresholds of the attributes are demanded to be properly set so that the new samples do not belong to contradictory types. For example, we set the thresholds of SL, SW, PL and PW to be 5.5, 3.1, 2.0 and 1.6 respectively, for the above requirements.

The results of preprocessed datasets of versicolour and virginica are shown in figure 3(b). The ‘category’ means different binary data types after preprocessing. For example, ‘1’ represents the binary data $(−1, −1, −1, 1)$ whose elements indicates that all four character values (SL, SW, PL and PW) are below the corresponding thresholds. ‘2’ represents the binary data $(−1, −1, 1, 1)$ whose elements indicates only the last character value (PW) of sample is greater than its threshold, and the remains are less than the corresponding thresholds. The larger numbers follow logically. We can observe that the samples of versicolour are mainly distributed in binary data ‘11’ in blue, but there are also a small number of virginica samples in binary data ‘11’ in black. For comparison, the initial data distributions are shown in figures 3(a1) and (a2). Figure 3(a1) shows the distribution of SL and SW, and figure 3(a2) shows that of PL and PW. It is seen clearly that the initial data points of versicolour and virginica are not linear separation. After preprocessing, two types of data are still not linearly separable. This indicates that the main character of the datasets does not change and the preprocessing methods contributes nothing to the classification task. The results of preprocessed datasets for other cases are similar, which are given by S2 of supporting materials.

After preprocessing, we run our QDTL algorithm by employing the source dataset with 100 samples in each binary classification task. According to the above theoretical scheme, we consider a QBM discussed in section 2.1. This model has 4 visual units and 4 hidden units. We allow full connectivity within the visible layer, the hidden layer and all-to-all connectivity between the layers. We train the QBM for 160 gradient steps in the update rule and set the learning rate to 0.05. The QBM is trained again with the MPD of the visible layer from the last run. After 50 times of such training, we express the obtained the density matrix of
the feature dataset and the target dataset as a kernel matrix according to the theory of section 2.2, and then apply quantum alignment to compare the similarity of the feature dataset and target dataset. Finally, we select 4 suitable feature datasets to expand the target data so that the size of the target dataset can be enlarged to 500. In order to complete the binary classification task, we consider the quantum neural network given by [36] and train the model with a three-layer quantum neural network. We train the model for 1000 gradient steps in the update rule and set the learning rate to 0.03.

For comparison, we also train the network on the original target dataset. The results are shown in figure 4. Here the ratio of training sample to test sample is 4:1. The green lines in figures 4(a) and (b) display classification accuracies of the QDTL cases as a function of training times. The classification accuracies are arithmetic averaged over ten test results. The corresponding results based on the original target dataset with 100 samples, whose distributions are given by figure 3, are plotted in blue. In the two cases, the final accuracies increase 16.9% when employing $S_1$, and 24.6% when employing $S_2$. At the same time, we see that the standard errors of TL data are smaller than the target data, indicating that the former has more stable test results on the test data. The results of other examples are given by S3 of the supporting materials. In conclusion, all the results show a significant improvement of the classification accuracies when the QDTL scheme is applied.

4. Complexity analysis

In this section, we discuss the complexity of the quantum algorithm. We first estimate the complexity of generating feature dataset. Its complexity is composed of the QBM training and projective measurement process. In this module, we use the QBM model to train source dataset.

In order to make the error of sampling the gradient of the objective function less than or equal to $\varepsilon$, the query complexity is $O(M/\epsilon^2 - M\epsilon^2)$ [38]. We assume that we have an oracle, $F_H(\epsilon_H)$, that is capable of taking parameters $(\Gamma_a, b_a, w_{ab})$ of QBM and outputs the state $\sigma$ such that $\|e^{-H}/Z - \sigma\|_\epsilon \leq \epsilon_H$ for $\epsilon_H \geq 0$, where $\|\|_\epsilon$ represents the trace distance and $\epsilon_H$ is a constant. Further let D be an approximation to the gradient of training objective function of the QBM, which is obtained by sampling the expectation values of Hamiltonian. If $\epsilon \geq \sqrt{M\epsilon_H}$, there exist algorithms that enable a gradient to achieve $E([D - D_{true}]^2) \leq \epsilon^2$, where $D_{true}$ is the true gradient calculating the vector of expectation value of Hamiltonian and $M \in o(\sqrt{2N}^2)$. This can be done by querying $F_H(\epsilon_H)$ and the source domain dataset $O(M/\epsilon^2 - M\epsilon^2)$ times per epoch. In the reference [53], the gradient of QBM, which is compatible with our scheme, is estimated by comparing finite differences method with automatic differentiation approach. It indicates that the number of circuits mightly impacts the time and resources and we could choose different methods according to actual problems. The number of circuits in the former is $\Theta(tq(p + p))$ and the latter is $\Theta(tq^2(p + p))$, where $p$ is the number of Hamiltonian parameters, $q$ is the number of trial state parameters, $t$ is the number of time steps during the Gibbs state preparation. Another essential part of QBM is sampling from the thermal distribution. The complexity of preparing a Gibbs state $\rho \in \mathbb{C}^{2^N \times 2^N}$ within error $\epsilon$, is $O(\sqrt{2^N}/Z)\text{poly log}(1/\epsilon)\sqrt{2^N/Z}$ [54]. This is roughly quadratically better than existing approaches for preparing general Gibbs states [55] if constant $\epsilon$ is required, but constitutes an exponential improvement if $1/\epsilon$ is large.

