DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Quantum Computing

Quantum and Classical Generative Modeling for Quantum States Preparation

Wiktor Jurasz

DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Quantum Computing

Quantum and Classical Generative Modeling for Quantum States Preparation

Quantenbasierte und klassische generative Modellierung zur Erzeugung von Quantenzuständen

Author:	Wiktor Jurasz
Supervisor:	Prof. Dr. Christian Mendl
Submission Date:	15.06.2021

I confirm that this master's thesis in quantum computing is my own work and I have documented all sources and material used.

Munich, 15.06.2021 Wiktor Jurasz

Acknowledgments

Abstract

Kurzfassung

Contents

1. Introduction

1.1. Problem Statement

Generative Modeling aims to learn a conditional probability $P(X|Z = z)$, where X is some observable variable and Z is a target variable. With knowledge of this conditional probability, it is possible to generate new observations $\bar{x} \in X$. In general case, one would not try to obtain the probability $P(X|Z)$ exactly, but learn an approximation. To do so a set of samples $x \in X$ is necessary to train a generator function $G : Z \to X$ which given a target variable $z \in Z$ generates new observation $x \in X$.

In the generative framework, the variable X is a multidimensional vector, in particular it can be used to describe an arbitrary quantum state. With this setup, given a finite set of quantum states $\mathcal{Q} = \{|\psi_i\rangle\}, |\psi_i\rangle \in X \forall i$ the generator function G prepares a new quantum state $|\hat{\psi}\rangle$. This new quantum state is expected to come from the same distribution as the samples in the input set $\mathcal{Q}$.

The only missing piece in the above description is the target variable Z. In the context of the function G, generating the quantum states, we can think about Z as a label of the generated state. That is, for a specific $z \in Z$ the function G always generates the same $|\hat{\psi}\rangle$.

In this work we evaluate different approaches to find the probability $P(X|Z = z)$ by learning the function G. We also address the limitations of the existing methods propose a new one that combines quantum and classical generative modeling.

1.2. Previous Work

There exist many different types of generative models. In this work we focus on one particular type, namely Generative Adversarial Networks (GANs). First version of GANs was proposed bu Goodfellow et al. **[goodfellow2014generative]** (to which we refer as Standard GANs - SGANs), since then many different variations of GANs were invented **[mirza2014conditional][karras2019stylebased][radford2016unsupervised]**. In context of this work, particularly interesting are Wasserstein GANs (WGANs)**[arjovsky2017wasserstein]** which minimize *Earth-Mover* distance between two probability distribution (see Chapter 3) instead of *Jensen–Shannon* divergence (see Chapter 3) as in SGANs.

In recent years there has been an increasing interest in realizing Generative Adversarial Networks in Quantum Computing (QC) realm. Dallaire-Demers et al. proposed QuGANs **[Dallaire_Demers_2018]** - Quantum Generative Adversarial Networks where generator and discriminator are parametrized quantum circuits. Similarly Benedetti et al. proposed fully quantum GANs for pure state approximation **[Benedetti_2019]**, but with different (more

suitable for NISQ [**Preskill_2018**]) learning method. Hybrid methods were also explored, Zoufal et al. build qGAN [**Zoufal_2019**] - with parametrized quantum circuit as the generator and classical neural network as the discriminator.

De Palma et al. proposed quantum equivalent of Wasserstein distance of order 1 [**depalma2020quantum**] which made the Quantum Wasserstein GANs (QWGANs) [**kiani2021quantum**] possible. This variation of quantum GANs consist of the parametrized quantum circuit as the generator and the classical linear program as the discriminator.

1.3. Our Contribution

There has been a substantial effort in the direction of bringing GANs into the quantum realm. Nevertheless, this is still very early stage and many more routs are yet to be explored. In this work we focus on building quantum GANs that can generate new, unseen before, quantum states. Majority of models proposed so far are only able to generate the states the has been a part of the training data. Only some architectures [**Dallaire_Demers_2018**] account for random noise in the input that allows to generate unseen states. However, as we discuss later, those are mostly theoretical and do not seem to work well in practice.

