National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"

Institute of Physics and Technology

Intellectual Data Analysis

Practice 8: Image Generation with Neural Nets

Dr. Nataliya K. Sakhnenko

Please review Lectures 12-13 before this practical

Neural Style Transfer

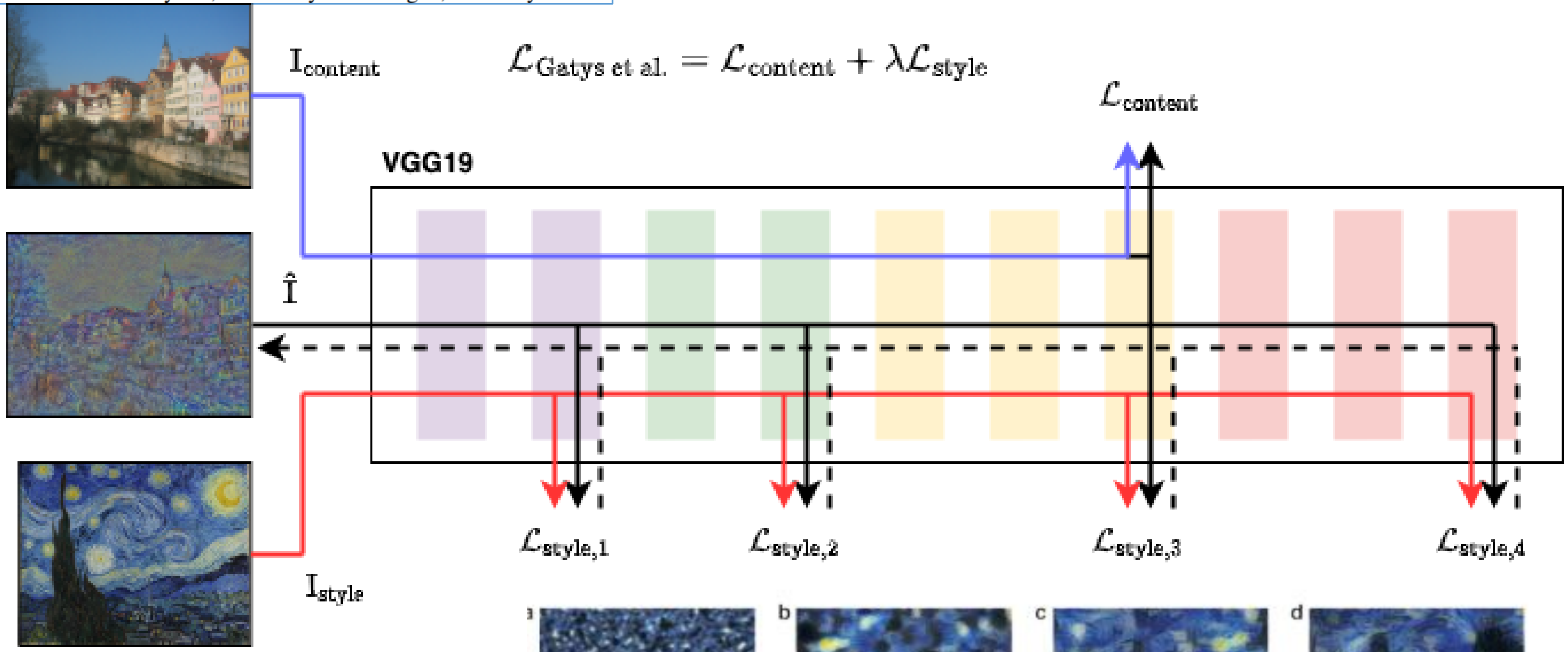
A Neural Algorithm of Artistic Style

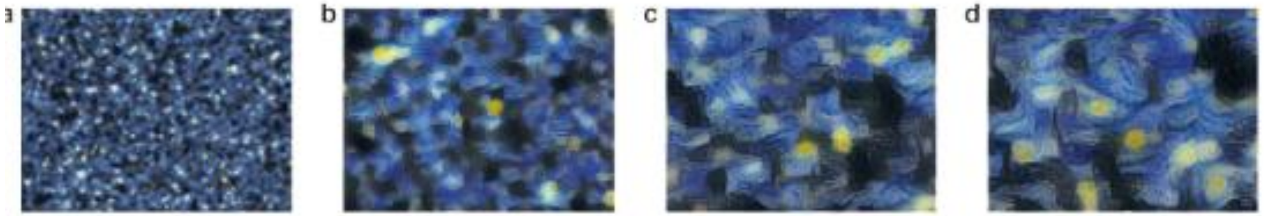
Leon A. Gatys,[1,2,3*] Alexander S. Ecker,[1,2,4,5] Matthias Bethge[1,2,4]

