Score-Based Generative Modeling

Paul O'Mahony
ENSAE

Pierre-Emmanuel Diot
ENSAE

Joé Vincent-Galtié
ENSAE

Abstract

The recent focus on generative models is the result of the abundant and particularly successful research on generating new data points in a specific domain, including image or text generation. In particular, VAEs and GANs have shown very good results. More recently, score-based models have attracted a lot of interest because of the wide range of tasks they cover. The article [1] presents these methods which consist in learning the latent structure of a dataset by modeling the way data points diffuse in the latent space in a score functions perspective.

This report first presents research results in this area. In particular, by score-based generative modeling through stochastic differential equations, researchers have succeeded in proposing a framework allowing : (1) to model in a flexible way, (2) to reach state of the art results in particular in image generation while (3) allowing an accurate evaluation of probabilities.

Secondly , this report presents the results of two successive approaches that complement each other. First [1], a naive scoremodeling and Langevin dynamics approach has been applied to tabular data in a data augmentation perspective. In this case, it did not turn out to outperform GANs or VAEs. It was then decided to approach the results [2] to the state of the art in image generation using SDE. We then obtained satisfactory results allowing to explore the impact of the different parameters although inferior to those presented in the paper. An additional optimization on the backbone model would have been necessary.

[1] Link to the code of first approach
[2] Link to the code of second approach

1 Theoretical presentation of score-based models

Let $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$ be a large dataset that can be used to build a deep generative model. These data points can be seen as following un underlying data distribution $p(\mathbf{x})$. As this distribution isn't known in its analytical form, one has to estimate it. Doing so, by creating and tuning a model, i.e a parametrized probability distribution, then enables to create new datapoints by sampling as well as evaluating probability for any potential datapoint.

1.1 Training generative models

Let $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \sim p_{data}$. The purpose of training a generative model is to find $\theta \in \Theta$ such that $p_{data} \approx p_\theta$.

As high dimensional data has very complex data distribution, deep neural network with one-dimensional output $f_\theta(\mathbf{x}) \in \mathbb{R}$, have to be used to represent data distribution.

$$p_\theta(\mathbf{x}) = \frac{e^{-f_\theta(\mathbf{x})}}{Z_\theta}, \tag{1}$$

where $Z_\theta > 0$ is a normalizing constant such that $\int p_\theta(\mathbf{x})\mathrm{d}\mathbf{x} = 1$. One main issue is that, except for gaussians model and other simple models, **such a normalizing constant can't be computed** (p-complete issue).

Therefore some approaches try to tackle this issue :

- by approximating this normalizing constant
- by restricting the models to compute analytically such a constant
- by using GANs in order to model the generation process

However, all these methods have flaws : the first one approximate the normalizing constant and therefore prevents a good probability evaluation, the second one is too restricted so that the generation of samples isn't good enough, while the third one is good for generation but can't be used for probability evaluation.

This background of generative models explains the introduction of score-based models.

1.2 Introduction of score functions

For a probability density function $p(\mathbf{x})$, the Stein score function is defined as

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}) \tag{2}$$

. While keeping the same level of information as the density function, such a function is easier to compute.

Working with such function will enable in the following framework :

- the use of flexible models
- improved generation
- good probability evaluation

1.3 Use more flexible models by bypassing the normalizing constant

Working with score functions can bypass the normalizing constant issue because :

$$\begin{aligned}\nabla_{\mathbf{x}} \log p_\theta(\mathbf{x}) &= \nabla_{\mathbf{x}} \log \frac{e^{-f_\theta(\mathbf{x})}}{Z_\theta} \\ &= -\nabla_{\mathbf{x}} f_\theta(\mathbf{x}) - \underbrace{\nabla_{\mathbf{x}} \log Z_\theta}_{=0} \\ &= -\nabla \mathbf{x} f_\theta(\mathbf{x}).\end{aligned}$$

Automatic differentiation enables efficient computation of such function. Let's then denote $\mathbf{s}_\theta(\mathbf{x})$ this score $-\nabla \mathbf{x} f_\theta(\mathbf{x})$.

