Frontiers of Quantum Complexity and Cryptography Spring 2022

Lecture 7 - Learning Unitaries

Lecturer: Henry Yuen *Scribes: Yiming Lin*

1 Overview

Now we switch gears to talk about a task called learning unitaries, and it's uses what we talked about in the previous half of the lecture: those quantum programs, as a sub routine. This is motivated by the the following classical analogue.

2 Classical Analogue

Let's say we want to learn a boolean function $F : \{0,1\}^n \to \{0,1\}^n$, and we only have black box access to this function: we input x, and we get $F(\mathrm{x})$. We would like to learn how it behaves on any input x. If F is completely arbitrary, the only thing we can do is to query this black box every single x possible, requires 2^n queries.

3 Quantum Analogue

Now, instead of having a function F, let's have an arbitrary n-qubit unitary U. It cannot be easier to learn an arbitrary unitary because this unitary is even more general than a function, but let's do something slightly different. What if we want to learn the behavior of this arbitrary unitary on a small dimensional subspace?

Goal: learn behavior of U on a small-dimensional subspace. Specifically, we have the following two phases.

Learning Phase: There is an underlying, unknown unitary U. We're given the following samples:

- Input samples: $|\theta_1\rangle, |\theta_2\rangle, \ldots, |\theta_K\rangle$.
- Output samples: $U|\theta_1\rangle, U|\theta_2\rangle, \ldots, U|\theta_K\rangle$.

In fact, we are given multiple copies of each input/output sample.

Prediction Phase:

- Given a single copy of an unknown input sample, $|\psi\rangle \in span\{|\theta_1\rangle, |\theta_2\rangle, \ldots, |\theta_K\rangle\}$,

- The goal is to output (an approximation of) $U|\psi\rangle$.

So the question is, when can we actually solve this? Normally in linear algebra, it is trivial to compute the output of $U|\psi\rangle$, since we can write $|\psi\rangle$ as a linear combination of the input sample. But in the quantum setting, this is not so easy. We're giving these states in quantum form, and we don't actually "know" what those vectors are, in the sense of having a classical description of them. In quantum physics, having the states in our hands is not the same as knowing them!

If we're given an exponentially many copies of the input sample states, we can perform tomography to get the classical description of the vectors and solve the linear algebra problem. We now try to achieve this with only polynomial number of copies of these input samples.

4 Universal Quantum Emulator of Marvian and Lloyd

Marvian and Lloyd (the same Lloyd from Lloyd-Mohseni-Rebentrost) presented a solution for this unitary learning problem.

Theorem 1 (Universal quantum emulator [1])**.** *There exists a polynomial-time quantum algorithm that for all unitaries n-qubit unitaries U, given input samples $\{|\theta_1\rangle^{\otimes r}, \ldots, |\theta_K\rangle^{\otimes r}\}$, and output samples $\{U|\theta_1\rangle^{\otimes r}, \ldots, U|\theta_K\rangle^{\otimes r}\}$, and (a single copy of an) unknown state $|\psi\rangle \in span\{|\theta_j\rangle\}$, the algorithm outputs a density matrix σ such that*

$$D(\sigma,\ U|\psi\rangle\langle\psi|U^\dagger\ ) \leq O\left(\frac{S^2}{r\Delta^2}\right)$$

where r is number of copies of inputs/output samples, S is dimension of the span of the $\{|\theta\rangle_j\}_j$, and Δ is the spectral gap of the following linear operator on the space of density matrices:

$$\mathcal{E}(\rho) = \frac{1}{K}\sum_{j=1}^{K}\ e^{-i|\theta_j\rangle\langle\theta_j|\pi}\ \rho\ e^{i|\theta_j\rangle\langle\theta_j|\pi}$$

.

Before we explain the linear operator $\mathcal{E}(\cdot)$ and its spectral gap, we encourage the reader to take a step back to appreciate how remarkable this algorithm is. The algorithm starts off knowing *nothing* about this unitary U, which can be arbitrarily complex. Yet given a few quantum examples of inputs and outputs for this unitary, the algorithm can *emulate* the behavior of U on inputs from the space spanned by the input samples. Provided that the dimension of the subspace S is polynomial, the spectral gap Δ is not too small, and the number of copies of r is polynomial, then the algorithm succeeds and is efficient.

