Denoising Likelihood Score Matching for Conditional Score-based Data Generation

Chen-Hao Chao[1,2], Wei-Fang Sun[1], Bo-Wun Cheng[1,2], Yi-Chen Lo[2], Chia-Che Chang[2], Yu-Lun Liu[2], Yu-Lin Chang[2], Chia-Ping Chen[2], and Chun-Yi Lee[1]

[1] *Elsa Lab, Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan*

[2] *MediaTek Inc., Hsinchu, Taiwan*

Abstract

Many existing conditional score-based data generation methods utilize Bayes' theorem to decompose the gradients of a log posterior density into a mixture of scores. These methods facilitate the training procedure of conditional score models, as a mixture of scores can be separately estimated using a score model and a classifier. However, our analysis indicates that the training objectives for the classifier in these methods may lead to a ***serious score mismatch issue***, which corresponds to the situation that the estimated scores deviate from the true ones. To resolve it, we theoretically formulate a novel training objective, called ***Denoising Likelihood Score Matching (DLSM)*** loss, for the classifier to match the gradients of the true log likelihood density. Our experimental evidences show that the proposed method outperforms the previous methods on benchmarks noticeably in terms of several key evaluation metrics.

Background

Langevin Diffusion

Langevin diffusion [1,2] can be used to ***generate data samples*** from an unknown data distribution $p(\tilde{\boldsymbol{x}})$ using only the score function $\nabla_{\tilde{\boldsymbol{x}}} \log p(\tilde{\boldsymbol{x}})$, where $\tilde{\boldsymbol{x}} \in \mathbb{R}^d$ represents a data sample:

$$\tilde{\boldsymbol{x}}_t = \tilde{\boldsymbol{x}}_{t-1} + \frac{\epsilon^2}{2} \nabla_{\tilde{\boldsymbol{x}}} \log p(\tilde{\boldsymbol{x}}_{t-1}) + \epsilon \boldsymbol{z}_t, \tag{1}$$

where $\tilde{\boldsymbol{x}}_0$ is sampled from an arbitrary distribution, ϵ is a fixed positive step size, and $\boldsymbol{z}_t$ is a noise vector sampled from a normal distribution $N(\boldsymbol{0}, \boldsymbol{I}_{d\times d})$ for simulating a d-dimensional standard Brownian motion.

Denoising Score Matching

To train a score model $\mathrm{s}(\tilde{\boldsymbol{x}}; \phi)$, a feasible approach is to minimize the Explicit Score-Matching (ESM) loss L_{ESM} [3], represented as:

$$L_{\mathrm{ESM}} = \mathbb{E}_{p_\sigma(\tilde{\boldsymbol{x}})} \left[\frac{1}{2} \| \mathrm{s}(\tilde{\boldsymbol{x}}; \phi) - \nabla_{\tilde{\boldsymbol{x}}} \log p_\sigma(\tilde{\boldsymbol{x}}) \|^2 \right]. \tag{2}$$

Based on Parzen density estimation, an efficient alternative, called ***Denoising Score-Matching (DSM)*** loss [3], is proposed to efficiently calculate the equivalent loss L_{DSM}, expressed as:

$$L_{\mathrm{DSM}} = \mathbb{E}_{p_\sigma(\tilde{\boldsymbol{x}}, \boldsymbol{x})} \left[\frac{1}{2} \| \mathrm{s}(\tilde{\boldsymbol{x}}; \phi) - \nabla_{\tilde{\boldsymbol{x}}} \log p_\sigma(\tilde{\boldsymbol{x}}|\boldsymbol{x}) \|^2 \right], \tag{3}$$

where $\nabla_{\tilde{\boldsymbol{x}}} \log p_\sigma(\tilde{\boldsymbol{x}}|\boldsymbol{x})$ is simply $\frac{1}{\sigma^2}(\boldsymbol{x} - \tilde{\boldsymbol{x}})$.