Then, given $\mathbf{s}_\theta(\mathbf{x})$, Fischer divergence enables to compare it to $\nabla_{\mathbf{x}} \log p_{data}(\mathbf{x})$.

If Fischer divergence, defined by

$$\mathbb{E}_{p_{data}(\mathbf{x})}[\|\nabla_{\mathbf{x}} \log p_{data}(\mathbf{x}) - \mathbf{s}_\theta(\mathbf{x})\|_2^2] \tag{3}$$

, can't be directly computed because the ground truth value of data distribution isn't known, a way to address this issue is to apply a score matching method [2] such that $\mathbb{E}_{p_{data}(\mathbf{x})}[\frac{1}{2}\|\mathbf{s}_\theta(\mathbf{x})\|_2^2 + trace(\nabla_{\mathbf{x}} \mathbf{s}_\theta(\mathbf{x}))] = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{p_{data}(\mathbf{x})}[\frac{1}{2}\|\mathbf{s}_\theta(\mathbf{x})\|_2^2 + trace(\nabla_{\mathbf{x}} \mathbf{s}_\theta(\mathbf{x}))]$

However, as $\mathbf{s}_\theta(\mathbf{x})$ is computationally too heavy (e.g proportional to the dimensions of the input space), sliced score matching ans denoising score matching[3] were introduced to further bypass this issue [4].

1.4 Improve generation with Langevin Dynamics

Once such a score function has been trained, it can be used through a Langevin dynamics procedures to draw samples.

Such a dynamics works as follow :

- initialize $\mathbf{x}_0 \sim \pi(\mathbf{x})$
- $\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + \epsilon \nabla_{\mathbf{x}} \log p(\mathbf{x}) + \sqrt{2\epsilon}\, \mathbf{z}_i$ for $i = 0, 1, \cdots, K$
- which in our context gives, $\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + \epsilon \mathbf{s}_\theta(\mathbf{x}) + \sqrt{2\epsilon}\, \mathbf{z}_i$ for $i = 0, 1, \cdots, K$

Then, when $\epsilon \to 0$ and $K \to \infty$, $x_K \sim p_{data}(\mathbf{x})$

Nevertheless, to apply just a Langevin Dynamics has shown bad results due to bad estimation of score functions in low density regions. The lack of data points in such regions prevents any satisfying computation of the Fischer divergence approximation defined above.

To tackle this issue, [5] introduced perturbations of data points by injecting gaussian noise in order to populate low density regions. In particular, not to corrupt too much the original distribution, multiple increasing noise levels are used to perturb the data and a Noise Conditions Score Based Model $\mathbf{s}_\theta(\mathbf{x}, i)$ is trained.

Therefore to get $\mathbf{s}_\theta(\mathbf{x}, i) \approx \nabla_{\mathbf{x}} \log p_{\sigma_i}(\mathbf{x})$ for $i \in [1, L]$ where L is the number of noise levels, the Noise Conditional Score Model is trained by minimizing the following objective function :

$$\sum_{i=1}^{L} \lambda(i) \mathbb{E}_{p_{\sigma_i}(\mathbf{x})}[\|\nabla_{\mathbf{x}} \log p_{\sigma_i}(\mathbf{x}) - \mathbf{s}_\theta(\mathbf{x}, i)\|_2^2] \tag{4}$$

where $\lambda(i) \in \mathbb{R}_{>0}$ is a positive weighting function.

This framework, a mixture of score matching loss function, is then a generalization to the training objective of diffusion probabilistic models [6]. Such a connection between score-based and diffusion models being unveiled by [7].

To generate new data points, annealed Langevin dynamics can be applied to sequentially sample from score models from highest to lowest noise levels.

If this method gives good results, even outperfoming GANs in some metrics, it is especially interesting as it requires much less computational power.

Furthermore, score-based models offer a **controllable generation process** thanks to the inverse distribution.