However, there are still some open problems in the QBM quantum algorithm that need to be solved. For
example, how to estimate the number of epochs so that the algorithm converges. This number is unknown and depends sensitively on the training data as well as the learning rate. If QBM algorithm can simulate effectively, then this program cannot be simulated by classical methods unless BQP = BPP [56, 57]. According to the above complexity analysis of the gradient and sampling process, especially the analysis of the BQP-hard decision problem, QBM training offers the potential for exponential acceleration compared to classical ML methods.

Further, we obtain the feature dataset by projecting the target domain dataset to visual layer of the QBM. The probability of success of measurement is $\Theta(1)$, so that the number of measurements is about $|\{P_r \colon P_r > 0\}|$ in the target dataset. In the case of dataset of the same size, the module of generating feature dataset can provide exponential acceleration compared to classic ML methods.

In the kernel matrix part, we need to calculate the kernel matrix $\hat{K}$ and $\hat{K}_{\text{ideal}}$, respectively. According to the kernel matrix preparation method proposed in reference [28], one way to realize the construction of these quantum states is through QRAM, which uses $O(M_dT_d)$ hardware resources but only requires $O(\log M_dT_d)$ operations to access them. One of the caveats of QRAM is that it either contains the partial sums of probability amplitude of the quantum states, or pointers to the large entries of the states [58]. Because we encode samples with qubits, $|1\rangle$ and $|-1\rangle$, our scheme can address the caveat. In this way, the approach to obtain the superposition state of QRAM is through an oracle with the state $\rho = \frac{1}{M_T}\sum_{k=1}^{M_T} |k\rangle \langle r|$. Then, by performing partial trace operations on the reference system $R$ and feature states register, we can get the kernel matrix. Therefore, the complexity of the kernel matrix is equal to the complexity of QRAM, that is $O(\log M_dT_d)$. A total of six kernel matrices need be represented each time. According to Big O notation [46], any constants in product does not affect the order of complexity. So, it can be omitted. This means that the run time of kernel matrix is $O(\log M_dT_d)$ compared with the classical counterpart [59]. For the QRAM, however, it is difficult to obtain a known oracle under the existing experimental conditions, and this oracle is required to be an exponentially less query complexity. Reference [60] proposed another method of preparing the kernel matrix in a new Bayesian deep learning algorithm on a quantum computer. This method avoids the use of oracle, but provides a polynomial speedup effect compared to the classic algorithm.

In the quantum alignment part, each result of trace is estimated by a swap test. In reference [50], the numbers of single bit quantum gate used in each swap test are $O(\log M_T)$. In the swap test of quantum circuit, the number of quantum gates passing through the ancilla bit path is the largest, and it is equal to the number of quantum gates in the whole swap test process. Therefore, we can estimate that the time complexity of swap test is equal to the circuit depth, that is $O(\log M_T)$. In measurement for the ancilla bit, if we require the measurement accuracy of quantum alignment to be $\kappa$, then the complexity of each trace operation is $O(\kappa^{-1} \log M_T)$. Three trace operations are required for each calculation of $\text{Align}(K)$. According to Big O notation, the any constants in product can be omitted. The complexity of quantum alignment is $O(\log M_T)$, however, the complexity of alignment in reference [20] is $O(M_T^2)$.

