Lampyrid 2023: Volume 13, 598–604
ISSN: 2041-4900
https://lampyridjournal.com

Understanding Taxonomy of Generative Models

G. T. V. Eswar, Pankaj Kumar Singh
eswar.gtv2000@outlook.com
pankaj.e12896@cumail.in
Chandigarh University, Department of Mathematics

Abstract
The study of generative models has gained popularity in the field of machine learning. These models are valuable for a range of applications, including picture and speech synthesis, text generation, and anomaly detection, since they can produce new data that is similar to the training data. We give a thorough overview of the many generative model types in this review paper, covering pixelCNN/RNN, variational autoencoders and generative adversarial networks but the main focus is on GAN nets. We compare each model's performance on a variety of tasks and talk about its advantages, disadvantages and applications. In conclusion, this review paper offers a thorough and current summary of the state-of-the-art in generative modeling and should be helpful for scholars and practitioners interested in this fascinating and quickly developing topic.

Keywords: Generative models, PixelCNN/RNN, Generative Adversarial nets, Variational autoencoder

Introduction

After significant advancements in generative pre-trained language models, generative models have become extremely popular but are still the subject of ongoing study. Unsupervised machine learning techniques are generative models. These models acquire the capacity to stimulate the input data's probability distribution. They can be used for tasks like picture synthesis, language translation, and anomaly detection as well as to create fresh data that is identical to the training data. Different from discriminative models, which are trained to estimate the conditional probability of the output given the input, are generative models. For classification tasks like image recognition and natural language processing, discriminative models are used. They do not learn to predict the probability distribution of the input data, unlike generative models, therefore, their samples are more realistic and have exceptional colourization[1]. Generative models represent latent variables and these are used for generating data that is similar to the input data. These models can be used for a variety of tasks such as Image Synthesis, Language Modeling and Anomaly detection. However, we can classify these models as shown in figure1.

Lampyrid 2023: Volume 13, 598–604
ISSN: 2041-4900
https://lampyridjournal.com

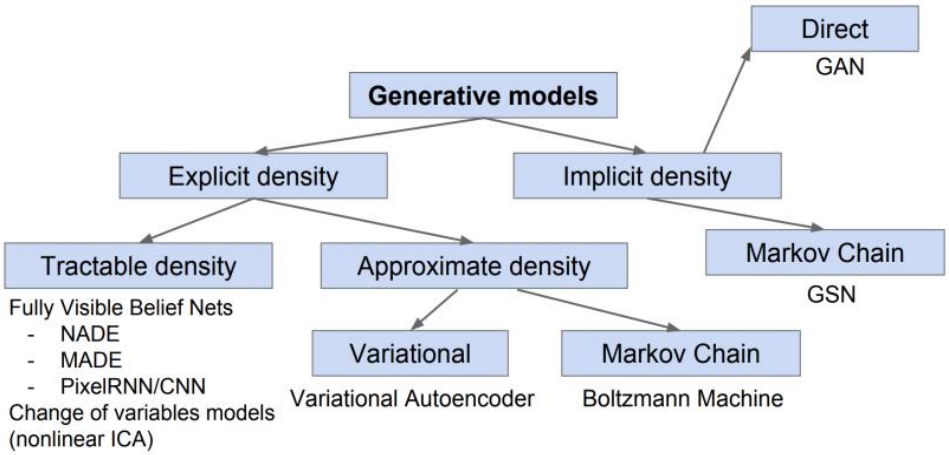

Fig1: Taxonomy of Generative models

Figure1 [2] shows the types of generative models based on their density estimations whether it addresses the density estimation explicitly or implicitly. However, this paper covers the popular and most used generative models such as PixelCNN/RNN, Autoencoders and GANs.

PixelCNN/RNN

PixelCNN is a type of deep convolutional neural network that is designed for generating images, introduced by van den Oord. This model is a conditional distribution developed over a convolution neural network or recurrent neural network [8]. These are fully visible belief networks and produce an explicit conditional probability distribution[4]. As seen in figure1 this uses an explicit density model that is it explicitly defines and solves Pmodel(x)[Generated samples]. Unlike other generative models, PixelCNN is autoregressive, meaning that it generates each pixel in an image one at a time starting from a corner[5][8]. The likelihood of an image X is defined as the conditional probability of i'th pixel Σvalue given i-1(previous pixels) p(xi/x<i)[8]. In recent years researchers have developed pixelCNN/RNN for videos and text. During training, the model is optimized to minimize the difference between the predicted pixel distribution and the actual distribution of the training data. Once trained, PixelCNN can be used to generate new images by feeding in a partial image and then iteratively generating each missing pixel based on the predicted distribution.