Following Bayes' rule, we can apply it for score functions so that:

$$\begin{aligned}\nabla_x logp(x|y) &= \nabla_x logp(x) + \nabla_x logp(y|x) \\ &\quad - \nabla_x logp(y) \\ &= \nabla_x logp(x) + \nabla_x logp(y|x)\end{aligned}$$

$\nabla_x logp(x)$ is then the unconditional score model while $\nabla_x logp(y|x)$ is the gradient of the log forward model, in most cases a classifier that can be easily computed. Therefore, the unconditionnal model has to be trained only once and only the forward model has to be trained. Moreover, the forward model can be directly specified is the task is in the same domain as the one concerned by the first training.

1.5 Beyond annealed Langevin Dynamics : outperforming probability evaluation with SDE, reverse SDE and corresponding ODE

The paper [8] generalized the framework defined above by using infinine levels of noise instead of finite ones. They introduced therefore a continuous-time stochastic process.

Let $\{\mathbf{x}\}_{t\in[0,T]}$ be a stochastic process, i.e. collection of an infinite number of random variables.

$\forall t \in [0,T], \mathbf{x}_t$ is associated to a corresponding probability densitiy $p_t(\mathbf{x})$.

To be so, the stochastic process has to be solution to the following Stochastic differential equation.

$$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x},t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$$

where $\mathbf{f}(;t) : \mathbb{R}^d \to \mathbb{R}^d$ is a deterministic drift and $\mathrm{d}\mathbf{w}$ is an infinitesimal gaussian noise.

Thus, $p_0(\mathbf{x}) \approx p_{data}(\mathbf{x})$ while $p_T(\mathbf{x}) \approx \pi(\mathbf{x})$ where π is a gaussian distribution.

Analogously to the previous framework, such perturbation process can be reversed through the use of the corresponding SDE

$$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x},t) - g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x})]\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}$$

where $\bar{\mathbf{w}}$ is a the infinitesimal noise in the reverse time direction.

The key is then to estimate the score function $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ by parametrizing $\mathbf{s}_\theta(\mathbf{x})$ a Time-Dependant Score based Model.

The training procedure depends on minimizing

$$\mathbb{E}_{t\in\mathcal{U}(0,T)}\mathbb{E}_{p_t(\mathbf{x})}[\lambda(t)\|\nabla_{\mathbf{x}} \log p_t(\mathbf{x}) - \mathbf{s}_\theta(\mathbf{x},t)\|_2^2] \tag{5}$$

where λ is a positive weighting function.

Once trained, the estimated reverse SDE is given by

$$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x},t) - g^2(t)\mathbf{s}_\theta(\mathbf{x},t)]\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$$

.

It can then be solved by, among others, the Euler-Maruyama method such that :

$$\begin{aligned}\mathbf{x} &\leftarrow x - \sigma(t)^2\mathbf{s}_\theta(\mathbf{x},t)]\Delta t + \sigma(t)\mathbf{z}_t \\ t &\leftarrow t + \Delta t\end{aligned}$$

where σ is the continuous time generalization of noise table σ_i such that in a simplified version of the SDE model, $\mathrm{d}\mathbf{x}_t = \sigma(t)\mathrm{d}\mathbf{w}_t$.

To go further and compute the exact log-likelihood of score-based generative models, [8] showed it is SDE can be converted into Ordinary differantial equation such that

$$\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x},t) - \frac{1}{2}g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right]\mathrm{d}t. \tag{6}$$

. Put another way, the stochastic process can be also seen as a corresponding deterministic process whose advantage is that it enables exact likelihood computation.

1.5.1 Score-based modeling summary

Therefore, [8] introduced methods that make score-based generative modeling performant and even outperforming GANs or VAEs by enabling:

- the use of flexible models due to bypass of the normalizing constant issue
- improved and controllable generation
- accurate probability evaluation

2 Experiments framework

Thus, score-based modeling has reached a state-of-the-art performance on several domains, from image generation to time series prediction. The remainder of this report consists in an objectivation of the results according to two successive approaches.

