

Quantum Circuit Learning and SVM

Faris Sbahi

10/28/18

Quantum Circuit Learning

Introduction

- ▶ Approximate any analytical function (with sufficient number of qubits)
- ▶ Hybrid classical-quantum algorithm
- ▶ Parameterize quantum gates by some θ , optimize θ iteratively using gradient descent or the like
- ▶ Low-depth quantum circuit. Goal: realizable near-term

Quantum Circuit Learning

Algorithm

- ▶ Inputs: training data $\{\vec{x}_i\}$ with outputs $\{f(\vec{x}_i)\}$
- ▶ Outputs: $\{y_i\}$ which closely approximates $\{f(\vec{x}_i)\}$.
- ▶ In particular, minimizes some loss function (locally) e.g. quadratic loss for least-squares $\sum_i |y_i - f(\vec{x}_i)|^2$

Quantum Circuit Learning

Algorithm

- ▶ Inputs: training data $\{\vec{x}_i\}$ with outputs $\{f(\vec{x}_i)\}$
 - ▶ Outputs: $\{y_i\}$ which closely approximates $\{f(\vec{x}_i)\}$.
 - ▶ In particular, minimizes some loss function (locally) e.g. quadratic loss for least-squares $\sum_i |y_i - f(\vec{x}_i)|^2$
1. Encode data into quantum state. $|\psi_{in}(\vec{x}_i)\rangle = U(\vec{x}_i)|0\rangle$ for some unitary U
 2. Apply θ -parameterized unitary. $|\psi_{out}(\vec{x}_i)\rangle = U(\theta)|\psi_{in}(\vec{x}_i)\rangle$
 3. Measure expectation of some tensor products of Paulis i.e. some set of operators $\{B_j\} \subseteq \{I, X, Y, Z\}^{\otimes N}$. Output $y_i \equiv g(\{\langle B_j(\vec{x}_i) \rangle\})$ using some function g .
 4. Minimize loss L by tuning θ iteratively
 5. Evaluate L on validation set $\{\vec{v}_i\}$ disjoint from $\{\vec{x}_i\}$

Quantum Circuit Learning

Why QCL?

- ▶ Closed form exists for least-squares (orthogonal projection of vector of y_i 's onto the column space of the data matrix)
- ▶ Hence, we can use HHL. But requires high depth.
- ▶ With QCL, we can solve iteratively and use a short-depth circuit

Quantum Circuit Learning

Example

- ▶ Consider 1-d input $\{x_i\}$
- ▶ Let $\{P_k\} = \{I, X, Y, Z\}^{\otimes N}$
- ▶ Expand $\rho_{in}(x) = |\psi_{in}(x)\rangle \langle \psi_{in}(x)| = \sum_k a_k P_k$
- ▶ Similarly, $\rho_{out}(x) = U(\theta)\rho_{in}(x)U(\theta)^\dagger = \sum_k b_k P_k$
- ▶ $b_m = \sum_k U_{(m,k)} a_k$ since this is a unitary change of basis
- ▶ Therefore, expectation measurement is a linear combination of input coefficients
- ▶ Constraint: row vector $U_{(m,k)}$ has unit norm for fixed m

Quantum Circuit Learning

Example

- ▶ Recall that for a single qubit, $\rho = \frac{1}{2}[I + \vec{a} \cdot \sigma]$, $|\vec{a}| \leq 1$
- ▶ Rotate $\vec{a} = (0, 0, 1)$ about Y axis by some $\theta = \sin^{-1}(x_{in})$ for each of N qubits and call this ρ_{in}
- ▶ $\rho_{in} = \frac{1}{2^N} \otimes_{i=1}^N [I + xX_i + \sqrt{1-x^2}Z_i]$
- ▶ Hence, unitary transformation can give arbitrary N^{th} order polynomial
- ▶ Generalize to d dimensional input

Quantum Circuit Learning

Compute Gradient

- ▶ Assume chain of unitary transformations $U_{l:1}(\theta)$
- ▶ $\langle B(\theta) \rangle = \text{tr}(BU_{l:1}(\theta)\rho_{in}U_{l:1}(\theta)^\dagger)$
- ▶ Assume $U_j(\theta) = \exp(-i\theta P_j/2)$ for pauli product P_j
- ▶ $\frac{\partial \langle B \rangle}{\partial \theta_j} = -\frac{i}{2} \text{tr}\left(BU_{l:1}[B, U_{j-1:1}\rho_{in}U_{j-1:1}^\dagger]U_{l:j}^\dagger\right)$
- ▶ Use identity for commutator of arbitrary ρ with a Pauli product
- ▶ Gives gradient in terms of additional $U(\pm\pi/2)$ transformations

$$\begin{aligned}\frac{\partial \langle B \rangle}{\partial \theta_j} &= \frac{1}{2} \text{Tr} \left[BU_{l:j+1} U_j \left(\frac{\pi}{2} \right) \rho_j U_j^\dagger \left(\frac{\pi}{2} \right) U_{l:j+1}^\dagger \right] \\ &\quad - \frac{1}{2} \text{Tr} \left[BU_{l:j+1} U_j \left(-\frac{\pi}{2} \right) \rho_j U_j^\dagger \left(-\frac{\pi}{2} \right) U_{l:j+1}^\dagger \right]\end{aligned}$$