The overall PixelCNN architecture typically consists of numerous output layers, followed by layers of masked convolutional layers. To capture dependencies between further away pixels, each layer of masked convolutional layers typically uses ever larger convolutional filters. These masked convolutions are known as segmentation-aware convolutions in the context of image segmentation[3]. Moreover, a rectified linear unit (ReLU) activation function is often used for the outputs of each layer to add nonlinearity to the model. At the conclusion of the model, a probability function is generated for each pixel of the picture x that can produce an image.

Autoencoders

Autoencoder is an unsupervised neural network algorithm that learns the latent representation of input data by transforming it from its original form to a lower-dimensional representation before recreating it. Minimizing the reconstruction error between the input and the output is the goal of an autoencoder. The autoencoder consists of two parts: an encoder and a decoder. The encoder creates latent variables, while the decoders only serve to reconstruct the loss function, also known as a generative model[9]. Applications including image denoising, dimensionality reduction, and anomaly detection have all made extensive use of autoencoders. They do, however, have certain drawbacks, including the inability to learn complicated characteristics and the propensity to capture the mean of the input data rather than its diversity.

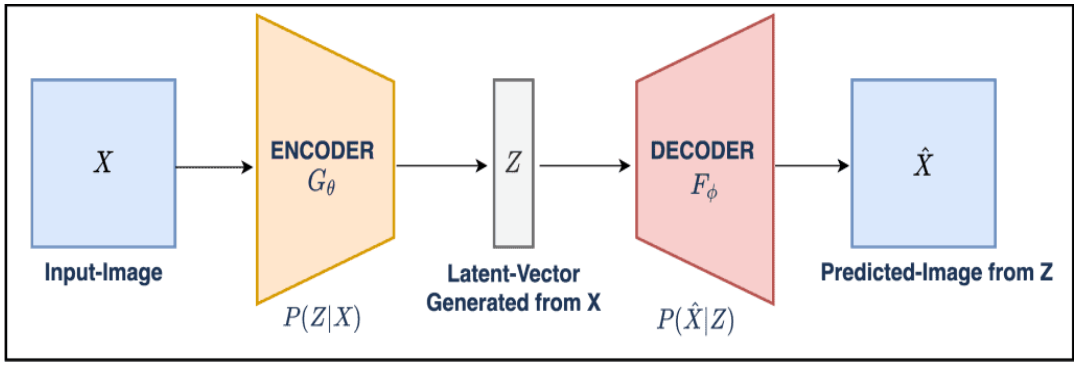

Fig2: An Autoencoder

Types of autoencoders:

Variational autoencoders:

Variational autoencoders are a probabilistic spin of autoencoders, I.e will let us sample from the model to generate data and a principled approach to generative models. They are a type of autoencoder, a particular kind of neural network that learns to reduce the dimensions of input data before reconstructing the original data from this reduced form. VAEs learn to encode data into a probabilistic latent space, which is the primary distinction between VAEs and conventional autoencoders. This implies that VAEs learn a distribution over potential latent representations rather than only learning a deterministic representation of the input. This probabilistic encoding allows for more flexible and robust representations[10], which can be sampled to generate new, realistic data.

Since VAE is an autoencoder it estimates the density model via decoder and latent representation via encoder[12][13]. The density function variational autoencoders can be written as $P\theta(x) = \int P\theta(z)P\theta(x/z)dz$ [11], and the encoder network can be derived as $P(Z/X) = N(\mu(z/x)\ \Sigma(z/x))$ and decoder equation is $P(X/Z) = N(\mu(x/z)\ \Sigma(x/z)$. Generative models represent an image as Xn where n denotes the number of images $n = 1,2,3,\ldots,N$. Although variational autoencoders produce

better samples than pixelCNN/RNN, they are blurrier and of worse quality when compared to GANs, which are state-of-the-art.