The first one is the **application of the naive approach and the Langevin dynamics**, methods that have proven very satisfactory results, especially on image generation, to the sampling of **tabular data** in a data augmentation perspective. The second one aims at **approaching the state of the art of score-based modeling on a proven domain. By applying different SDE to image generation, we compare these results to those of the initial paper**.

2.1 Score-based generative modeling on tabular data

A large panel of studies have focused on generating synthetic images using score-based models. In this experiment, emphasis is placed on using score-matching methods and Langevin sampling to generate synthetic tabular data. When dealing with tabular data, the imbalanced class issue can make it difficult to build a scorer which correctly predicts the minority class. One solution to this problem is to synthetically generate data to increase the number of entries in the minority class.

Method. The approach we adopt to perform data augmentation is as follows:

1. select samples from the target class to perform data augmentation on.
2. learn score functions using **gradients of log probability density functions** on a large number of noise-perturbed data distributions.
3. generate samples with Langevin-type sampling using the trained score functions.
4. fit classifiers using both the original training set and newly created data points as inputs.

Model. We use the MLP architecture with Layer Normalization and ELU activation function to approximate the score function $s_\theta(\mathrm{x})$. This choice is motivated by the hypothesis that convolution and pooling layers are less necessary to embed tabular data than images.

Dataset. The selected dataset is the Default Payments of Credit Card Clients [9] which contains information on 30,000 default payments, demographic factors, credit data, history of payment, and bill statements of credit card clients in Taiwan from April 2005 to September 2005. This dataset is usually used as input in classification models in order to build payment default scoring models. Entries which have defaulted on their credit only account for 22% of the entire dataset. We thus focus the generative process on these samples.

Loss functions. Score matching is used to learn the score-based generative model $s_\theta(\mathrm{x})$. This technique minimizes the Fisher divergence without knowledge of the ground-truth data score. Two objective functions are used:

1. denoising score matching objective (`dsm`)
2. sliced score matching objective (`ssm`) to enhance the computation

To overcome the difficulty of accurate score estimation in regions of low data density, gaussian pertubations are applied on input data points. The training objective is then the one defined in 4.

Sampling methods. Once the score functions learned with score matching, we apply Langevin dynamics and annealed Langevin dynamics to generate new tabular data points, as explained in 1.4. Note the dimension of each synthetic entry is the same as each training sample's.

Data quality assessment. It is important to measure the quality of synthetic generated samples using a reliable methodology. For this purpose, we leverage the scoring methods provided by the `SDV` [10] python library. The

quality score is the average between two general scores which are Column Pair Trends and Column Shapes. The former describes how two columns vary in relation to each other while the latter computes the similarity of a real column *wrt* a synthetic column in terms of the marginal distributions. More details on these scores can be found in subsection A.a in the appendix. We also evaluate the change in classification metrics when training classifiers on the augmented dataset. The selected metric is F_1 score as it symmetrically incorporates both precision and recall in one metric.

Method comparison. We compare the score-based approach to other generative methods such as Gaussian Copula, GAN and VAE which are also available in the `SDV` package.

2.2 Score-based generative modeling with stochastic differential equations (SDEs)

Model. We the use the U-Net [11] architecture as the backbone of the score network $s_\theta(\mathbf{c}, t)$. It has been indicated by the authors of [1] that this architecture performs well in many cases and, being limited in our computational resources, we use only this one. We train the model with the objective of minimizing (5).

Exponential Moving Average (EMA). EMA computes the weighted mean of all the previous data points and the weights decay exponentially. For a given sequence of data points $z_1, z_2, \ldots, z_t, \ldots$, the EMA at data point t is calculated by:

$$\mathrm{EMA}_t = \begin{cases} z_1 & \text{if } t = 1 \\ \alpha z_t + (1-\alpha)\mathrm{EMA}_{t-1} & \text{else} \end{cases} \tag{7}$$

In many cases, using the EMA of the parameters over all training iterations improves the model performances. In our experiments we train our model with and without EMA to compare the performances.

Dataset. We use the FashionMNIST [12] dataset which contains 28x28 grayscale images of 70,000 fashion products from 10 categories, with 7,000 images per category.