Now let's turn to the linear operator $\mathcal{E}(\cdot)$ and its spectral gap. It's an intimidating function, but let's figure out what it means. $\mathcal{E}$ is a function that takes input some density matrix ρ, and it's going to pick one of the $|\theta\rangle_j$ at random, and apply the unitary $e^{-i|\theta_j\rangle\langle\theta_j|\pi}$ to ρ. Note that $e^{-i|\theta_j\rangle\langle\theta_j|\pi}$ is a unitary matrix because the exponent is a Hermitian matrix. This can be interpreted as unitary evolution corresponding to the Hamiltonian $|\theta_j\rangle\langle\theta_j|$ for time π.

Clearly $\mathcal{E}$ is a linear function of ρ. Since the space of matrices forms a linear vector space, we can interpret $\mathcal{E}$ as a linear operator on this vector space. Thus $\mathcal{E}$ has eigenvalues; we can sort them by

$\lambda_1 \geq \lambda_2 \geq \cdots$. The spectral gap Δ is the difference $\Delta := \lambda_1 - \lambda_2$. We note that Δ can be zero if the largest eigenvalue occurs with multiplicity greater than 1.

For the universal quantum emulator algorithm to perform well, we want the spectral gap Δ (which is a number between 0 and 1) to be as large as possible. Intuitively, Δ represents how well the $|\theta_j\rangle$'s span their subspace. A really bad (small) spectral gap would be if the $|\theta_j\rangle$'s are all orthogonal to each other. A good (large) spectral gap is when $|\theta_j\rangle$'s redundantly span their subspace. Let's illustrate this with two examples.

Spectral Gap Example 1: Let's consider the case of $n = 1$ (i.e. we're dealing with a single-qubit state). Let $|\theta_1\rangle = |0\rangle$, $|\theta_2\rangle = |1\rangle$. They are orthogonal, and span $\mathbb{C}^2$. We now compute a closed-form expression for the associated map $\mathcal{E}$.

$$e^{-i|0\rangle\langle 0|\pi} = \mathbb{I} - 2|0\rangle\langle 0| = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} = -Z \ .$$

$$e^{-i|1\rangle\langle 1|\pi} = \mathbb{I} - 2|1\rangle\langle 1| = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = Z \ .$$

$$\mathcal{E}(\rho) = \frac{1}{2}((-Z)\ \rho\ (-Z)) + \frac{1}{2}(Z\ \rho\ Z) = Z\rho Z$$

We claim that there are two inputs that are orthogonal[1] to each other that has eigenvalue of 1:

$$\mathcal{E}(\mathbb{I}) = Z\mathbb{I}Z = \mathbb{I} \ .$$

$$\mathcal{E}(Z) = ZZZ = Z \ .$$

That means the identity matrix and the Pauli Z matrix (which are orthogonal to each other) are eigenvectors of this linear map, and both has eigenvalue 1. Therefore the spectral gap $\Delta = 0$. Bad case.

Spectral Gap Example 2: We consider a different set of states. Let $\{|\theta_1\rangle, |\theta_2\rangle, |\theta_3\rangle, |\theta_4\rangle\} = \{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$. This set forms an overcomplete basis of $\mathbb{C}^2$ – it's redundantly spanned. The spectral gap of the associated $\mathcal{E}$ is non-zero. (Calculation details omitted during class).

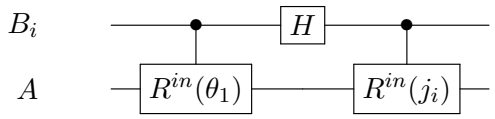
Universal Quantum Emulator Algorithm. Let's finally describe how the Universal Quantum Emulator algorithm works. We won't analyze it, as it is quite involved.

[1]For the vector space of complex matrices, the inner product between two matrices A and B is defined to be $\langle A, B\rangle = \mathrm{Tr}(A^\dagger B)$.