VQ Variational Autoencoder (Video Generation):
In 2017, Van den Oord [17] proposed the vector quantized VAE. Subsequent researchers improved this technique for text, audio, and video production. Given that this model is an autoencoder, it has a structure that is comparable to autoencoders, however, vector-quantized VAE stand out by discovering discrete latent variables. This benefit led Wilson Yan and the team to successfully implement video GPT[16] and identified video creation as the next significant problem for generative algorithms.
The encoder E(x) of the VQ VAE transforms information from the input layer into discretized latent variables, while the decoder D(e) uses the latent variables to reconstruct the input data x using quantized encoding. This approach uses 3D or transposed convolutional nets for each layer[18].

Applications of Autoencoders:
A group of Google researchers developed MusicLM[19] in 2017, which uses autoencoders to create conditional music from the audio input. In 2020, Prafulla Dhariwal proposed a vector quantized variational autoencoder-based jukebox[20] to produce music. They were motivated by the VQ VAE's hierarchical framework for producing images.

Generative Adversarial Networks(GANs)
GANs are Generative models for producing synthetic data that resembles actual data. Ian Goodfellow originally described GANs in 2014. As seen in fig. 1, explicit density functions are not compatible with GANs, Instead, they play a two-player game to learn how to produce from training distribution. The Generator network and the Discriminator network are the two components of GAN.

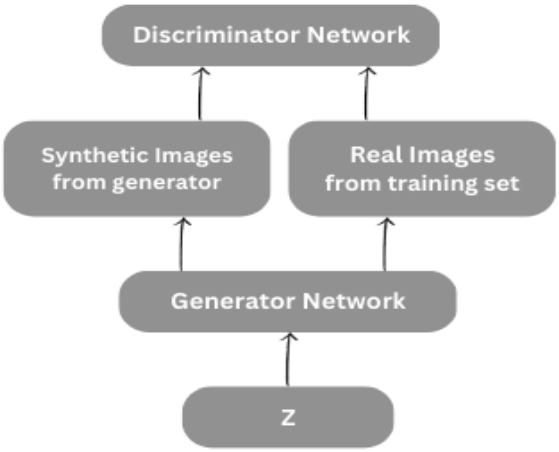

Fig3: Workflow of GAN

Generator: The work of this network is to generate fake synthetic images that are nearly close to natural images, generative network is well performed when the output of it is very close to real images.

Discriminator: this network matches both the generator's output and real images [7], tries to distinguish between both images and produces a loss function. However, the discriminative network output is categorical, either True (1) or False (0), which represents a real or synthetic fake image.

When training begins, the generator Gθ produces fake data, and the discriminator tries to predict whether it is fake(0) or real(1).

Fig4: Overview of GAN Structure

The discriminator maximizes correctly predicting D(x) such that it is close to 1, but when the generator performs well in generating fake synthetic images, the accuracy of the discriminator decreases, and the model is successful. A well-performed model output is as follows

As per James Jordan the generator Gθ takes the random noise z and sends it to the discriminant along with the input real image [6]. The training of the GAN network involves training jointly both networks by the minimax method, which can be defined as

$\min_{\theta g} \max_{\theta d}[E_{x\sim pdata} \log D_{\theta d}(x) + E_{z\sim p(z)} \log(1-D_{\theta d}(G_{\theta g}(z)))]$

Dθ referred as discriminator network and Gθ is generative network,[14] for discriminator we use gradient ascent

$\max_{\theta d} [E_{x\sim pdata} \log D_{\theta d}(x)+E_{z\sim p(z)} \log(1-D_{\theta d}(G_{\theta g}(z)))]$

And generator's gradient accent can be derived as

$\max_{\theta g}[E_{z\sim p(z)}\log(D_{\theta d}(G_{\theta g}(z)))]$

Applications of GAN:

Uses of GAN networks include text-to-image conversion, the creation of animated characters, the creation of three-dimensional objects, and many other things. In a publication, Jinsung Yoon and his team used a GAN network, called GAIN

(Generative Adversarial Imputation Nets)[6] to impute missing data. The development process occurs as the discriminator network compares the generated values to the absorbed values and the generator network generates imputed values for missing data.

