Machine Learning (part II)

Diffusion models

Angelo Ciaramella

Università Parthenope
DiST
Dipartimento di Scienze e Tecnologie

Introduction

- Denoising Diffusion Probabilistic Models
  - take each training image and to corrupt it using a multi-step noise process to transform it into a sample from a Gaussian distribution
  - a DNN is then trained to invert this process, and once trained the network can then generate new images starting with samples from a Gaussian as input

Encoding process

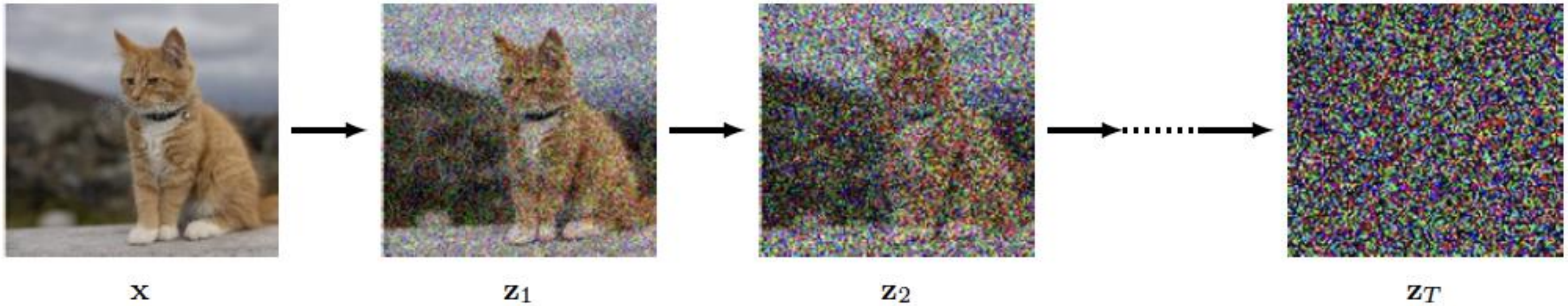

Figure 20.1 Illustration of the encoding process in a diffusion model showing an image $\mathbf{x}$ that is gradually corrupted with multiple stages of additive Gaussian noise giving a sequence of increasingly noisy images. After a large number T of steps the result is indistinguishable from a sample drawn from a Gaussian distribution. A deep neural network is then trained to reverse this process.

Forward encoder

- Image $\mathbf{x}$ from the training set

$$\mathbf{z}_1 = \sqrt{1-\beta_1}\mathbf{x} + \sqrt{\beta_1}\epsilon_1$$

$$\epsilon_1 \sim \mathcal{N}(\epsilon_1|\mathbf{0}, \mathbf{I})$$

$$\beta_1 < 1$$

variance of the noise distribution

z_t with zero mean and unit variance

- rewriting the transformation

$$q(\mathbf{z}_1|\mathbf{x}) = \mathcal{N}(\mathbf{z}_1|\sqrt{1-\beta_1}\mathbf{x}, \beta_1\mathbf{I})$$

Markov chain

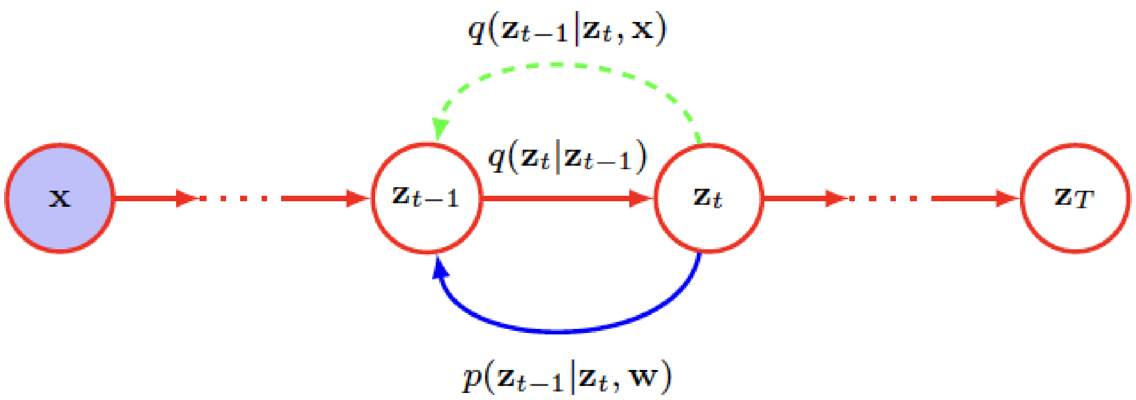

Figure 20.2 A diffusion process represented as a probabilistic graphical model. The original image $\mathbf{x}$ is shown by the shaded node, since it is an observed variable, whereas the noise-corrupted images $\mathbf{z}_1, \ldots, \mathbf{z}_T$ are considered to be latent variables. The noise process is defined by the forward distribution $q(\mathbf{z}_t|\mathbf{z}_{t-1})$ and can be viewed as an encoder. Our goal is to learn a model $p(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{w})$ that tries to reverse this noise process and which can be viewed as a decoder. As we will see later, the conditional distribution $q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{x})$ plays an important role in defining the training procedure.