The above content provides a detailed analysis about the three main parts of the QDTL algorithm. In the module of generating feature dataset, QBM provides a potential exponential acceleration result. In the kernel matrix modules, the complexity is $O(\log M_dT_d)$. In the quantum alignment modules, the complexity is $O(\log M_T)$. Further, we also provide the probability of successful postselection of the quantum alignment, which is approximately 0.79. The detailed results are given by S4 of the supporting materials. So we believe that the QDTL algorithm potentially offer an exponential advantage. However, the probability of successful postselection depends sensitively on the similarity of target dataset and source dataset. Further work will be needed to provide good empirical and theoretical bounds that are needed to obtain sufficient feature dataset.

5. Discussion and conclusion

Recent developments in quantum annealing processors have made QBM possible to experimentally test some of the quantum machine-learning ideas. With minor modifications to the hardware design, such a model can become available. However, as argued in reference [45], not all optimization problems can be solved using QBM. The quantum annealers have a significant dependence on annealing time and temperature. We hope this could open new possibilities in areas of ML and quantum information processing.

We would like to emphasize that for the simple dataset (e.g., the linearly separable one), larger networks are naturally low in efficiency because searching the optimal values of the object function in a large parameter space is high consuming and unnecessary. Great advantage is not shown using the QDTL scheme. In fact, the advantage of the QDTL scheme is for the classification tasks involving complex datasets, not the linearly separable ones. In these cases, the specific linear classifiers do not work well.
In conclusion, we have proposed a complete QDTL process for feature-based knowledge transfer. The key part of the knowledge transfer is composed of three parts: a model for generating quantum feature data based on QBM, an algorithm for computing kernel matrices of mixed quantum states, and a quantum alignment algorithm for comparing the characters of the kernel matrices. By applying our scheme to the classification task of Iris dataset, we have also demonstrated that the classification accuracy of the considered QML model can be greatly improved with the help of the QDTL scheme. Based on our scheme, the exponential speedup over classical counterpart has been demonstrated.

Acknowledgments

This work was supported by the National Key R & D Program of China under Grant No. 2017YFA0303800 and the National Natural Science Foundation of China (No. 91850205). We also acknowledge the support by National Science Foundation for Young Scientists of China (No. 11904022) and Beijing Institute of Technology Research Fund Program for Young Scholars.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

ORCID iDs

Yifan Sun https://orcid.org/0000-0003-4935-630X
Xiangdong Zhang https://orcid.org/0000-0002-7725-8814

References

[1] Bishop C. M 2006 Pattern Recognition and Machine Learning (Berlin: Springer)
[2] Jordan M I and Mitchell T M 2015 Science 349 255–60
[3] LeCun Y, Bengio Y and Hinton G 2015 Deep learning Nature 521 436
[4] Turaga S C, Murray J F, Jain V, Roth F, Helmstaedter M, Briggman K, Denk W and Seung H S 2010 Convolutional networks can learn to generate affinity graphs for image segmentation Neural Comput. 22 511
[5] Mnih V et al 2015 Human-level control through deep reinforcement learning Nature 518 529
[6] Hinton G E, Osindero S and Teh Y W 2006 A fast learning algorithm for deep belief nets Neural Comput. 18 1527
[7] Hinton G E and Salakhutdinov R 2006 Reducing the dimensionality of data with neural networks Science 313 504
[8] Chai H, Liang Y, Wang S and Shen H W 2018 A novel logistic regression model combining semi-supervised learning and active learning for disease classification Sci. Rep. 8 13009
[9] Hornik K, Stinchcombe M and White H 1989 Multilayer feedforward networks are universal approximators Neural Netw. 2 359–66
[10] Kuo-Sheng Cheng K S, Jzu-Sheng Lin J S and Chi-Wu Mao C W 1996 The application of competitive Hopfield neural network to medical image segmentation IEEE Trans. Med. Imaging 15 560–7
[11] Cristianini N and Shawe-Taylor J 2000 An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods (Cambridge: Cambridge University Press)
[12] Suykens J A K and Vandewalle J 1999 Least squares support vector machine classifiers Neural Process. Lett. 9 293–300
[13] Friedman N, Geiger D and Goldszmidt M 1997 Bayesian network classifiers Mach. Learn. 29 131–63
[14] Pan S and Yang Q 2010 A survey on transfer learning IEEE Trans. Knowl. Data Eng. 22 1345–59
[15] Tan B, Zhang Y, Pan S and Yang Q 2017 Distant domain transfer learning Proc. 31st AAAI Conf. on Artificial Intelligence vol 31
[16] Wang Z, Dai Z, Póczos B and Carbonell J 2019 Characterizing and avoiding negative transfer Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition pp 11293–302
[17] Torrey L and Shavlik J 2010 Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques (New York: Hershey) pp 242–64
[18] Taylor M E and Stone P 2009 Transfer learning for reinforcement learning domains: a survey J. Mach. Learn. Res. 10 1633
[19] Tan B, Zhang Y, Pan S and Yang Q 2017 Distant domain transfer learning in Proc. of the 31st AAAI Conf. on Artificial Intelligence 31 p 1
[20] Zhang J 2011 Deep transfer learning via restricted Boltzmann machine for document classification 2011 10th Int. Conf. on Machine Learning and Applications and Workshops pp 323–6
[21] Peruzzo A et al 2014 A variational eigenvalue solver on a photonic quantum processor Nat. Commun. 5 4213
[22] Gu M, Wiesner K, Rieper E and Vedral V 2012 Quantum mechanics can reduce the complexity of classical models Nat. Commun. 3 762
[23] Pudenz K L and Lidar D A 2013 Quantum adiabatic machine learning Quantum Inf. Process. 12 2027
[24] Biamonte J, Wittek P, Pancotti N, Rebentrost P, Wiebe N and Lloyd S 2017 Quantum machine learning Nature 549 195–202
[25] Cai X-D, Wu D, Su Z-E, Chen M-C, Wang X-L, Li L, Liu N-L, Lu C-Y and Pan J-W 2015 Entanglement-based machine learning on a quantum computer Phys. Rev. Lett. 114 110504
[26] Dunjko V, Taylor J M and Briegel H J 2016 Quantum-enhanced machine learning Phys. Rev. Lett. 117 130501
[27] Wossnig L, Zhao Z and Prakash A 2018 Quantum linear system algorithm for dense matrices Phys. Rev. Lett. 120 050502
[28] Rebentrost P, Mohseni M and Lloyd S 2014 Quantum support vector machine for big data classification Phys. Rev. Lett. 113 130503

