Generative Modeling

Danilo Comminiello

Generative Deep Learning 2021/2022

June 6, 2022

Sapienza
Università di Roma

Lecture content highlights

- Generative modeling is an important branch of machine learning that complements the more widely studied discriminative modeling.

- Classic machine learning models may be not able to produce a probability distribution that represents inherent structure in the data and to generate examples outside of the training set.

- Generative modeling challenges arise in “modern” AI problems, when we need to deal with big data and complex models.

Table of contents

1 Generative Modeling

Why study generative models?

- Excellent test of our ability to use high-dimensional, complicated probability distributions
- Simulate possible futures for planning or simulated RL
- Missing data
- Semi-supervised learning
- Multi-modal outputs
- Realistic generation tasks

Discriminative modeling

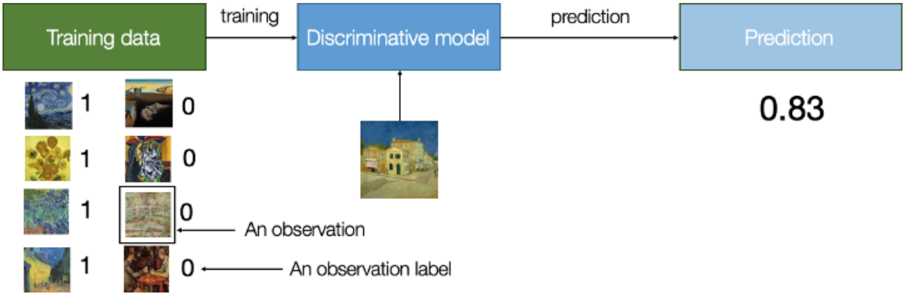


Figure 1: **Discriminative modeling** usually refers to *supervised learning*, or learning a function that maps an input to an output using a labeled dataset [1].

Generative modeling

A **generative model** must also be *probabilistic* rather than *deterministic*.

A deterministic model is merely characterized by a fixed calculation.

A generative model must include a stochastic element affecting the individual samples.

Unlike discriminative modeling, generative modeling is usually performed with an unlabeled dataset.

The generative modeling process

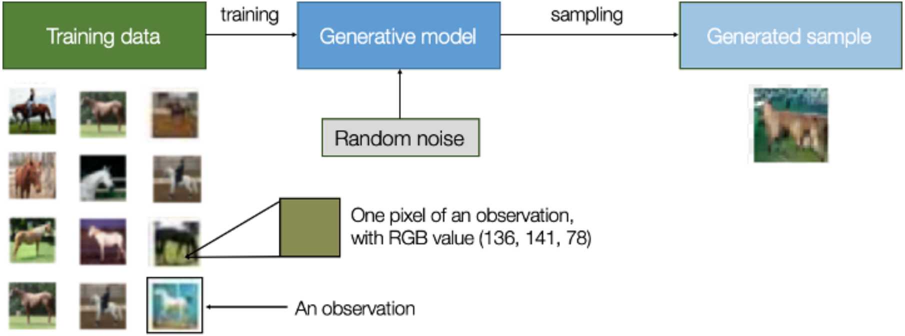


Figure 2: A **generative model** describes how a dataset is generated, in terms of a probabilistic model. By sampling from this model, we can generate new data [1].

Generative vs discriminative

Discriminative modeling estimates the probability of a label y given observation, i.e.: $p\left(y|\mathbf{x}\right)$.

Generative modeling estimates the probability of observing observation $\mathbf{x}$, i.e.: $p\left(\mathbf{x}\right)$.

If the dataset is labeled, we can also build a generative model that estimates the distribution $p\left(\mathbf{x}|y\right)$.

Generative modeling doesn't care about labeling observations. Instead, it attempts to estimate the probability of seeing the observation at all.

Why generative modeling is attractive

- Generative processes are able to express physical laws, while considering meaningless details as *noise*.
- Generative models are usually highly intuitive and interpretable.
- Generative processes express causal relations, thus being able to generalize much better to new situations than mere correlations.