SDEs. We use two different SDE that perturbs the data distribution p_0 to a prior distribution p_T:

$$d\mathbf{x} = \sigma^t d\mathbf{w}. \tag{8}$$

and,

$$\mathrm{d}\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}\mathrm{d}t + \sqrt{\beta(t)\left(1 - e^{-2\int_0^t \beta(s)\mathrm{d}s}\right)}\mathrm{d}\mathbf{w} \tag{9}$$

We incorporate the time information via Gaussian random features which is defined by:

$$[\sin(2\pi\omega t); \cos(2\pi\omega t)], \tag{10}$$

with $\omega \sim \mathcal{N}\left(0, s^2\mathbf{I}\right)$.
We denote the first one `simpleSDE` and the second one is denoted by `subVPSDE` in [1].

Samplers. We use three different sampling algorithms to generate new samples:

1. Probability flow ordinary differential equation based sampler (ODE) [1]
2. Euler-Maruyama sampler [13]
3. Predictor-Corrector sampler [13]

Sample quality evaluation metric. To evaluate the quality of the samples generated by our models, we use the Fréchet Inception Distance [14] (FID). The FID is based on the inception score which estimates the quality of a collection of synthetic images based on how well the top-performing image classification model Inception v3 classifies them as one of 1000 known objects. The FID is defined by:

$$\mathrm{FID} = |\mu_1 - \mu_2| + Tr(\sigma_1 + \sigma_2 - 2\sqrt{\sigma_1 * \sigma_2}) \tag{11}$$

where μ_1 and μ_2 are the feature-wise mean of the real and generated images. σ_1 and σ_2 are the covariance matrix for the real and generated feature vectors.

3 Results

This section aims at highlighting the empirical results obtained.

3.1 Score-based generative modeling on tabular data

We first report the data quality results obtained for different modeling strategies.The `AnnealScoreNet` $s_\theta(\mathrm{x}, i)$ model is an improvement of `ScoreNet` $s_\theta(\mathrm{x})$ as it is trained on noise-pertubated data to improve score estimation in regions of low data density.

Model	Loss	Overall Quality Score (%)
ScoreNet	dsm	52.35
	ssm	52.12
AnnealScoreNet	dsm	**55.44**
GaussianCopula		84.66
GAN		88.11
VAE		89.51

Table 1: Data quality for different generative modeling strategies.

More details on Column Shapes & Column Pair Trend scores can be found in table 4. One can also find distribution plots for numeric and categorical variables in figures 1 and 2. This figures somehow illustrate the limited data quality obtained with our score-based approach. It can indeed be noted important distribution differences between real and synthetic data.

Then, we compare the test F_1 score obtained with LogisticRegression fitted on different training sets: the original dataset and the augmented datasets with score-based and other generative models. Note there are more Default entries in the augmented training sets since we added 5,000 synthetic Default data points in each of them.

Model	Loss	Default	No Default
Original dataset		0.357	0.889
ScoreNet	dsm	0.363	0.89
	ssm	0.362	0.889
AnnealScoreNet	dsm	**0.384**	**0.88**
GaussianCopula		0.372	0.873
GAN		0.465	0.881
VAE		0.453	0.87

Table 2: F_1 score on test data with LogisticRegression trained on original training set vs. augmented training sets from different generative modeling strategies. Both the GAN and VAE strategies outperform our approach when comparing the scores on the Default class. However, adding synthetic samples created from score-based models lead to an increase in the F_1 score for the Default class, while keeping constant the No Default F_1 score. Using the AnnealScoreNet framework with dsm training objective give us the most encouraging results.

3.2 Score-based generative modeling through SDE

To tune the hyperparameters while training our model we use the framework wandb[3]. The framework randomly chooses values for the hyperparameters learning rate, batch size, σ (for simpleSDE) and the number of epochs. In the appendix with the figures 3, 4 and 5, we can observe on the graphs that no matter the hyperparameters our model converges quite quickly. Moreover a large value for sigma and a small learning rate are preferable. The value of β in subVPSDE does not affect the convergence. Finally we deduce by the final value of the loss function that we have a better learning with the perturbation of the data realized with simpleSDE .

