A Mean-Field Optimal Control Formulation of GANs

Yirong Bian

New York University Shanghai

May. 9th

Outline

Introduction

Generative Model Example

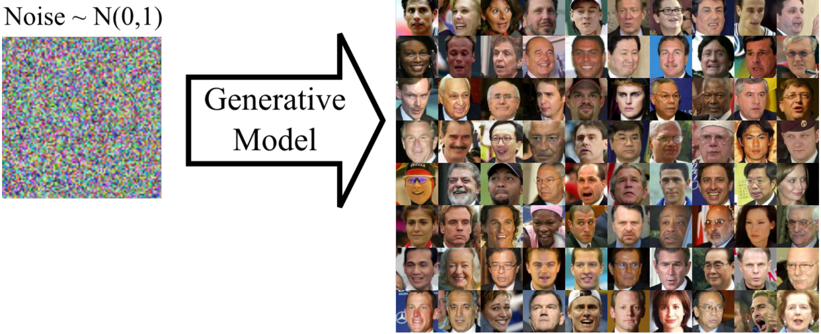


Figure 1: Generative model to generate human faces from a prior noise distribution

Introduction

- Generative modeling is an unsupervised learning task that can be used to generate or output new examples that plausibly could have been drawn from the original dataset.
- Despite the empirical success of generative adversarial networks (GANs), a solid mathematical framework is needed to provide a theoretical interpretation of its mechanisms.
- Mean field optimal control theory framework is a candidate.

Background

GAN-Definition

- The GAN algorithm contains two neural networks, generator G and discriminator D, with parameters θ^G, θ^D.
- We denote the true data distribution by p_x, and define a prior noise distribution to be p_z.

$$\min_{\theta^G} \max_{\theta^D} V(\theta^D, \theta^G) = \mathbb{E}_{x \sim p_x}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z)))]$$

resNN-Definition

- We assume the data is drawn from a joint probability distribution $(x_0, y_0) \sim \mu$, For a N_T layered resNN, we have the feed-forward propagation difference equation:

$$x_{t+1} = x_t + f(x_t, \theta_t), \quad t = 0, 1, 2, ..., N_T - 1$$

- Continuous-time idealization:

$$\dot{x}_t = f(x_t, \theta_t),\ x(0) = x_0 \quad t \in [0, T]$$

resNN, continuous time

Problem: The deep learning problem to minimize the population risk can be structured as:

$$J(\boldsymbol{\theta}) = \mathbb{E}_{(x_0,y_0)\sim\mu}\left[\Phi(x_T, y_0) + \int_0^T L(x_t, \theta_t)\, dt\right], \quad \boldsymbol{\theta} \in L^\infty([0,T], \Theta)$$

Remark

- The mean field emphasizes that the control itself is deterministic and only depends on the distribution.
- θ_t is an open-loop control which means that it only depends on t.
- In other words, the parameters in each layer are fixed numbers.

Mathematical Formulation

GAN with two resNNs

We will present the optimal control formulation of the GAN deep learning algorithm composed by two residual neural networks (resNN).

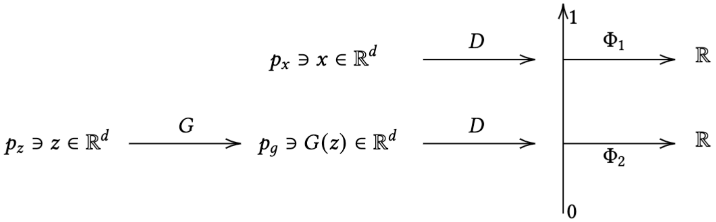


Figure 2: Process of Points in the Algorithm

Definitions

Definition

We stack the true data sample x and the noise prior one z as a single input X drawn from the joint distribution μ.

$$X_0 := (x_0, z_0)^T \sim \mu = (p_x, p_z)^T, \quad X_t := (x_t, z_t)$$

Assume g, f to be the feed-forward rule for the generator of layer T_1 and the discriminator of layer T_2 respectively. Then

$$\begin{aligned} \dot{z}_t &= g(z_t, \theta_t^G) \quad 0 < t < T_1, \quad g : \mathbb{R}^d \times \Theta^G \mapsto \mathbb{R}^d \\ \dot{x}_t &= f(x_t, \theta_t^G) \quad T_1 \leq t \leq T_2, \quad f : \mathbb{R}^d \times \Theta^D \mapsto \mathbb{R}^d \end{aligned}$$

$$\theta_t := (\theta_t^G, \theta_t^D)^T \in \Theta.$$

GAN Problem

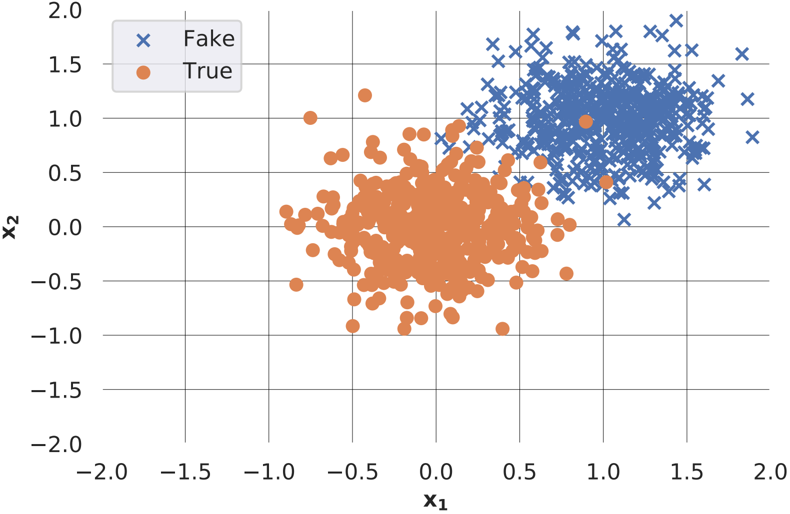
Problem

Given the terminal loss function Φ, the running cost or regularization function L and the distribution μ, write the terminal time $T_1 + T_2 = T$, we want to find the optimal for the following minimax problem:

