Adversarial Stein Learning for Generating Point Clouds

Group name: No Name

Shayan Shekarforoush
1007025390
shayan@cs.toronto.edu

Mohammadreza Salehi
1006890081
mrsalehi@cs.toronto.edu

1 Abstract

In this work, by considering point clouds as probability distributions where higher likelihoods is assigned to the regions near the surface, we learn a generative model that approximates the gradient of the log probability density of these underlying distributions. We train the model by minimizing the Stein Discrepancy between the data and the model's distribution in an adversarial setting. Finally, by drawing samples using the Langevin dynamics technique, we generate point clouds which are then compared to the ground truth for evaluation.

2 Technical ideas

Modeling shapes as distributions. Each 2D or 3D shape can be represented as a set of points sampled from its surface, denoted by $\{x_i\}_{i=1}^N$, with $x_i \in \mathbb{R}^d$ ($d = 2, 3$). Let P_s be the uniform distribution over the points on the surface of a shape. In theory, this distribution typically covers a subspace with zero support, which makes it challenging to learn in practice. To alleviate this problem, following [1], we adjust the distribution with a Gaussian kernel which results in

$$Q_\sigma(x) = \int_{x' \in \mathcal{X}} P_s(x')\mathcal{N}(x; x', \sigma^2 I)dx',$$

where $\mathcal{X}$ shows the ambient space such as $\mathbb{R}^2$ or $\mathbb{R}^3$. This smoothing is also more realistic as the obtained point clouds are noisy samples from the surface of interest.

Adversarial stein learning. As mentioned earlier, we would like to learn a model distribution P_θ which approximates the true data distribution Q_σ defined above. To achieve this, similar to [1], we directly predict the gradient of log probability, i.e. $\nabla_x \log P_\theta$. However, our training objective is different than [1], where take an approach similar to [2, 3] and compare the distributions by using the Stein Discrepancy [4] measure as follows:

$$D_{\text{Stein}}(Q_\sigma || P_\theta) = \sup_{f \in \mathcal{F}} \mathbb{E}_{x \sim Q_\sigma}[(\nabla_x \log P_\theta(x))^T f(x) + \text{Tr}(\nabla_x f(x))],$$

where f is selected from a family functions mapping from $\mathbb{R}^d$ to $\mathbb{R}^d$, thus $\nabla_x f(x)$ is a $d \times d$ Jacobian matrix which is can be computed easily in low dimensional spaces (e.g. $d = 2, 3$). Moreover, following [1, 2, 3], we parameterize $\nabla_x \log P_\theta$ as a highly flexible neural network $g_\theta(x)$. Finally, minimizing the discrepancy can be rendered as the following MinMax optimization problem:

$$\min_\theta D_{\text{Stein}}(Q_\sigma || P_\theta) = \min_\theta \max_{f \in \mathcal{F}} \mathbb{E}_{x \sim Q_\sigma}[g_\theta(x)^T f(x) + \text{Tr}(\nabla_x f(x))].$$

In the above, the inner optimization estimates the discrepancy and the outer one attempts to minimize it. Therefore, to solve the optimization problem, we take an alternating procedure similar to those utilized for training a GAN [5]. To maintain a valid divergence [4] and estimate stein discrepancy [2, 3], we parameterize f, the critic, as a neural network with a bounded output. Multiple techniques can be employed to constrain the output of this critic network such as spectral normalization [6], or regularizing norm of output [2]. Moreover, to compute the objective, an expectation over Q_σ is required which can obtained by following Monte Carlo approximation:

$$\mathbb{E}_{x\sim Q_\sigma}[(\nabla_x \log P_\theta)^T f(x)+\text{Tr}(\nabla_x f(x))] \approx \sum_{i=1}^{N}(\nabla_x \log P_\theta(x_i+\epsilon_i))^T f(x_i+\epsilon_i)+\text{Tr}(\nabla_x f(x_i+\epsilon_i)),$$

where $\epsilon_i \sim \mathcal{N}(0, \sigma^2 I)$. As long as σ is small, the set of points $\{x_i + \epsilon_i\}_{i=1}^N$ that we optimize on, are concentrated near the surface. Hence, we do not get much supervision about the points that lie far away from the surface, and, in turn, their predicted gradient of log density is not reliable. As this gradient is used during sampling, final retrieved surface can be negatively affected. To sidestep this issue, we follow previous work [1, 7] and take multiple values for σ_i, $1 \leq i \leq M$, and jointly learn $\nabla_x \log P_\theta$ conditioned on σ_i.