After several experiments, here are the results we get Table 3.

Sampler	SDE	EMA	FID Score
PC	Simple	EMA	63.55
		Sans EMA	80.67
	SUBVPSDE	EMA	**36.89**
		Sans EMA	45.78
EULER	Simple	EMA	71.12
		Sans EMA	74.73
	SUBVPSDE	EMA	90.16
		Sans EMA	83.72
ODE	Simple	EMA	137.4
		Sans EMA	137.81
	SUBVPSDE	EMA	284.17
		Sans EMA	226.18

Table 3: GM SDE Results

We observe that the SDE solver that gives us the best results is the PC. Despite the fact that ODE can be very useful because it allows an exact calculation of likelihood, the samples generated are of much lower quality than those generated by the other solvers. Finally we notice that training with EMA does not significantly increase the quality of the generated samples. In the appendix you will find overviews of the generated samples with the best (Figure 6) and worst quality (Figure 7).

4 Conclusion / Areas of improvement

In this study, we performed several experiments using several generative modeling methods and on different types of datasets.

Generating tabular data using score-based generative models is a challenging task. Our approach has shown mitigated results as synthetic data qual-

[3]For more information, visit wandb.ai.

ity is low compared to other generative models like GANs and VAE. Nonetheless, we managed to improve the classifier's ability to detect defaulted payments with specific configurations. We are convinced that our results could be enhanced using more complex neural network architecture which could help to generate samples closer to the original data. Another area for improvement would be to optimise the hyperparameters that intervene in the data generation process. These two promising approaches could be implemented with the use of more computational power.

In the case of the generation through SDEs we had the opportunity to test two SDEs and three different samplers on a dataset of images. To estimate the score function we used and trained the U-Net [11] achitecture with or without integrating the EMA. Our results are consistent and satisfactory. However, as an axis of improvement we can imagine testing several other backbone architectures and further optimize their hyperparameters. The use of other SDEs to perturb the data may also be interesting.

References

[1] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[2] Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. *J. Mach. Learn. Res.*, 6:695–709, 2005.

[3] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011.

[4] Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced score matching: A scalable approach to density and score estimation, 2019.

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

[6] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics, 2015.

[7] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020.

[8] Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models, 2021.

[9] I-Cheng Yeh. default of credit card clients. UCI Machine Learning Repository, 2016. DOI: 10.24432/C55S3H.

[10] Neha Patki, Roy Wedge, and Kalyan Veeramachaneni. The synthetic data vault. In *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 399–410, Oct 2016.

[11] Olaf Ronneberger, Philipp Fischer, Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *arXiv preprint arXiv:1505.04597*, 2015.

[12] Han Xiao, Kashif Rasul, Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[13] Yang Song. Generative Modeling by Estimating Gradients of the Data Distribution. https://yang-song.net/blog/2021/score/, 2021.

[14] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *arXiv preprint arXiv:1706.08500*, 2017.

Appendix

A Experiments framework

A.a Score-based generative modeling on tabular data

The quality score is the average between two general scores which are Column Pair Trends and Column Shapes.

Column Pair Trends

It is the average between `ContingencySimilarity` and `CorrelationSimilarity` scores.

For categorical variables, the score is `ContingencySimilarity` which computes the similarity of a pair of categorical columns between the real and synthetic datasets. For a pair of columns, A and B, the test computes a normalized contingency table for the real and synthetic data. This table describes the proportion of rows that have each combination of categories in A and B. Then, it computes the difference between the contingency tables using the Total Variation Distance. Finally, we subtract the distance from 1 to ensure that a high score means high similarity.

$$s_{con} = 1 - \frac{1}{2} \sum_{\alpha \in A} \sum_{\beta \in B} |S_{\alpha,\beta} - R_{\alpha,\beta}|$$

where α describes all the possible categories in column A and β describes all the possible categories in column B. R and S refer to the real and synthetic frequencies for those categories.