References

[1] Odena, A., Olah, C., & Shlens, J. (2016). Conditional Image Synthesis With Auxiliary Classifier GANs. ArXiv. /abs/1610.09585

[2] Ian Goodfellow, Tutorial on Generative Adversarial /networks, 2017

[3] Liu, Guilin, et al. "Image inpainting for irregular holes using partial convolutions." Proceedings of the European conference on computer vision (ECCV). 2018.

[4] Zhang, S., Yang, Z., Tu, H., Yang, J., & Huang, Y. (2021). Pixel-Stega: Generative Image Steganography Based on Autoregressive Models. ArXiv. /abs/2112.10945

[5] Salimans, T., Karpathy, A., Chen, X., & Kingma, D. P. (2017). PixelCNN++: Improving the PixelCNN with Discretized Logistic Mixture Likelihood and Other Modifications. ArXiv. /abs/1701.05517

[6] Yoon, J., & Jordon, J. (2018). GAIN: Missing Data Imputation using Generative Adversarial Nets. ArXiv. /abs/1806.02920

[7] Kahng, M., Thorat, N., Chau, D. H., Viégas, F., & Wattenberg, M. (2018). GAN Lab: Understanding Complex Deep Generative Models using Interactive Visual Experimentation. ArXiv. https://doi.org/10.1109/TVCG.2018.2864500

[8] Van Den Oord, Aäron, Nal Kalchbrenner, and Koray Kavukcuoglu. "Pixel recurrent neural networks." International conference on machine learning. PMLR, 2016.

[9] Pu, Y., Gan, Z., Henao, R., Yuan, X., Li, C., Stevens, A., & Carin, L. (2016). Variational Autoencoder for Deep Learning of Images, Labels and Captions. ArXiv. /abs/1609.08976

[10] Casale, F. P., Dalca, A. V., Saglietti, L., Listgarten, J., & Fusi, N. (2018). Gaussian Process Prior Variational Autoencoders. ArXiv. /abs/1810.11738

[11] Kingma, D. P., & Welling, M. (2013). Auto-Encoding Variational Bayes. ArXiv. /abs/1312.6114

[12] Cemgil, A. T., Ghaisas, S., Dvijotham, K., Gowal, S., & Kohli, P. (2020). Autoencoding Variational Autoencoder. ArXiv. /abs/2012.03715

[13] Pu, Y., Gan, Z., Henao, R., Yuan, X., Li, C., Stevens, A., & Carin, L. (2016). Variational Autoencoder for Deep Learning of Images, Labels and Captions. ArXiv. /abs/1609.08976

[14] Goodfellow, I. J., Mirza, M., Xu, B., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative Adversarial Networks. ArXiv. /abs/1406.2661

[15] Ö. Kırbıyık, E. Simsar and A. T. Cemgil, "Comparison of Deep Generative Models for the Generation of Handwritten Character Images," 2019 27th

Signal Processing and Communications Applications Conference (SIU), Sivas, Turkey, 2019, pp. 1-4, doi: 10.1109/SIU.2019.8806416.

[16] Yan, W., Zhang, Y., Abbeel, P., & Srinivas, A. (2021). VideoGPT: Video Generation using VQ-VAE and Transformers. ArXiv. /abs/2104.10157

[17] Van Den Oord, A., Vinyals, O., et al. Neural discrete representation learning. In Advances in Neural Information Processing Systems, pp. 6306–6315, 2017

[18] Tran, D., Bourdev, L., Fergus, R., Torresani, L., and Paluri, M. Learning spatiotemporal features with 3d convolutional networks. In Proceedings of the IEEE international conference on computer vision, pp. 4489–4497, 2015.

[19] Agostinelli, A., Denk, T. I., Borsos, Z., Engel, J., Verzetti, M., Caillon, A., Huang, Q., Jansen, A., Roberts, A., Tagliasacchi, M., Sharifi, M., Zeghidour, N., & Frank, C. (2023). MusicLM: Generating Music From Text. ArXiv. /abs/2301.11325

[20] Dhariwal, P., Jun, H., Payne, C., Kim, J. W., Radford, A., and Sutskever, I. Jukebox: A generative model for music. arXiv preprint arXiv:2005.00341, 2020.