The success of generative models has been boosted by several big companies, like NVIDIA [2, 3].

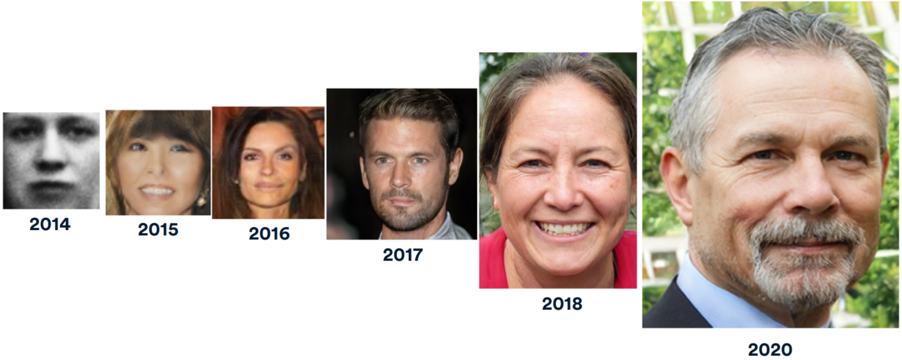


Figure 3: Face generation has been one of the most explored applications of the last years [1, 4].

(a)

(b)

Figure 4: Samples generated by: (a) a deep hierachical variational autoencoder trained on the CelebA-HQ dataset [5]; (b) a U-Net generative adversarial network on the COCO Animals dataset [6].

Generative modeling framework

A generative framework can be summarize as:

- We consider a dataset of observation $\mathbf{X}$.
- We assume $\mathbf{X}$ drawn from some unknown distribution p_{data}.
- A generative model p_{model} attempts to estimate p_{data}, thus generating observations that appear to have been drawn from p_{data}.

An impressive generative model p_{model} is able to:

– generate examples that appear to have been drawn from p_{data};

– generate examples that didn't exist before in $\mathbf{X}$.

Example of generative modeling framework

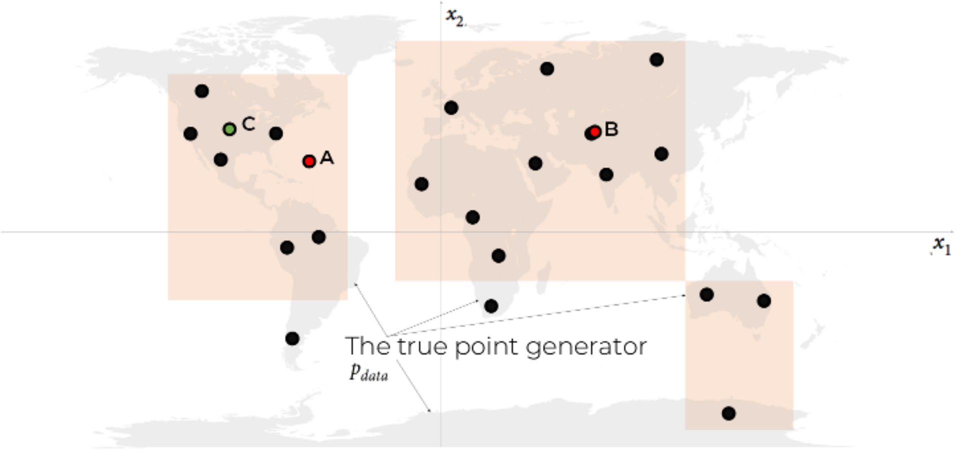


Figure 5: The orange box, p_{model}, is an estimate of the true data-generating distribution p_{data} (the gray area). The data-generating rule is simply a *uniform distribution* over the land mass of the world, with no chance of finding a point in the sea [1].