Sampling. Supposing that we learned the model distribution for all $\sigma_1 \geq \sigma_2 \cdots \geq \sigma_k$, we leverage an annealed version of Langevin dynamics, firstly proposed by [7], and similarly utilized by [1], to draw samples from the learned distribution as follows

$$x'_{t+1} = x_t + \sqrt{\alpha}\frac{\sigma_i}{\sigma_k}\epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, I)$$

$$x_{t+1} = x'_{t+1} + \alpha\frac{\sigma_i^2}{2\sigma_k^2}\nabla_x \log P_\theta(x'_{t+1}, \sigma_i).$$

3 Goals

Viable objectives. We would like to evaluate our model on a recently proposed dataset called MNIST-CP [8] which consists of points on the 2D contours of MNIST digits. For each individual digit, we adversarially train its specific neural network g_θ to capture the score function of the distribution representing the surface of the digit. To assess the fidelity of point cloud reconstruction, we evaluate some well-known metrics in the literature such as symmetric Chamfer Distance and Earth's Mover Distance. We also compare our results with a variant of previous work [1] which learns the shape distribution using *Denoising Score Matching*. Furthermore, since ideally the gradient is almost zero on surface, our model allows us to extract an implicit representation of the surface [1]. Thereby, we can also qualitatively compare reconstructions by visualizing the obtained surface.

Nice-to-haves. If everything works well, in an attempt to make our model generalizable to many shapes, we will incorporate an encoder to infer latent codes for different shapes and condition the decoder on this code. Moreover, we may experiment on the more challenging ShapeNet dataset [9].

4 Related work

Energy-based models (EBM) have gained huge traction during the past two years. In general, it is challenging to train EBMs using maximum likelihood estimation (MLE), because of the presence of an intractable normalizing factor in the MLE objective. [10] reviews various methods that can be used for efficient training of EBMs. The most well-known method is estimating the gradients through MCMC sampling. So far, lots of techniques have been proposed to make this sampling process more efficient and Langevin MCMC [11, 12] is one of the most important ones. The main advantage of this method is that it is based on the gradient of the energy, thus gets rid of the intractable normalizing factor. In literature, this gradient is termed the *score function* and is the foundation of many other methods for training EBMs, such as Score Matching (SM) and Denoising Score Matching (DNS).

[1] is the main paper on which we base our work. In this work, the authors have viewed each shape as a probability distribution. Moreover, this is the first work that models the gradient of log density for generating shapes. With this gradient at hand, one can start sampling from a simple prior distribution and move the sampled points in the direction of gradient to reach the desired target shape. This is achieved iteratively using a new version of Langevin dynamics [1, 7]. We take the same approach to generate samples in our work. However, as stated in the previous section, unlike [1] that minimizes L2 loss between the gradient of model and data distribution, we use a different objective function, Stein Discrepancy, to adversarially train the model.

Parameterizing the critic in the stein discrepancy is another important topic related to our work. According to [4], the space of functions $\mathcal{F}$ has to be bounded and [2] accomplishes this by adding a regularization to training objective in order to encourage finite expectation of squared norm of the output under the data distribution. We similarly constrain the critic network.

References

[1] Ruojin Cai, Guandao Yang, Hadar Averbuch-Elor, Zekun Hao, Serge Belongie, Noah Snavely, and Bharath Hariharan. Learning gradient fields for shape generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[2] Will Grathwohl, Kuan-Chieh Wang, Jorn-Henrik Jacobsen, David Duvenaud, and Richard Zemel. Learning the stein discrepancy for training and evaluating energy-based models without sampling, 2020.

[3] Shiv Shankar. Adversarial stein training for graph energy models, 2020.

[4] Jackson Gorham and Lester Mackey. Measuring sample quality with stein's method, 2019.

[5] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[6] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks, 2018.

[7] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution, 2020.

[8] Wang Yifan, Shihao Wu, Hui Huang, Daniel Cohen-Or, and Olga Sorkine-Hornung. Patch-based progressive 3d point set upsampling, 2019.

[9] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository, 2015.

[10] Yang Song and Diederik P. Kingma. How to train your energy-based models, 2021.

[11] G. Parisi. Correlation functions and computer simulations. *Nuclear Physics B*, 180(3):378–384, 1981.

[12] Ulf Grenander and Michael I. Miller. Representations of knowledge in complex systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 56(4):549–603, 1994.