[29] Li Z, Liu X, Xu N and Du J 2015 Experimental realization of a quantum support vector machine Phys. Rev. Lett. 114 140504

[30] Wiebe N, Kapoor A and Svore K M 2014 Quantum deep learning (arXiv:1412.3489)

[31] Rebentrost P, Bromley T R, Weedbrook C and Lloyd S 2018 Quantum Hopfield neural network Phys. Rev. A 98 042308

[32] Benedetti M, Realpe-Gómez J and Perdomo-Ortiz A 2018 Quantum-assisted Helmholtz machines: a quantum—classical deep learning framework for industrial datasets Quantum Sci. Technol. 3 034007

[33] Cong I, Choi S and Lukin M D 2015 Experimental realization of a quantum support vector machine Phys. Rev. Lett. 114 140504

[34] Wiebe N, Kapoor A and Svore K M 2014 Quantum deep learning (arXiv:1412.3489)

[35] Benedetti M, Realpe-Gómez J and Perdomo-Ortiz A 2018 Quantum-assisted Helmholtz machines: a quantum–classical deep learning framework for industrial datasets Quantum Sci. Technol. 3 034007

[36] Wang Z and Zhang X 2020 Entanglement-based quantum deep learning New J. Phys. 22 033041

[37] Amin M H, Andriyash E, Rolfe J, Kulchytskyy B and Melko R 2018 Quantum Boltzmann machine Phys. Rev. X 8 021050

[38] Kieferova M and Wiebe N 2019 Quantum generative adversarial learning with quantum Boltzmann machines Phys. Rev. A 96 062327

[39] Lloyd S and Weedbrook C 2018 Quantum generative adversarial learning Phys. Rev. Lett. 121 040502

[40] Amin M H 2019 Searching for quantum speedup in quasistatic quantum annealers Phys. Rev. A 92 052323

[41] Nielsen M A and Chuang I 2010 Quantum Computation and Quantum Information (Cambridge: Cambridge University Press)

[42] Cristianini N, Kandola J, Elisseeff A and Shawe-Taylor J 2006 On Kernel Target alignment in Innovations in Machine Learning (Berlin: Springer)

[43] Brown B, Flammia S T and Schuch N 2011 Computational difficulty of computing the density of states Phys. Rev. Lett. 107 040501

[44] Aaronson S 2015 Read the fine print Nat. Phys. 11 291–3

[45] Aaronson S 2010 BQP and the polynomial hierarchy Proc. of the 42nd ACM Symp. on Theory of Computing pp 141–50

[46] Zhao Z, Poyas-Kerstjens A, Rebentrost P and Wittek P 2019 Bayesian deep learning on a quantum computer Quantum Mach. Intell. 1 41–51