If the contingency table is exactly the same between the real vs. synthetic data, then $s_{cor} = 1$. The more different the contingency tables, the closer s_{con} to 0.

For numerical variables, the score is `CorrelationSimilarity` measures the correlation between a pair of numerical columns and computes the similarity between the real and synthetic data. This metric supports both the Pearson and Spearman's rank coefficients to measure correlation. For a pair of columns, A and B, this test computes a correlation coefficient on the real and synthetic data, R and S. This yields two separate correlation values. The test normalizes and returns a similarity score using the formula below.

$$s_{cor} = 1 - \frac{1}{2}|S_{A,B} - R_{A,B}|$$

The pairwise correlations of the real and synthetic data are exactly the same if $s_{cor} = 1$. The more different the pairwise correlations, the closer s_{cor} to 0.

Column Shapes

This score is computed as the average between the `KSComplement` and `TVComplement` scores which computes the similarity of a real column vs. a synthetic column in terms of the column shapes (marginal distributions).

The `KSComplement` score uses the Kolmogorov-Smirnov statistic to compare the distributions of the two continuous columns using the empirical CDF. It returns 1 minus the KS Test D statistic, which indicates the maximum distance between the expected CDF and the observed CDF values.

`TVComplement` is the Total Variation Distance (TVD) between the real and synthetic columns. The `TVComplement` score is $s_{TV} = 1 - \delta(R, S)$ where $\delta(R, S)$ is the TVD statistic.

$$\delta(R, S) = \frac{1}{2} \sum_{\omega \in \Omega} |R_\omega - S_\omega|$$

where ω describes all the possible categories in a column Ω.

B Results

B.a Score-based generative modeling on tabular data

Model	Loss	Column Pair Trends (%)	Column shapes
ScoreNet	dsm	65.07	39.62
	ssm	64.75	39.49
AnnealScoreNet	dsm	67.01	43.87
GaussianCopula		91.43	77.9
GAN		92.34	88.11
VAE		89.56	89.46

Table 4: Data quality for different generative modeling strategies. See subsection A.a for more details on Column Pair Trends and Column Shapes.

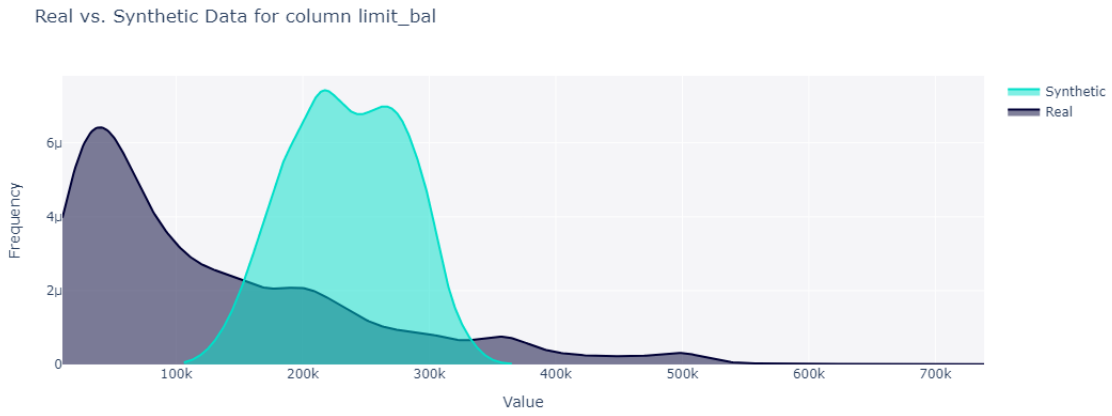

Figure 1: Distribution of amount of given credit for real (limit_bal) vs. synthetic data. The modeling process used is ScoreNet with dsm loss function. New data point are sampled with Langevin dynamics.