$$\mathbf{z}_t = \sqrt{1-\beta_t}\mathbf{z}_{t-1} + \sqrt{\beta_t}\epsilon_t$$

$$q(\mathbf{z}_t|\mathbf{z}_{t-1}) = \mathcal{N}(\mathbf{z}_t|\sqrt{1-\beta_t}\mathbf{z}_{t-1}, \beta_t\mathbf{I})$$

Diffusion kernel

- Joint distribution of the latent variables

$$q(\mathbf{z}_1, \ldots, \mathbf{z}_t|\mathbf{x}) = q(\mathbf{z}_1|\mathbf{x}) \prod_{\tau=2}^{t} q(\mathbf{z}_\tau|\mathbf{z}_{\tau-1})$$

- Marginalizing over the intermediate variables z_1, . . . , z_{t-1} we obtain the diffusion kernel

$$q(\mathbf{z}_t|\mathbf{x}) = \mathcal{N}(\mathbf{z}_t|\sqrt{\alpha_t}\mathbf{x}, (1-\alpha_t)\mathbf{I})$$

$$\alpha_t = \prod_{\tau=1}^{t}(1-\beta_\tau)$$

Diffusion kernel

- After many steps the image becomes indistinguishable from Gaussian noise

$$T \to \infty$$

$$q(\mathbf{z}_T|\mathbf{x}) = \mathcal{N}(\mathbf{z}_T|\mathbf{0}, \mathbf{I})$$

- Independence of x

$$q(\mathbf{z}_T) = \mathcal{N}(\mathbf{z}_T|\mathbf{0}, \mathbf{I})$$

Conditiona distribution

- Using Bayes' theorem reversing the conditonal distribution

$$q(\mathbf{z}_{t-1}|\mathbf{z}_t) = \frac{q(\mathbf{z}_t|\mathbf{z}_{t-1})q(\mathbf{z}_{t-1})}{q(\mathbf{z}_t)}$$

intractable for p(**x**)

$$q(\mathbf{z}_{t-1}) = \int q(\mathbf{z}_{t-1}|\mathbf{x})p(\mathbf{x})\,\mathrm{d}\mathbf{x}$$

$$q(\mathbf{z}_t|\mathbf{x}) = \mathcal{N}(\mathbf{z}_t|\sqrt{\alpha_t}\mathbf{x}, (1-\alpha_t)\mathbf{I})$$

Conditiona distribution

- Using Bayes' theorem

$$q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{x}) = \frac{q(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{x})q(\mathbf{z}_{t-1}|\mathbf{x})}{q(\mathbf{z}_t|\mathbf{x})}$$

$$q(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{x}) = q(\mathbf{z}_t|\mathbf{z}_{t-1})$$

$$q(\mathbf{z}_t|\mathbf{z}_{t-1}) = \mathcal{N}(\mathbf{z}_t|\sqrt{1-\beta_t}\mathbf{z}_{t-1}, \beta_t\mathbf{I})$$

Reverse decoder

- Revers process by Gaussian distribution

$$p(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{w}) = \mathcal{N}(\mathbf{z}_{t-1}|\boldsymbol{\mu}(\mathbf{z}_t, \mathbf{w}, t), \beta_t \mathbf{I})$$

deep neural network governed by a set of parameters w

- reverse denoising process then takes the form of a Markov chain given by

$$p(\mathbf{x}, \mathbf{z}_1, \ldots, \mathbf{z}_T|\mathbf{w}) = p(\mathbf{z}_T) \left\{ \prod_{t=2}^{T} p(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{w}) \right\} p(\mathbf{x}|\mathbf{z}_1, \mathbf{w})$$

network takes the step index t explicitly as an input so that it can account
for the variation of the variance across different steps of the chain.
This allows us to use a single network to invert all the steps in the Markov chain

Training the decoder

- Objective function for training the NN (likelihood)

$$p(\mathbf{x}|\mathbf{w}) = \int \cdots \int p(\mathbf{x}, \mathbf{z}_1, \ldots, \mathbf{z}_T|\mathbf{w})\, \mathrm{d}\mathbf{z}_1 \ldots \mathrm{d}\mathbf{z}_T$$