$$J(t,\mu,\theta) = \min_{\theta^G \in L^\infty([0,T_1)\times\Theta)} \max_{\theta^D \in L^\infty([T_1,T_2]\times\Theta)} \mathbb{E}_{X\sim\mu}[\Phi(X_T) + \int_t^T L(X_s,\theta_s)ds] \tag{1}$$

Experiments

Normal Distribution experiment



Algorithms

- Generator: 20 layer residual Neural network, with activation function tanh.

$$G : \mathbb{R}^2 \mapsto \mathbb{R}^2$$

- Discriminator: 20 layer residual neural network with activation function tanh, last fixed layer with Sigmoid.

$$D : \mathbb{R}^2 \mapsto \mathbb{R}$$

- Why 20 layer?

Algorithms

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets.

Denote $\theta_g = (\theta_1^G, ..., \theta_{N_g}^G)$; $\theta_d = (\theta_1^D, ..., \theta_{N_d}^D)$
Parameters: Learning rate of generator and discriminator: α_g, α_d

for number of iteration steps **do**

- Sample minibatch of m noise samples $\{z_1, ..., z_m\}$ from prior input noise distribution p_z
- Sample minibatch of m true examples $\{x_1, ..., x_m\}$ from true data distribution p_x

Gradient Descent for θ_d **/ Ascent** θ_g
end for

Definition of Negative log-generated-likelihood

- Since the true distribution is known, we will define the generated likelihood from a fixed noise input $z_1, z_2, ..., z_n$ as:

$$\mathcal{L}(\theta^G) = \prod_{i=1}^{n} f\left(G_{\theta^G}(z_i)\right),$$

  f is the probability density function of true distribution.
- Use negative log-generated-likelihood to measure the success of the algorithm.

Results

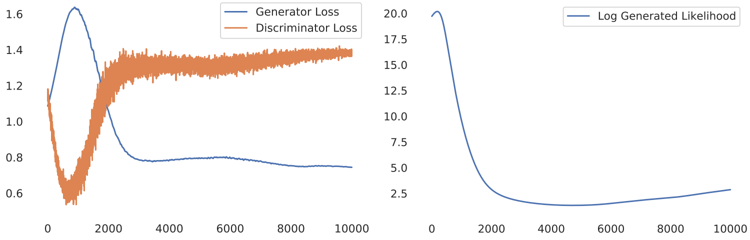


Figure 4: Loss, negative log-generated likelihood over iteration

Results

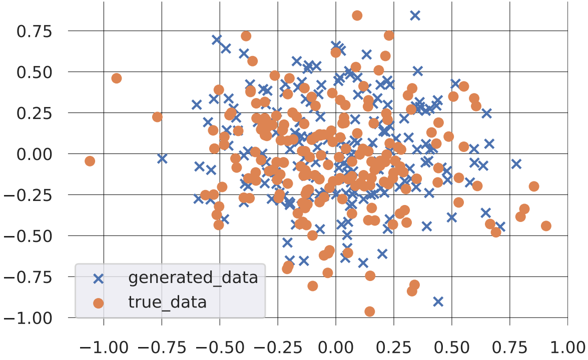


Figure 5: Generated Data vs. True Data

Results

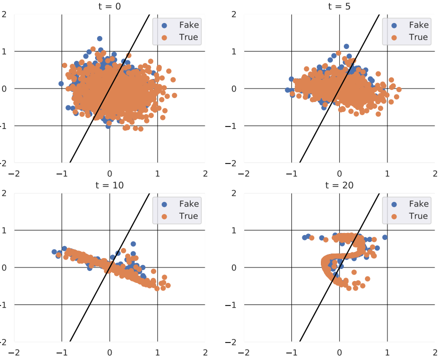


Figure 6: Evolution of Points through Discriminator

Results

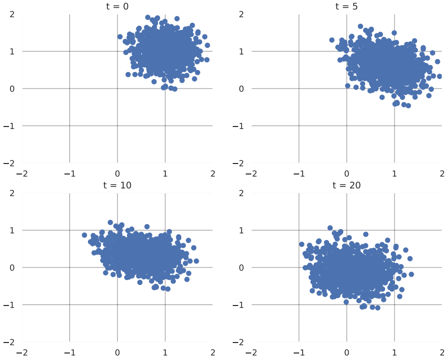


Figure 7: Evolution of Points through Generator

Conclusion

Conclusion

- We introduce a mathematical formulation of the GAN algorithm in terms of the mean field optimal control theory.
- Experiments on imitation of normal distribution and uniform distribution are done to provide empirical understanding on the training strategy.
- The evolution of points under the control of both the discriminator and the generator is visualized to understand the resNN from the perspective of dynamical systems.

Fin