We propose a new quantum-classical hybrid approach that allows to generate an unlimited number of unseen quantum states. We utilize the fully quantum and fully classical generative models that work together in one framework.

We proceed as follows. In Chapter 2 we introduce briefly the quantum computing concepts necessary to understand the problems we are solving. We also establish the quantum computing notation used in the reminder of this work. In Chapter 3 we give a general introduction to classical GANs. In Chapter 4 we combine the knowledge from the previous chapters to introduce the concept of quantum GANs and talk more about the different variations of quantum GANs and their limitations. In Chapter 5 we introduce and describe in depth about our concept of hybrid quantum-classical generative framework. In Chapter **??** we and analyse numerous experiments with the proposed framework. We empirically prove the quality of the states generated by the proposed framework. Finally, in Chapter 6 we conclude our finding and talk briefly about the possible future directions.

2. Quantum Computing Introduction

In this chapter we provide a very brief introduction to the key concepts of quantum computing and introduce the notation used in the rest of this paper.

2.1. Parametric Circuits

3. Generative Adversarial Networks Introduction

Generative Adversarial Network (GAN)[**goodfellow2014generative**] is a machine learning framework designed to estimate generative models using adversarial process. At the core it consists of two models: the generative one G which is capable of learning the distribution of the provided data and discriminative model D that, given a data point, estimates whether it comes from input data or was generated by G. The models are set to compete with each other in a minmax game. D is trained to maximize the probability of correctly distinguishing between the generated and real samples, while G is trained to minimize it.

To approximate the true distribution p_r over data X we define a prior noise distribution p_z over noise input Z and the generator distribution p_g over data X. The generator represents a mapping $G(z \sim Z; \theta_g) \rightarrow x \sim X$, where θ_g is some learnable parameter (summary in Table 3.1).

The discriminator $D(x, \theta_d) \rightarrow [0; 1]$ is a function that given a sample $x \sim X$ outputs the probability whether x comes from data or was generated by G.

In classical GANs both G and D are most often modeled as multi-layer perceptrons[**goodfellow2014generati** or other types of neural networks (e.g. convolutional neural networks [**radford2016unsupervised**]). However, there exist many different variations of cost functions used during the mixmax training procedure. In the following paragraphs we take a closer look at two of them, namely SGANs and WGANs, which are the most relevant in the context of this work. To simply the notation, we skip the θ parameters, i.e. $G(z, \theta_g) = G(z)$ and $D(x, \theta_d) = D(x)$

3.1. Standard Generative Adversarial Networks (SGANs)

The goal of D is to distinguish between the real and generated samples. For $x \sim X$ the output $D(x)$ should approach 1, while for $x \sim G(z)$ it should approach 0. In other terms it simultaneously maximizes $\mathbb{E}_{x \sim p_r(x)}[\log D(x)]$, and $\mathbb{E}_{z \sim p_z(z)}[\log 1 - D(G(z))]$.

Generator G has an opposite objective, thus it minimizes the $\mathbb{E}_{z \sim p_z(z)}[\log 1 - D(G(z))]$.

Probability Distribution	Description
p_z	True distribution over noise input Z
p_g	Approximated distribution over input data X given by G
p_r	True distribution over input data X

Table 3.1.: Description of the probability distributions used in GANs

Putting it all together, we get the loss function for SGANs.

$$\begin{aligned}\min_G \max_D \mathcal{L}(\mathcal{G}, \mathcal{D}) &= \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log 1 - D(G(z))] \\ &= \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{x\sim p_g(x)}[\log 1 - D(x)]\end{aligned} \tag{3.1}$$

It can be shown, that if D is optimal, this loss function measures the similarity between p_r and p_g according to Jensen–Shannon divergence (see Appendix A.1 for detailed analysis).

SGANs are a very powerful tool, however, the training process is very difficult and unstable [**salimans2016improved**]. Additionally, if the data exists in low dimensional manifold (which seems to be the case for most real world data [**narayanan2010proceedings**]), p_r and p_g are very likely disjoint and we are always capable of finding the perfect discriminator [**arjovsky2017principled**]. This might seem like a good characteristic, but if we examine the loss function closer, we find that it makes generator incapable of learning. For the perfect D we have $\forall x \sim X, D(x) = 1$ and $\forall z \sim Z, D(G(z)) = 0$. For those values $\mathcal{L}(\mathcal{G}, \mathcal{D}) = 0$ gradient vanishes and the gradient based optimizers cannot improve the generator anymore.