Algorithm 1 Emulator($\{|\theta_j\rangle^{\otimes r}\}, \{(U|\theta_j\rangle)^{\otimes r}\}, |\psi\rangle$)

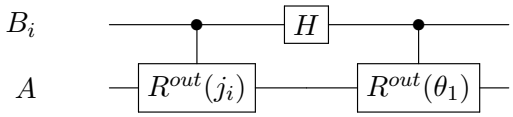
load $|\psi\rangle$ in register A
pick a sequence of indicies $j_1, \ldots, j_T \in [K]$
for $i = 1, \ldots, T$ **do**
  initialize register B_i (which is a single qubit) as state $|-\rangle$
  run Circuit1 on registers B_i and A
end for
Swap $U|\theta_i\rangle$ into register A
for $i = T, \ldots, 1$ **do**
  run Circuit 2 on B_i and A
end for
return the contents of register A

Circuit1:



Define $R^{in}(j) = e^{-i|\theta_j\rangle\langle\theta_j|\pi} = \mathbb{I} - 2|\theta_j\rangle\langle\theta_j|$ (the reflection about θ_j)

Circuit2:



Define $R^{out}(j) = \mathbb{I} - 2U|\theta_j\rangle\langle\theta_j|U^\dagger$.

This algorithm is really remarkable; one of the reasons is that it does not resemble any other quantum algorithm. In the world of quantum algorithms, there are a few templates: Grover search-style algorithms and Quantum Fourier Transform-based algorithms (such as Shor's algorithm). But this algorithm is unlike any of those.

Intuition behind the Universal Quantum Emulator. We won't analyze it formally because it'll take too long, but here some intuition of what it is trying to do. Basically, in the first for loop, its reflecting input state $|\psi\rangle$ around the first input sample state $|\theta_1\rangle$, and then reflecting around a randomly chosen input sample $|\theta_{j_1}\rangle$. It does this reflection controlled on an ancilla qubit in register B_1. It repeats this again, controlled on a new ancilla qubit in register B_2, where the first reflection is about $|\theta_1\rangle$ and the second reflection is about $|\theta_{j_2}\rangle$. It keeps doing this for a number of iterations.

At a high level, these reflection steps in the j'th step are roughly measuring the coordinates of $|\psi\rangle$ relative to the "frame" spanned by $|\theta_1\rangle$ and $|\theta_j\rangle$. These coordinates are stored, in quantum form, in the ancilla qubit in register B_j.

After a while, the register A contains no more information about the state $|\psi\rangle$; its identity has been "erased" from register A and instead distilled into the ancilla qubits in registers $B_1, \ldots, B_T$. In other words, the qubits in registers $B_1, \ldots, B_T$ store information about how the input sample $|\psi\rangle$ is oriented relative to the input samples $|\theta_1\rangle, \ldots, |\theta_K\rangle$. The register A will then contain an approximation of the fixed input sample $|\theta_1\rangle$.

After the first "for" loop, we load $U|\theta_1\rangle$ into register A (swapping out the state $|\theta_1\rangle$ that's sitting there), and run everything in reverse: instead of using input samples, we use output samples. And we're taking all the information about $|\psi\rangle$ stored in the B_i registers and transferring it back to register A, except now all the coordinates relative to the output samples $\{U|\theta_j\rangle\}_j$. In the end, we get, not $|\psi\rangle$, but $U|\psi\rangle$.

The choice of $|\theta_1\rangle$ as the fixed point of this algorithm is arbitrary; in fact the original Marvian-Lloyd paper suggests to pick one at random. This is because it is not guaranteed that $|\theta_1\rangle$ is well-spread within other $|\theta_j\rangle$.

How do we actually implement those controlled reflection gates? This is where we use the LMR algorithm described in the first half of the lecture! These reflection gates takes as input the samples $|\theta_j\rangle$ and $U|\theta_j\rangle$, and LMR can be used to approximate the reflection unitaries $e^{-i|\theta_j\rangle\langle\theta_j|\pi}$ and $e^{-iU|\theta_j\rangle\langle\theta_j|U^\dagger\pi}$.

References

[1] Iman Marvian and Seth Lloyd. Universal quantum emulator, 2016. arXiv:1606.02734.