Some application tasks

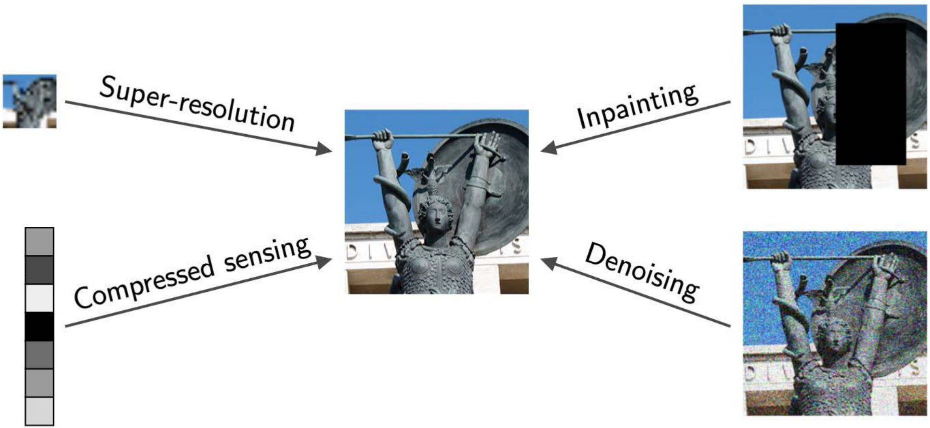
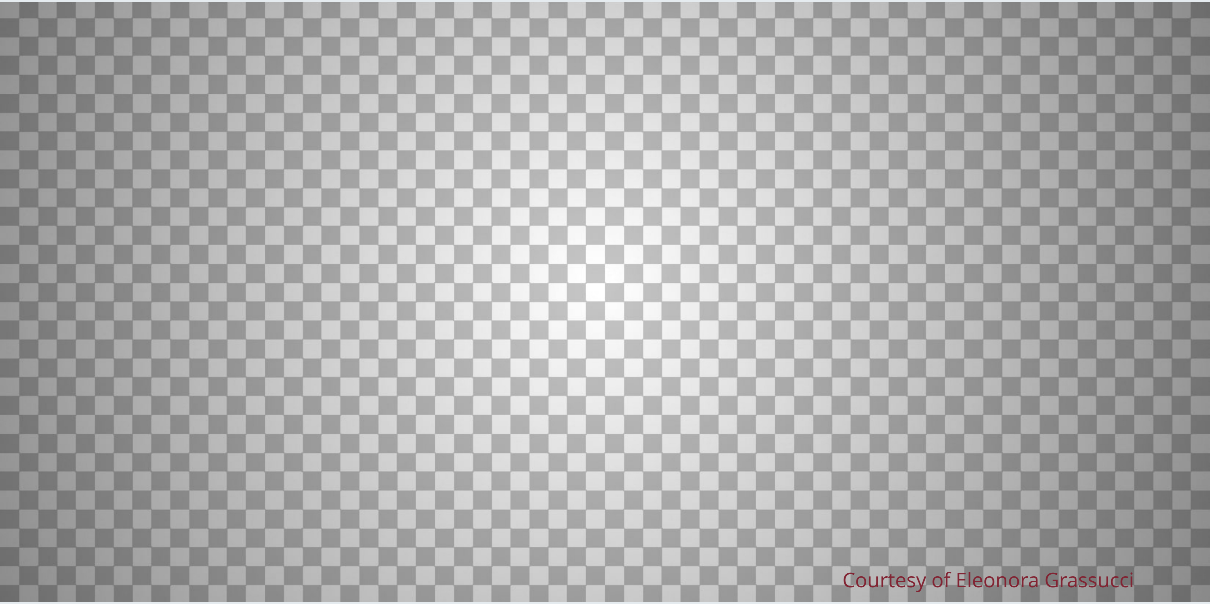


Figure 6: Application tasks may include super-resolution, compressed sensing, inpainting, denoising, data augmentation, image-to-image translation and domain translation, among others. Courtesy of Christian Marinoni [7].

This slide does not exist



Courtesy of Eleonora Grassucci

2 Probabilistic Generative Models

- Reasoning About Certainty
- Basic Concepts on Probability
- Bayesian Generative Modeling
- Challenges of Generative Modeling

Reasoning about certainty I

Machine learning (ML) is all about making predictions.