SVM

Kernels

- ▶ Linear classifier (e.g. perceptron): perform binary classification by separating labels in feature space with hyperplane.
- ▶ Data is not always separable
- ▶ Idea: Map data to higher dimensional space where data is separable
- ▶ Kernel function does this implicitly by providing a means to compute the inner product in that space

Definition 1. Let \mathcal{F} be a Hilbert space, called the feature space, \mathcal{X} an input set and x a sample from the input set. A feature map is a map $\phi : \mathcal{X} \rightarrow \mathcal{F}$ from inputs to vectors in the Hilbert space. The vectors $\phi(x) \in \mathcal{F}$ are called feature vectors.

Feature maps play an important role in machine learning, since they map any type of input data into a space with a well-defined metric. This space is usually of much higher dimension. If the feature map is a nonlinear function it changes the relative position between data points (as in the example of Figure 1), and a dataset can become a lot easier to classify in feature space. Feature maps are intimately connected to kernels [16].

Definition 2. Let \mathcal{X} be a nonempty set, called the input set. A function $\kappa : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{C}$ is called a kernel if the Gram matrix K with entries $K_{m,m'} = \kappa(x^m, x^{m'})$ is positive semidefinite, in other words, if for any finite subset $\{x^1, \dots, x^M\} \subseteq \mathcal{X}$ with $M \geq 2$ and $c_1, \dots, c_M \in \mathbb{C}$,

$$\sum_{m,m'=1}^M c_m c_{m'}^* \kappa(x^m, x^{m'}) \geq 0.$$

By definition of the inner product, every feature map gives rise to a kernel.

SVM

Kernels

Theorem 1. *Let $\phi : \mathcal{X} \rightarrow \mathcal{F}$ be a feature map. The inner product of two inputs mapped to feature space defines a kernel via*

$$\kappa(x, x') := \langle \phi(x), \phi(x') \rangle_{\mathcal{F}}, \quad (1)$$

where $\langle \cdot, \cdot \rangle_{\mathcal{F}}$ is the inner product defined on \mathcal{F} .

Proof. We must show that the Gram matrix of this kernel is positive definite. For arbitrary $c_m, c_{m'} \in \mathbb{C}$ and any $\{x^1, \dots, x^M\} \subseteq \mathcal{X}$ with $M \geq 2$, we find that

$$\begin{aligned} \sum_{m, m'=1}^M c_m c_{m'}^* \kappa(x_m, x_{m'}) &= \left\langle \sum_m c_m \phi(x_m), \sum_{m'} c_{m'} \phi(x_{m'}) \right\rangle \\ &= \left\| \sum_m c_m \phi(x_m) \right\|^2 \geq 0 \end{aligned}$$

□

Definition 3. Let \mathcal{X} be a non-empty input set and \mathcal{R} a Hilbert space of functions $f : \mathcal{X} \rightarrow \mathbb{C}$ that map inputs to the real numbers. Let $\langle \cdot, \cdot \rangle$ be an inner product defined on \mathcal{R} (which gives rise to a norm via $\|f\| = \sqrt{\langle f, f \rangle}$). \mathcal{R} is a reproducing kernel Hilbert space if every point evaluation is a continuous functional $F : f \rightarrow f(x)$ for all $x \in \mathcal{X}$. This is equivalent to the condition that there exists a function $\kappa : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{C}$ for which

$$\langle f, \kappa(x, \cdot) \rangle = f(x) \quad (2)$$

with $\kappa(x, \cdot) \in \mathcal{R}$ and for all $f \in \mathcal{H}$, $x \in \mathcal{X}$.

- ▶ Can construct unique RKHS for any feature map
- ▶ Mercer's Theorem: If kernel k is positive, we can expand k (as a uniformly convergent series) in terms of eigenfunctions and eigenvalues of a positive operator that come from k .
- ▶ Representer Theorem: Even if we were trying to solve an optimization problem in an infinite dimensional space H_k containing linear combinations of kernels centered on arbitrary x_i 's, then the solution lies in the span of the n kernels centered on the x_i s.