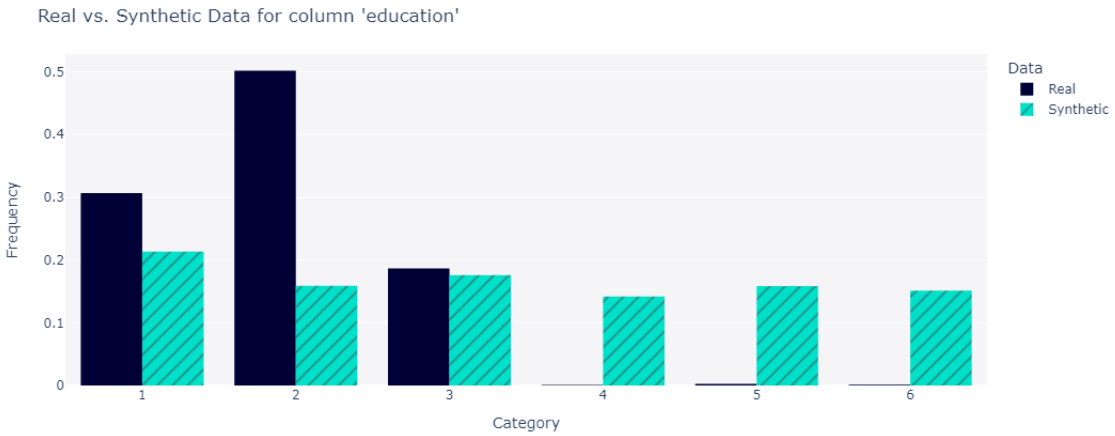

Figure 2: Repartition of education levels for real vs. synthetic data. The modeling process used is ScoreNet with dsm loss function. New data point are sampled with Langevin dynamics.

B.b Score-based generative modeling through SDEs

Figure 3: Train loss

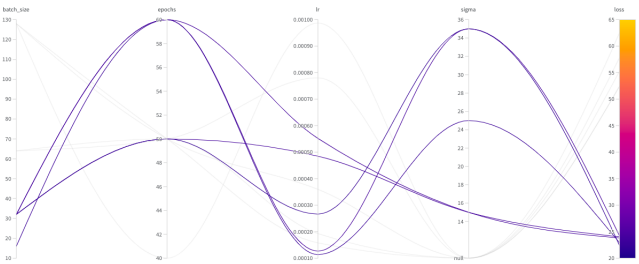

Figure 4: Hyperparameters tuning simpleSDE

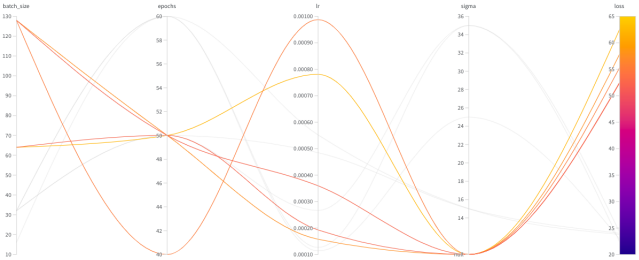

Figure 5: Hyperparameters tuning subVPSDE

Figure 6: Sample generated by our model with PC sampling trained with EMA

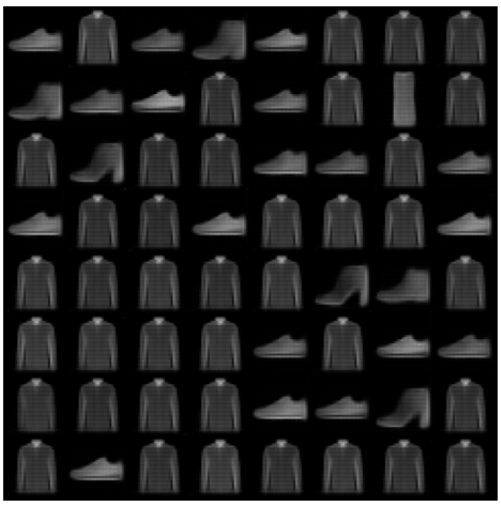

Figure 7: Sample generated by our model with PC sampling trained with EMA