We might use probability many ML applications, as well in several real-life situations.

Probability gives us a formal way of reasoning about our **level of certainty**.

As an example, consider distinguishing cats and dogs based on photographs. Although, it might sound simple, the difficulty of the problem may depend on the resolution of the image.

Reasoning about certainty II

Figure 7: Our ability to distinguish cats and dogs apart at a low resolution might approach uninformed guessing [8].

Reasoning about certainty III

- If we are completely sure that the image depicts a cat, we say that the *probability* that the corresponding label y is "cat", denoted $p(y = \text{"cat"})$ equals 1.

- If we had no evidence to suggest that $y = \text{"cat"}$ or that $y = \text{"dog"}$, then we might say that the two possibilities were equally *likely* expressing this as $p(y = \text{"cat"}) = p(y = \text{"dog"}) = 0.5$.

- If we were reasonably confident, but not sure that the image depicted a cat, we might assign a probability $0.5 < p(y = \text{"cat"}) < 1$.

Probability is a flexible language for reasoning about our level of certainty, and it can be applied effectively in several contexts in ML.

Stochastic events and sample space

In probability[1] theory, an *observable phenomenon*, or *occurrence*, or **stochastic event** $\mathbf{x}$ is considered based on the possibility that it may occur or not.

An event can be seen as the outcome of an experiment, also said a *random variable* (RV).

The set of all the possible experiments is the *abstract probability space*, or **sample space**, Ω.

Thus, the sample space contains all the values an observation $\mathbf{x}$ can take.

[1] From the Latin *probare*: test, try, and *ilis*: to be able to.

Probability and its interpretations

The **probability density function**, $p(\mathbf{x})$, is a function that maps a point $\mathbf{x}$ in the sample space into a number between 0 and 1.

A **parametric model**, $p_\theta(\mathbf{x})$, is a family of density functions that can be described using a finite number of parameters, θ.

There are actually at least two different interpretations of probability.

- **Frequentist** interpretation: to represent long run *frequencies of events*.
- **Bayesian** interpretation: to quantify our *uncertainty* about something.

> Probability theory is nothing but common sense reduced to calculation.
> *– Pierre Laplace, 1812*

Set of probabilities

Random variables can be described in terms of a set of probabilities.

- The **joint probability** of $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ represents the probability that both the events occur simultaneously:

$$p(x, y) = p(x \cap y) = p(x|y)\, p(y)\,. \tag{1}$$

- Given $p(x, y)$, we can define the **marginal probability** of x as:

$$p(x) = \sum_{x \in \mathcal{Y}} p(x, y) = \sum_{x \in \mathcal{Y}} p(x|y)\, p(y) \tag{2}$$

  also known as *sum rule*. We can define $p(y)$ similarly.

- The **conditional probability** of x, given y is true, is defined as:

$$p(x|y) = \frac{p(x, y)}{p(y)} \qquad \text{if } p(y) > 0. \tag{3}$$

Likelihood

The **likelihood** $\mathcal{L}(\theta|\mathbf{x})$ of a parameter set θ is a function that measures the plausibility of θ, given some observed point $\mathbf{x}$.

It is defined as the value of the density function parameterized by θ, i.e.: $\mathcal{L}(\theta|\mathbf{x}) = p_\theta(\mathbf{x})$.

For a whole dataset $\mathbf{X}$ of independent observation we have:

$$\mathcal{L}(\theta|\mathbf{X}) = \prod_{\mathbf{x}\in\mathbf{X}} p_\theta(\mathbf{x}) \,.$$

Since this product can be quite computationally difficult to work with, the **log-likelihood** $\ell(\theta|\mathbf{X})$ is often used instead:

$$\ell(\theta|\mathbf{X}) = \sum_{\mathbf{x}\in\mathbf{X}} \log p_\theta(\mathbf{x}) \,.$$

Maximum likelihood estimation

The **maximum likelihood estimation** allows to estimate the set of parameters θ of a density function, $p_\theta(x)$, that are most likely to explain some observed data $\mathbf{X}$:

$$\hat{\theta} = \arg\max_{\theta} \mathcal{L}(\theta|\mathbf{X})$$

where $\hat{\theta}$ is the maximum likelihood estimate (MLE).