Parzen Density Estimation

Previous literature [3] utilized Parzen density estimation to replace the ***Dirac function*** $p(\boldsymbol{x}) = \frac{1}{m}\sum_{i=1}^m \delta(\| \boldsymbol{x} - \boldsymbol{x}^{(i)} \|)$ with ***isotropic Gaussian smoothing kernels*** $p_\sigma(\tilde{\boldsymbol{x}}|\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2}\sigma^d} e^{\frac{-1}{2\sigma^2}\|\tilde{\boldsymbol{x}} - \boldsymbol{x}\|^2}$ with variance σ^2, where $\boldsymbol{x} \in \mathbb{R}^d$ represents a data sample, and $\{\boldsymbol{x}^{(i)}\}_{i=1}^m$ are identically distributed data points. Specifically, Parzen density estimation enables the calculation of $p(\tilde{\boldsymbol{x}}) = \frac{1}{m}\sum_{i=1}^m p_\sigma(\tilde{\boldsymbol{x}}|\boldsymbol{x}^{(i)})$.

Conditional Score Decomposition

Score models can be extended to conditional models when conditioned on some extra information $\tilde{\boldsymbol{y}}$. A popular approach adopted by researchers utilizes ***Bayes' theorem*** to decompose the conditional score $\nabla_{\tilde{\boldsymbol{x}}} \log p_{\sigma,\tau}(\tilde{\boldsymbol{x}}|\tilde{\boldsymbol{y}})$ as a ***mixture of scores***, i.e.,

$$\nabla_{\tilde{\boldsymbol{x}}} \log p_{\sigma,\tau}(\tilde{\boldsymbol{x}}|\tilde{\boldsymbol{y}}; \theta, \phi) = \nabla_{\tilde{\boldsymbol{x}}} \log p(\tilde{\boldsymbol{y}}|\tilde{\boldsymbol{x}}; \boldsymbol{\theta}) + \mathrm{s}(\tilde{\boldsymbol{x}}; \phi), \tag{4}$$

where $p(\tilde{\boldsymbol{y}}|\tilde{\boldsymbol{x}}; \boldsymbol{\theta})$ is trained with $L_{\mathrm{CE}} = H(p_{\sigma,\tau}(\tilde{\boldsymbol{y}}|\tilde{\boldsymbol{x}}),\ p(\tilde{\boldsymbol{y}}|\tilde{\boldsymbol{x}}; \boldsymbol{\theta}))$, and $\mathrm{s}(\tilde{\boldsymbol{x}}; \phi)$ is trained with L_{DSM}.

Motivational Example

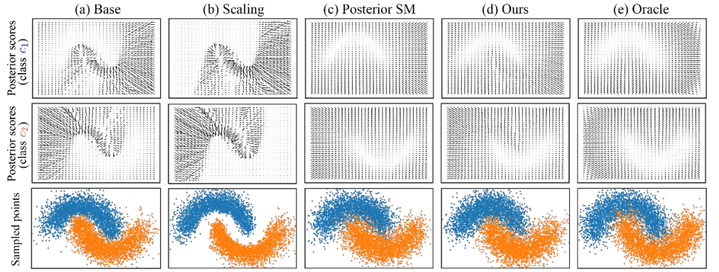

Figure 1. The visualized results on the *inter-twining moon* dataset.

(a) Base method: $\nabla_{\tilde{\boldsymbol{x}}} \log p(\tilde{\boldsymbol{x}} \mid \tilde{\boldsymbol{y}}\,; \theta, \phi) = \nabla_{\tilde{\boldsymbol{x}}} \log p(\tilde{\boldsymbol{y}} \mid \tilde{\boldsymbol{x}}\,; \theta) + \mathrm{s}(\tilde{\boldsymbol{x}}; \phi)$, described in Eq. (4).
(b) Scaling method: $\nabla_{\tilde{\boldsymbol{x}}} \log p(\tilde{\boldsymbol{x}} \mid \tilde{\boldsymbol{y}}\,; \theta, \phi) = \alpha \nabla_{\tilde{\boldsymbol{x}}} \log p(\tilde{\boldsymbol{y}} \mid \tilde{\boldsymbol{x}}\,; \theta) + \mathrm{s}(\tilde{\boldsymbol{x}}; \phi)$, where $\alpha > 0$.
(c) Posterior SM method: The posterior scores for different class conditions, in this case, c_1 and c_2, are separately estimated using different score models $\mathrm{s}(\tilde{\boldsymbol{x}}; \phi_1)$ and $\mathrm{s}(\tilde{\boldsymbol{x}}; \phi_2)$ trained with L_{DSM}.
(d) Ours: The posterior scores are estimated in a similar fashion as method (a) except that the classifier is trained with the proposed loss function.
(e) Oracle: The oracle posterior scores are directly computed.