3.2. Wasserstein Generative Adversarial Networks (WGANs)

One of the very prominent improvement to SGANs and a way to mitigate the vanishing gradient problem is changing the cost function to use Wasserstein Distance.

The Wasserstein Distance, also called Earth-Mover's Distance is a distance measure between two probability distributions. The name "Earth Mover" comes from the fact, that informally the distance can be described as follows: Given two probability distributions, if we imagine them as piles of dirt, the "Earth Mover" distance says what is the minimum cost of turning one pile of dirt into the other one. Where cost is the volume that has to be moved times the distance is has to be moved.

The main advantage of the Wasserstein Distance over Jensen–Shannon Divergence is, that even in the case of disjoint distribution we get meaningful, stable distance which is well suited for gradient based learning.

Formally, the Wasserstein or Earth-Mover's Distance (EM Distance) is defined as follows:

$$W(p_r, p_g) = \inf_{\gamma\in\Pi(p_r,p_g)} \mathbb{E}_{(x,y)\sim\gamma}[\|x - y\|] \tag{3.2}$$

Where $\Pi(p_r, p_g)$ denotes the set of all possible joint distributions, $\gamma(x, y)$ says how much "dirt" has to be transported in order to transform p_g into p_r. Since we take inf, the EM Distance is then the cost of such transport done in the optimal way.

The computation in 3.2 is highly intractable, but according to Kantorovich-Rubinstein duality [**villani@optimal**], we can rewrite it as follows.

$$W(p_r, p_g) = \frac{1}{K} \sup_{\|f\|_L<K} \mathbb{E}_{x\sim p_r}[f(x)] - \mathbb{E}_{x\sim p_g}[f(x)] \tag{3.3}$$

Where the supremum is overall all K-Lipschitz functions f. In practice it is impossible to go over all such functions. Instead we define a family of parameterized functions $\{f_w\}_{w \in W}$ and using the same notation for the generator as in SGANs the WGANs loss function becomes

$$\mathcal{L}(f, G) = \max_w \min_\theta \mathbb{E}_{x \sim p_r}[f_w(x)] - \mathbb{E}_{z \sim p_z}[f_w(G_\theta(z))] \tag{3.4}$$

.

Here the function f_w represents the "discriminator", however it does not anymore tries to distinguish between real and fake samples. Instead it is trained to to approximate K-continous Lipschitz function and compute the Wasserstein distance. The one missing part from the Equation 3.4 is ensuring the f_w Lipschitz continuity, this can be achieved by gradient clipping [**arjovsky2017wasserstein**] or gradient penalty [**gulrajani2017improved**].

4. Quantum Generative Adversarial Networks

The field of Quantum Machine Learning (QML) is still in very early stages and there has been an ongoing effort on translating the classical Machine Learning (ML) concepts into QML realm. Because of the limitations of current quantum computers (NISQ) [**bharti2021noisy**] and overall different paradigm of Quantum Computing (QC), this process is difficult and there is no clear answer to the question "how this concept should be realized on QC device?". While in classical GANs generator and discriminator are realized as deep neural networks, NISQ devices are not yet powerful enough to support such architecture. Instead parametric quantum circuits [**Schuld_2020**] are used.

In this chapter we take a closer look at two different designs of quantum GANs. We briefly explain theory behind them and show the results of our evaluation. The quantum GANs introduced in this chapter are the base of our hybrid classical-quantum generative framework.

4.1. Standard Quantum GANs (SQGANs)

Theoretical description and experimental results

4.2. Wasserstein Quantum GANs (WQGANs)

Theoretical description and experimental results

5. Unknown Quantum State Generation

5.1. Labeled State Generation

5.2. Unlabeled State Generation

6. Conclusions

A. Appendix

A.1. SGANs approximate Jensen–Shannon Divergence

B. Figures

List of Figures

List of Tables