Bayesian generative modeling

One of the most popular examples of generative model in machine learning is represented by the Naive Bayes algorithm.

The basic assumption is that the features (i.e., RVs), x_k, for $k = 1, \ldots, l$, in the feature vector $\mathbf{x}$ are statistically independent, i.e.:

$$p(x_i, x_j) = p(x_i)\, p(x_j), \qquad \forall i, j = 1, \ldots, l, \quad i \neq j \tag{4}$$

Bayes rule

The uncertainty of an output variable y_i, $i = 1, \ldots, M$, given the value of a feature vector $\mathbf{x}$, as expressed by the conditional probability $p(y_i|\mathbf{x})$, can be equivalently expressed as:

$$p(y_i|\mathbf{x}) = \frac{p(\mathbf{x}|y_i)\, p(y_i)}{p(\mathbf{x})}. \tag{5}$$

In (5), $p(\mathbf{x})$ and $p(y_i)$ are also denoted as prior probabilities, $p(y_i|\mathbf{x})$ as posterior probability and $p(\mathbf{x}|y_i)$ as likelihood.

The joint pdf can be written as a product of l marginals, i.e.:

$$p(\mathbf{x}|y_i) = \prod_{k=1}^{l} p_\theta(x_k|y_i), \quad i = 1, 2, \ldots, M.$$

This leads to a total of $2l$ unknown parameters to be estimated per class.

Challenges of generative modeling

A generative model must overcome two key challenges in order to be successful:

- How does the model cope with the high degree of conditional dependence between features?
- How does the model find a tiny proportion of the high-dimensional sample space satisfying possible generated observations?

Deep learning is the key to solving both of these challenges due to its ability to form its own features in a lower-dimensional space.

Next lecture

- We will introduce deep representation learning, which is one of the key concepts to overcome the generative modeling challenges.
  - We see how to exploit the latent space to extract useful information from data.
- Deep generative models are built by stacking layers, as any *deep neural network*.
  - We will see how to merge *generative modeling* and *deep learning* to build deep generative models.

References I

[1] D. Foster, *Generative Deep Learning – Teaching Machines to Paint, Write, Compose and Play*. O'Reilly Media, Inc., Jun. 2019.

[2] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," *arXiv preprint arXiv:1812.04948v1*, Dec. 2018.

[3] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *preprint*, 2019.

[4] D. Zhang, S. Mishra, E. Brynjolfsson, J. Etchemendy, D. Ganguli, B. Grosz, T. Lyons, J. Manyika, J. C. Niebles, M. Sellitto, Y. Shoham, J. Clark, and R. Perrault, "The AI index 2021 annual report," *arXiv preprint arXiv:2103.06312*, Mar. 2021.

[5] A. Vahdat and J. Kautz, "NVAE: A deep hierarchical variational autoencoder," in *Neural Information Processing Systems (NeurIPS)*, 2020.

[6] E. Schönfeld, B. Schiele, and A. Khoreva, "A U-Net based discriminator for generative adversarial networks," in *IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 8207–8216.

[7] C. Marinoni, "Attention-based generative deep learning for reference-guided image-to-image translation," Master's thesis, Sapienza University of Rome, Italy, 2022.

References II

[8] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, *Dive Into Deep Learning*, 2020.

[9] I. J. Goodfellow, "Generative adversarial networks (GANs) – NIPS 2016 tutorial," Dec. 2016.

[10] D. P. Kingma and M. Welling, "An introduction to variational autoencoders," *Foundations and Trends in Machine Learning*, vol. 12, no. 4, pp. 307–392, Nov. 2019.

Generative Modeling

Generative Deep Learning
2021/2022

Danilo Comminiello

PhD Course in Information and Communication Technology (ICT)

http://danilocomminiello.site.uniroma1.it

danilo.comminiello@uniroma1.it