Methodology

Denoising Likelihood Score Matching

A score model trained with the score-matching objective can potentially be beneficial in producing a better posterior score estimation. In light of this, a classifier may be enhanced if the score-matching process is involved during its training procedure. An intuitive way to accomplish this aim is through minimizing the explicit likelihood score-matching loss L_{ELSM}, which is defined as the following:

$$L_{\mathrm{ELSM}} = \mathbb{E}_{p_{\sigma,\tau}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}})} \left[\frac{1}{2} \| \nabla_{\tilde{\boldsymbol{x}}} \log p(\tilde{\boldsymbol{y}}|\tilde{\boldsymbol{x}}; \theta) - \nabla_{\tilde{\boldsymbol{x}}} \log p_{\sigma,\tau}(\tilde{\boldsymbol{y}}|\tilde{\boldsymbol{x}}) \|^2 \right]. \tag{5}$$

This loss term, however, requires the derivation of the true likelihood score $\nabla_{\tilde{\boldsymbol{x}}} \log p_{\sigma,\tau}(\tilde{\boldsymbol{y}}|\tilde{\boldsymbol{x}})$, which is computationally unaffordable during the classifier training. To reduce the computational cost, we follow the derivation of DSM as well as Bayes' theorem, and formulate an alternative objective L_{DLSM}:

$$L_{\mathrm{DLSM}} = \mathbb{E}_{p_{\sigma,\tau}(\tilde{\boldsymbol{x}}, \boldsymbol{x}, \tilde{\boldsymbol{y}}, \boldsymbol{y})} \left[\frac{1}{2} \| \nabla_{\tilde{\boldsymbol{x}}} \log p(\tilde{\boldsymbol{y}}|\tilde{\boldsymbol{x}}; \theta) + \nabla_{\tilde{\boldsymbol{x}}} \log p_\sigma(\tilde{\boldsymbol{x}}) - \nabla_{\tilde{\boldsymbol{x}}} \log p_\sigma(\tilde{\boldsymbol{x}}|\boldsymbol{x}) \|^2 \right]. \tag{6}$$

In contrast to L_{ELSM}, L_{DLSM} is more ***computationally feasible*** since $\nabla_{\tilde{\boldsymbol{x}}} \log p_\sigma(\tilde{\boldsymbol{x}})$ can be estimated using a score model $\mathrm{s}(\tilde{\boldsymbol{x}}; \phi)$ trained with L_{DSM}, and $\nabla_{\tilde{\boldsymbol{x}}} \log p_\sigma(\tilde{\boldsymbol{x}}|\boldsymbol{x})$ equals to $\frac{1}{\sigma^2}(\boldsymbol{x} - \tilde{\boldsymbol{x}})$.

The Introduction of the DLSM Loss in the Classifier Training Process

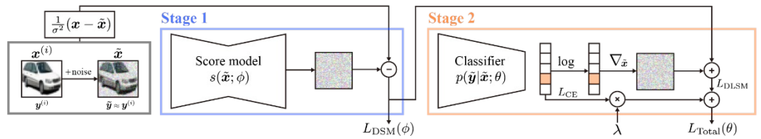

Figure 2. The training procedure of the proposed methodology.