  - the likelihood is intractable

ELBO

- Evidence Lower Bound (ELBO)

$$\ln p(\mathbf{x}|\mathbf{w}) = \mathcal{L}(\mathbf{w}) + \mathrm{KL}\left(q(\mathbf{z}) \| p(\mathbf{z}|\mathbf{x}, \mathbf{w})\right)$$

$$\mathcal{L}(\mathbf{w}) = \int q(\mathbf{z}) \ln \left\{ \frac{p(\mathbf{x}, \mathbf{z}|\mathbf{w})}{q(\mathbf{z})} \right\} \mathrm{d}\mathbf{z}$$

$$\mathrm{KL}\left(f(\mathbf{z}) \| g(\mathbf{z})\right) = -\int f(\mathbf{z}) \ln \left\{ \frac{g(\mathbf{z})}{f(\mathbf{z})} \right\} \mathrm{d}\mathbf{z}$$

- from

$$p(\mathbf{x}, \mathbf{z}|\mathbf{w}) = p(\mathbf{z}|\mathbf{x}, \mathbf{w}) p(\mathbf{x}|\mathbf{w})$$

- we obtain

$$\ln p(\mathbf{x}|\mathbf{w}) \geqslant \mathcal{L}(\mathbf{w})$$

ELBO

$$\mathcal{L}(\mathbf{w}) = \underbrace{\int q(\mathbf{z}_1|\mathbf{x}) \ln p(\mathbf{x}|\mathbf{z}_1, \mathbf{w})\, \mathrm{d}\mathbf{z}_1}_{\text{reconstruction term}}$$

$$- \underbrace{\sum_{t=2}^{T} \int \mathrm{KL}(q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{x}) \| p(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{w}))q(\mathbf{z}_t|\mathbf{x})\, \mathrm{d}\mathbf{z}_t}_{\text{consistency terms}}$$

ELBO

Algorithm 20.1: Training a denoising diffusion probabilistic model

Input: Training data $\mathcal{D} = \{\mathbf{x}_n\}$
Noise schedule $\{\beta_1, \ldots, \beta_T\}$
Output: Network parameters $\mathbf{w}$

---

for $t \in \{1, \ldots, T\}$ **do**
  $\alpha_t \leftarrow \prod_{\tau=1}^{t}(1-\beta_\tau)$ `// Calculate alphas from betas`
end for
repeat
  $\mathbf{x} \sim \mathcal{D}$ `// Sample a data point`
  $t \sim \{1, \ldots, T\}$ `// Sample a point along the Markov chain`
  $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}|\mathbf{0}, \mathbf{I})$ `// Sample a noise vector`
  $\mathbf{z}_t \leftarrow \sqrt{\alpha_t}\mathbf{x} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}$ `// Evaluate noisy latent variable`
  $\mathcal{L}(\mathbf{w}) \leftarrow \|\mathbf{g}(\mathbf{z}_t, \mathbf{w}, t) - \boldsymbol{\epsilon}\|^2$ `// Compute loss term`
  Take optimizer step
until converged
return $\mathbf{w}$

ELBO

Algorithm 20.2: Sampling from a denoising diffusion probabilistic model

Input: Trained denoising network $\mathbf{g}(\mathbf{z}, \mathbf{w}, t)$

Noise schedule $\{\beta_1, \ldots, \beta_T\}$

Output: Sample vector $\mathbf{x}$ in data space

$\mathbf{z}_T \sim \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$ `// Sample from final latent space`

for $t \in T, \ldots, 2$ **do**

$\alpha_t \leftarrow \prod_{\tau=1}^{t}(1 - \beta_\tau)$ `// Calculate alpha`

`// Evaluate network output`

$\boldsymbol{\mu}(\mathbf{z}_t, \mathbf{w}, t) \leftarrow \frac{1}{\sqrt{1-\beta_t}}\left\{\mathbf{z}_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\mathbf{g}(\mathbf{z}_t, \mathbf{w}, t)\right\}$

$\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}|\mathbf{0}, \mathbf{I})$ `// Sample a noise vector`

$\mathbf{z}_{t-1} \leftarrow \boldsymbol{\mu}(\mathbf{z}_t, \mathbf{w}, t) + \sqrt{\beta_t}\boldsymbol{\epsilon}$ `// Add scaled noise`

end for

$\mathbf{x} = \frac{1}{\sqrt{1-\beta_1}}\left\{\mathbf{z}_1 - \frac{\beta_1}{\sqrt{1-\alpha_1}}\mathbf{g}(\mathbf{z}_1, \mathbf{w}, t)\right\}$ `// Final denoising step`

return $\mathbf{x}$