Seminar Report
Denoising Diffusion Probabilistic Models

Jorge Augusto Calvimontes Robles (7024517)
Deep Generative Diffusion Models
Saarland University

1 Introduction

Generative models aim to produce data based on an unsupervised training of a model, which in most cases consists of a deep neural network. At the time of this paper, the most popular generative architectures consisted of Autoencoders [1], Variational Autoencoders [2] and Generative Adversial Networks [3]. In order to generate data, these models are trained such that it learns a transformation from an input sample noise to a desired distribution. The most popular use case is image generation, where the distribution correspond to a certain class type. In order to learn such transformation, these architectures use an encoder-decoder architecture. Which implies that the input data is transformed to a low-dimensional latent feature representation by the encoder, which then is used to generate a sample by the decoder. Implying that the encoder must learn to create a good latent and the decoder to use such latent to generate a correct distribution. The problem these models face is that by using a latent, there is not a tractable density function in which to optimize. A remedy for that is to use variational inference, the Evidence Lower Bound likelihood is commonly used to model the loss on such models.

Considering the ideas of [4] this paper presents a new generative approach based on diffusion. On the context of this paper, diffusion is understood as the process of gradually adding Gaussian noise to the data. This process is represented by a Markov chain in which the initial state is represented by the input data, each transition involves adding noise to the previous state. The authors present a diffusion probabilistic model which aims to reverse this process, that is, learn to denoise on each state. Unfortunately, this process is also intractable, as such variational inference is used during training.

2 Methodology

Given that it is intractable to learn the noise added from one state at step t to a previous one $t-1$, the authors propose the use of variational inference. This implies that such noise will be approximated using a similar distribution. Thus, a parametrized Gaussian $\mathcal{N}(x_t; \mu, \Sigma)$ will be employed, from which their mean (μ) and variance (Σ) will be approximated using a neural network. In other words, given a state at t the model aims to approximate a correct mean and variance such

that the generated noise is similar to the one added at $t-1$. In order to model the diffusion process and its denoising counterpart, a forward and backward process are presented respectively. Additionally, a further consideration is presented in order to generate the image on the final denoising step.

2.1 Forward process

A transition from one state to another by gradually adding Gaussian noise can be represented as:

$$q(x_t|x_{t-1}) := \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}) \quad (1)$$

where q represents the posterior, x_t a sample, $\mathbf{I}$ an identity matrix with dimensions equal to the samples and β the variance according to a schedule $\beta_1 ... \beta_T$.

An interesting property pointed out by [4] which considers: $\alpha_t := 1 - \beta_t$ and $\overline{\alpha}_t := \Pi_{s=1}^{t}\alpha_s$ allows to sample at an arbitrary time step from the initial sample x_0

$$q(x_t|x_0) := \mathcal{N}(x_t; \sqrt{\overline{\alpha}_t}x_0, (1-\overline{\alpha}_t)\mathbf{I}) \quad (2)$$

This technique allows to have a more efficient sampling process during training.

2.2 Backward process

A denoising transition can be modeled as:

$$p(x_{t-1}|x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \beta_t) \quad (3)$$

where θ denotes the parameters of a neural network. It's worth noting that the variance was not parametrized in this paper. In order to make tractable the diffusion process, the authors propose that a state x_{t-1} can be conditioned on their posterior x_t and initial x_0 state as

$$q(x_{t-1}|x_t, x_0) := \mathcal{N}(x_{t-1}; \overline{\mu}_t(x_t, x_0), \overline{\beta_t}\mathbf{I}) \quad (4)$$

The mean of such noise can be expressed as:

$$\overline{\mu}_t(x_t, x_0) := \frac{\sqrt{\overline{\alpha}_{t-1}}\beta_t}{1-\overline{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\overline{\alpha}_{t-1})}{1-\overline{\alpha}_t}x_t \quad (5)$$

In order to approximate a denoised sample given a current sample x_t. Considering Equation 4, the parametrized mean μ_θ should approximate $\overline{\mu}$. Therefore, a training objective can consist of minimizing a squared norm between these two terms. By re-parameterizing the current sample as:

$$x_t(x_0, \epsilon) = \sqrt{\overline{\alpha}_t}x_0 + \sqrt{1-\overline{\alpha}_t}\epsilon \quad (6)$$

with $\epsilon \sim \mathcal{N}(0, \mathbf{I})$, a noised sample is only dependent on only one noise ϵ. This allows to express equations 4 and 6 differently. The main contribution of the authors is to express the training objective in terms of ϵ_t (sampled from the forward process) and a parametrized ϵ_θ. Which has the form:

$$L(\theta) := \mathbb{E}_{t,x_0,\epsilon}\left[||\epsilon - \epsilon_\theta(x_t, t)||^2\right] \quad (7)$$

Its worth noting that ϵ is obtained from x_t using Equation 6. Finally, the mean for approximating a denoised version of the sample x_t can be computed as:

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(x_t, t)\right) \quad (8)$$

2.3 Data scaling for images

Throughout all the forward and backward process, data is scaled to be in the range $[-1, 1]$. This ensures consistency while working with the neural network. While working with images, one requirement for displaying it on a correct format is that the data must be discrete and in the range $\{0, 1, 2, ..., 255\}$. In order to achieve this scaling,

a different training objective is used on this last step. This objective consist of minimizing discrete log likelihoods between the sample and the original image.