In practice, the total training objective of the classifier is written as follows:

$$L_{\mathrm{Total}} = L_{\mathrm{DLSM}} + \lambda\, L_{\mathrm{CE}}, \tag{7}$$

where $\lambda > 0$ is a balancing coefficient. Fig. 2 depicts a ***two-stage training procedure*** adopted in this work. In stage 1, a score model $\mathrm{s}(\tilde{\boldsymbol{x}}; \phi)$ is updated using L_{DSM} to match $\nabla_{\tilde{\boldsymbol{x}}} \log p_\sigma(\tilde{\boldsymbol{x}})$. In stage 2, the weights of the trained score model are fixed, and a classifier $\nabla_{\tilde{\boldsymbol{x}}} \log p(\tilde{\boldsymbol{y}}|\tilde{\boldsymbol{x}}; \theta)$ is updated using L_{DLSM}. After these two training stages, $\mathrm{s}(\tilde{\boldsymbol{x}}; \phi)$ and $\nabla_{\tilde{\boldsymbol{x}}} \log p(\tilde{\boldsymbol{y}}|\tilde{\boldsymbol{x}}; \theta)$ can then be added together based on Eq. (4) to perform conditional sampling.

Experimental Results

		Cifar-10			Cifar-100		
		Base	Scaling	Ours	Base	Scaling	Ours
FID	↓	4.10	12.48	**2.25**	4.52	12.58	**3.86**
IS	↑	9.08	9.37	**9.90**	11.53	11.59	**11.62**
Precision	↑	0.67	**0.75**	0.65	0.62	**0.65**	0.61
Recall	↑	0.61	0.49	**0.62**	0.62	0.52	**0.63**
Density	↑	1.05	**1.36**	0.96	0.84	**0.93**	0.82
Coverage	↑	0.80	0.75	**0.81**	**0.71**	0.61	**0.71**
CAS	↑	0.38	0.46	**0.58**	0.15	0.24	**0.28**
(CW) Precision	↑	0.51	**0.70**	0.56	0.33	**0.59**	0.42
(CW) Recall	↑	0.59	0.42	**0.61**	**0.68**	0.41	**0.68**
(CW) Density	↑	0.63	**1.23**	0.76	0.30	**0.78**	0.43
(CW) Coverage	↑	0.60	0.66	**0.71**	0.38	0.51	**0.52**

Table 1. The evaluation results on the Cifar-10 and Cifar-100 datasets. The P / R / D / C metrics with `(CW)' in the last four rows represents the average class-wise P / R / D / C metrics . The arrow symbols ↑ /↓ represent that a higher / lower evaluation result correspond to a better performance.

- Our method outperforms the other two methods with substantial margins in terms of ***Frechet Inception Distance (FID)*** [5] and ***Inception Score (IS)*** [6], indicating that the generated samples bear closer resemblance to the real data.
- The evaluation results on the ***classification accuracy score (CAS)*** [7] and the ***class-wise (CW) Precision, Recall, Density, and Coverage (P / R / D / C)*** [8] ***metrics*** suggest that our method offers a superior ability for a classifier to learn accurate class information as compared to the base method.

[1] G. O Roberts and J. S Rosenthal. Optimal scaling of discrete approximations to langevin diffusions. Journal of the Royal Statistical Society. 1998.
[2] G. O Roberts and R. L. Tweedie. Exponential convergence of Langevin distributions and their discrete approximations. Bernoulli. 1996.
[3] P. Vincent. A connection between score matching and denoising autoencoders. Neural Computation. 2011.
[4] P. Dhariwal and A. Nichol. Diffusion models beat GANs on image synthesis. arXiv:2105.05233. 2021.
[5] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. NeurIPS. 2017.
[6] S. Barratt and R. Sharma. A note on the inception score. ICML Workshop. 2018.
[7] S. Ravuri and O. Vinyals. Classification accuracy score for conditional generative models. NeurIPS. 2019.
[8] M. F. Naeem, S. J. Oh, Y. Uh, Y. Choi, and J. Yoo. Reliable fidelity and diversity metrics for generative models. ICML. 2020.

Acknowledgement

Questions?

lance_chao@gapp.nthu.edu.tw