SVM

Optimization – Dual Formulation

$$\begin{aligned}\text{maximize } f(c_1 \dots c_n) &= \sum_{i=1}^n c_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i c_i (\varphi(\vec{x}_i) \cdot \varphi(\vec{x}_j)) y_j c_j \\ &= \sum_{i=1}^n c_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i c_i k(\vec{x}_i, \vec{x}_j) y_j c_j\end{aligned}$$

subject to $\sum_{i=1}^n c_i y_i = 0$, and $0 \leq c_i \leq \frac{1}{2n\lambda}$ for all i .

- ▶ Use quadratic programming, coordinate descent, gradient descent, SMO...
- ▶ Representer's Theorem
- ▶ Only ever need to evaluate inner product in RKHS to train and test model

SVM

Explicit Classification

- ▶ Encode data, just as with QCL. This time, encode data under feature map transformation directly.
- ▶ $U_{\phi}(x) |0\rangle^N = |\Phi(x)\rangle$
- ▶ Hence, $K(x, z) = |\langle \phi(x) | \phi(z) \rangle|^2$
- ▶ Map must be hard to compute classically. Simple kernel like RBF allows efficient classical computation in infinite-dimensional Hilbert space. Large quantum Hilbert space is not enough.

- ▶ Consider feature map $U_{\phi}(x) = V_{\phi(x)} H^{\otimes n} V_{\phi(x)} H^{\otimes n}$ where

$$V_{\phi(x)} = \exp\left(i \sum_{S \subseteq [n]} \phi_S(x) \prod_{i \in S} Z_i\right), |S| \leq 2$$
- ▶ Conjecture: Classical evaluation of inner products generated from circuits with two basis changes and diagonal gates up to additive error is "hard"

- ▶ Consider feature map $U_{\phi}(x) = V_{\phi(x)} H^{\otimes n} V_{\phi(x)} H^{\otimes n}$ where $V_{\phi(x)} = \exp\left(i \sum_{S \subseteq [n]} \phi_S(x) \prod_{i \in S} Z_i\right)$, $|S| \leq 2$
- ▶ Conjecture: Classical evaluation of inner products generated from circuits with two basis changes and diagonal gates up to additive error is "hard"
- ▶ $n = d = 2$ qubits
- ▶ $\phi_{\{i\}}(\vec{x}) = x_i$ and $\phi_{\{1,2\}} = (\pi - x_1)(\pi - x_2)$ with $\vec{x} \in (0, 2\pi]^2$
- ▶ Take some random unitary $V \in SU(4)$ and defined data "gap" $\Delta = 0.3$.
- ▶ Then, we define $\langle \Phi(x) | V^\dagger Z_1 Z_2 V | \Phi(x) \rangle \geq \Delta \implies +1$ label and -1 otherwise

SVM

Havlicek et al

- ▶ Now, apply θ -parameterized layers of l unitaries. $\theta \in \mathbb{R}^{2N(l+1)}$
- ▶ For binary classification, $y \in \{+1, -1\}$.
- ▶ In our case, $\vec{f} = Z_1 Z_2$ and $0 \leq l \leq 4$
- ▶ Measure $M_y = \frac{1}{2}(I + y\vec{f})$ with $\vec{f} = \sum_{z \in \{0,1\}^N} f(z) |z\rangle \langle z|$ and $f : \{0,1\}^n \rightarrow \{+1, -1\}$
- ▶ Obtain empirical distribution $p_y(x)$ which is expectation of M_y sampled with repeated shots
- ▶ Classify by comparing $p_y(+1)$ to $p_{-1}(x)$ (can add bias as well)
- ▶ Define loss function to be average number of misclassifications

SVM

Havlicek et al

- ▶ How is this SVM?
- ▶ Decision rule for label:
$$m(x) = \text{sgn}(\langle \Phi(x) | W^\dagger(\theta) \vec{f} W(\theta) | \Phi(x) \rangle)$$
- ▶ Decompose density operator ρ_{in} and ρ_{out} in *Pauli* product basis $\{P_\alpha\}$.
- ▶ Expectation value of binary measurement and decision rule can be expressed in terms of $w_\alpha(\theta) = \text{tr}[W^\dagger(\theta) \vec{f} W(\theta) P_\alpha]$ and $\Phi_\alpha(x) = \langle \Phi(x) | P_\alpha | \Phi(x) \rangle$
- ▶ $m(x) = \text{sgn}(1/2^N \sum_\alpha w_\alpha(\theta) \Phi_\alpha(x))$. SVM separating hyperplane.
- ▶ Decomposition shows we should think of $|\phi(x)\rangle \langle \phi(x)|$ as the feature vectors i.e. feature vectors don't live directly in the Hilbert space $H = (\mathbb{C}^2)^{\otimes N}$. As expected, since global phase would make this problematic.