2.4 Architecture

The architecture used on this paper implements the backbone of PixelCNN [5], which is a U-Net [6] with certain changes. First, group normalization is used instead of weight normalization. The number of residual blocks was reduced to two. And sinusoidal position embedding [7] was used on each residual block to distinguish time steps t during diffusion.

3 Experiments

3.1 Sample Quality

The model was used on different datasets to generate new images. Using the CIFAR10 dataset, the model was able to achieve a low FID score, which at the time became the best score among their competitors. By the other hand, for the inception score, it didn't score higher than its competitors. This indicated that the model was able to generate better features, but not so good as evaluated in a classifier.

It's worth noting, the inception score measures the quality of a collection of images based on how well the classification of such images performs on an inception v3 classifier. On the other hand, the FID score also uses the inception v3 classifier, but measures how similar are the features (on the last layer) generated by feeding the synthetic data in comparison to ground truth images for such class.

3.2 Reparameterization and ablation study

Considering Equation 3 it was stated that μ_θ must approximate $\overline{\mu}$ as shown in Equation 4. For such, a reparameterization of x_t considering ϵ was proposed on Equation 6. The authors wanted to see how much impact such change implied in the model. In order to measure that, they trained the model to approximate $\overline{\mu}$ directly. The results compared to just approximating ϵ show that by using the reparameterization, better scores are obtained. This indicates that by reducing the complexity of the variable to approximate, the model is able to yield more accurate results.

3.3 Progressive coding

Considering that the backward process starts from a noisy data and progresses to the generation of concrete data through the process of denoising. It can be interpreted that on each time step there is a recovery of information. In the same manner, certain information can be lost or corrupted. This is known as the rate-distortion behavior. It was demonstrated by the authors that on the initial steps of the backward pass there was high rate and low distortion, while on the latter steps low rate and high distortion. It can be interpreted that on initial steps information w.r.t the final distribution is being recovered while on latter such information is being corrupted, reason why it makes sense to use another decoder on the last step.

3.4 Interpolation

An interesting property of generative models is their ability to create an interpolated version of two sources. GANs and VAEs perform this operation by interpolating their latent vectors before

decoding it. The model of this paper takes a similar approach. First, the sources are diffused using the forward process, then interpolation is performed between the two diffused data. Finally, the interpolation is passed to the backward process. The results presented by the authors indicate that the generated data does capture features of the sources depending on the distance of interpolation. The authors further claim that during interpolation, certain artifacts could be generated, which are corrected during the backward process.

4 Conclusions

Generating data in an unsupervised manner has been an ongoing problem throughout recent years. There exists many approaches to tackle this problem. The diffusion probabilistic model presented in this paper brought a new point of view on this subject, which served as the base for other works presented in the seminar [8], [9], [10], [11] and for big projects [12] [13]. The impact of this model can be attributed to its simplicity w.r.t to the training objective and the minimal architecture it uses to approximate its objective.

By the other hand, it also has drawbacks. The first one, is that training and inference time are longer than other models like GANs. This can be attributed to the sampling of data which involve a high number of time steps on the forward and backward process. Another drawback presented in the paper was that the model was unable to be trained on a conditioned setting, reason why it was unable to compare it with another models on that setting. It was also observed that although the model obtained a good FID score, it didn't had a good inception and negative log likelihood in comparison with their competitors.

Considering the drawbacks, certain improvements can be considered and proposed. In order to tackle training time and a conditioned setting, [8] already proposes an improved version of the model, which is able to reduce training time and work on a conditioned setting. In order to improve the metrics mentioned, it can be considered the observation of the authors w.r.t the rate-distortion behavior. As it was pointed out, on later steps on the backward process there is high distortion which could be corrected using another network. Considering that the model employs a different training objective on the last step, a solution could involve to use of a decoder network on early steps when rate is low and distortion starts to rise. A good candidate could be a VAE or and auto encoder as proposed by [8].

References

[1] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[2] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[4] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *International Conference*

on Machine Learning, pp. 2256–2265, PMLR, 2015.

[5] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma, "Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications," *arXiv preprint arXiv:1701.05517*, 2017.

[6] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pp. 234–241, Springer, 2015.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[8] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models. 2022 ieee," in *CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10674–10685, 2022.

[9] A. Schneuing, Y. Du, C. Harris, A. Jamasb, I. Igashov, W. Du, T. Blundell, P. Lió, C. Gomes, M. Welling, *et al.*, "Structure-based drug design with equivariant diffusion models," *arXiv preprint arXiv:2210.13695*, 2022.

[10] L. Zhou, Y. Du, and J. Wu, "3d shape generation and completion through point-voxel diffusion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5826–5835, 2021.

[11] U. Singer, A. Polyak, T. Hayes, X. Yin, J. An, S. Zhang, Q. Hu, H. Yang, O. Ashual, O. Gafni, *et al.*, "Make-a-video: Text-to-video generation without text-video data," *arXiv preprint arXiv:2209.14792*, 2022.

[12] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, "Hierarchical text-conditional image generation with clip latents," *arXiv preprint arXiv:2204.06125*, 2022.

[13] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.