[1]Werner Reichardt Centre for Integrative Neuroscience
and Institute of Theoretical Physics, University of Tübingen, Germany

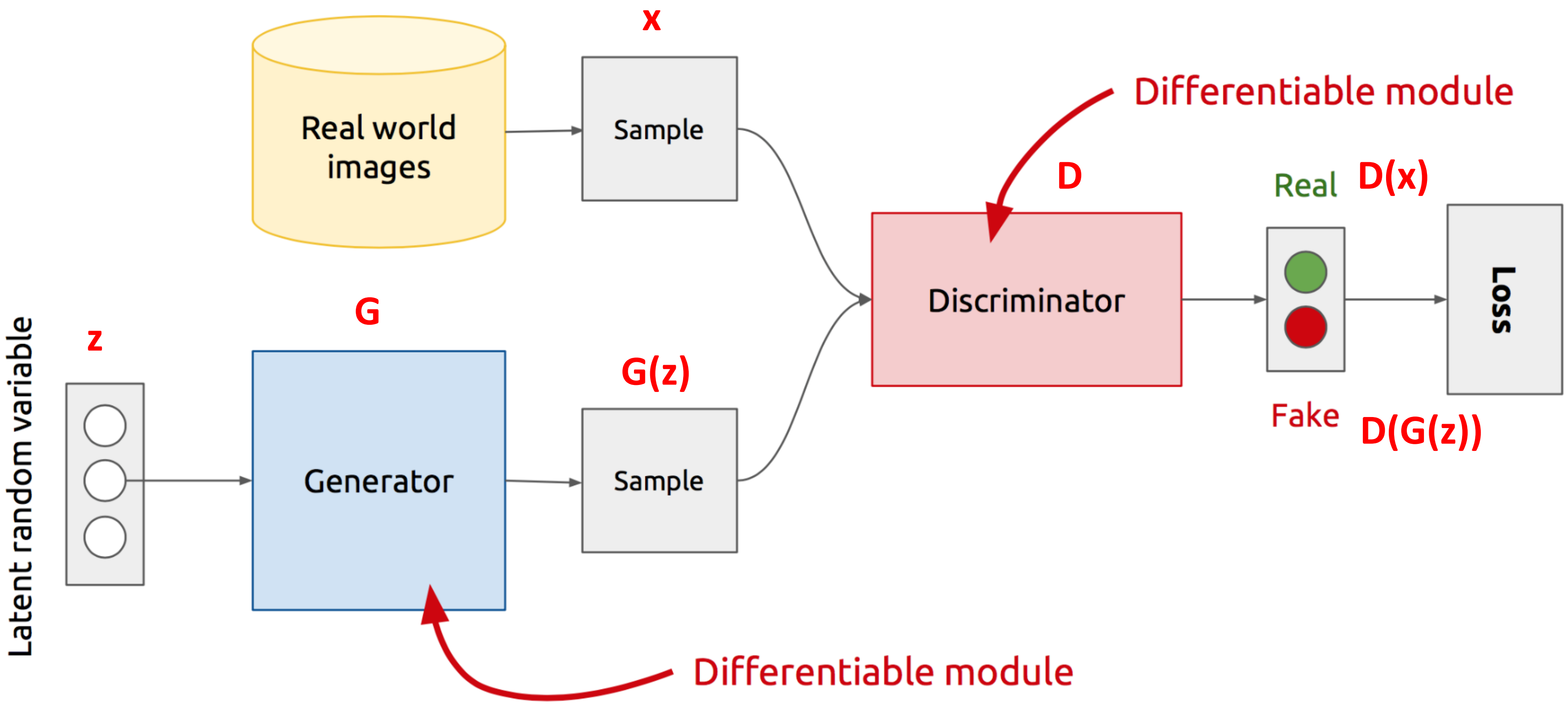
https://docs.pytorch.org/tutorials/advanced/neural_style_tutorial.html

2015





GAN's Architecture



Generator: generate fake samples, tries to fool the Discriminator
Discriminator: tries to distinguish between real and fake samples
Train them against each other
Repeat this and we get better Generator and Discriminator

- **Z** is some random noise (Gaussian/Uniform).
- **Z** can be thought as the latent representation of the image.

Training GANs: Two-player game

Ian Goodfellow et al., "Generative Adversarial Nets", NIPS 2014

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images

Train jointly in **minimax game**

$$\min_{G} \max_{D} V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

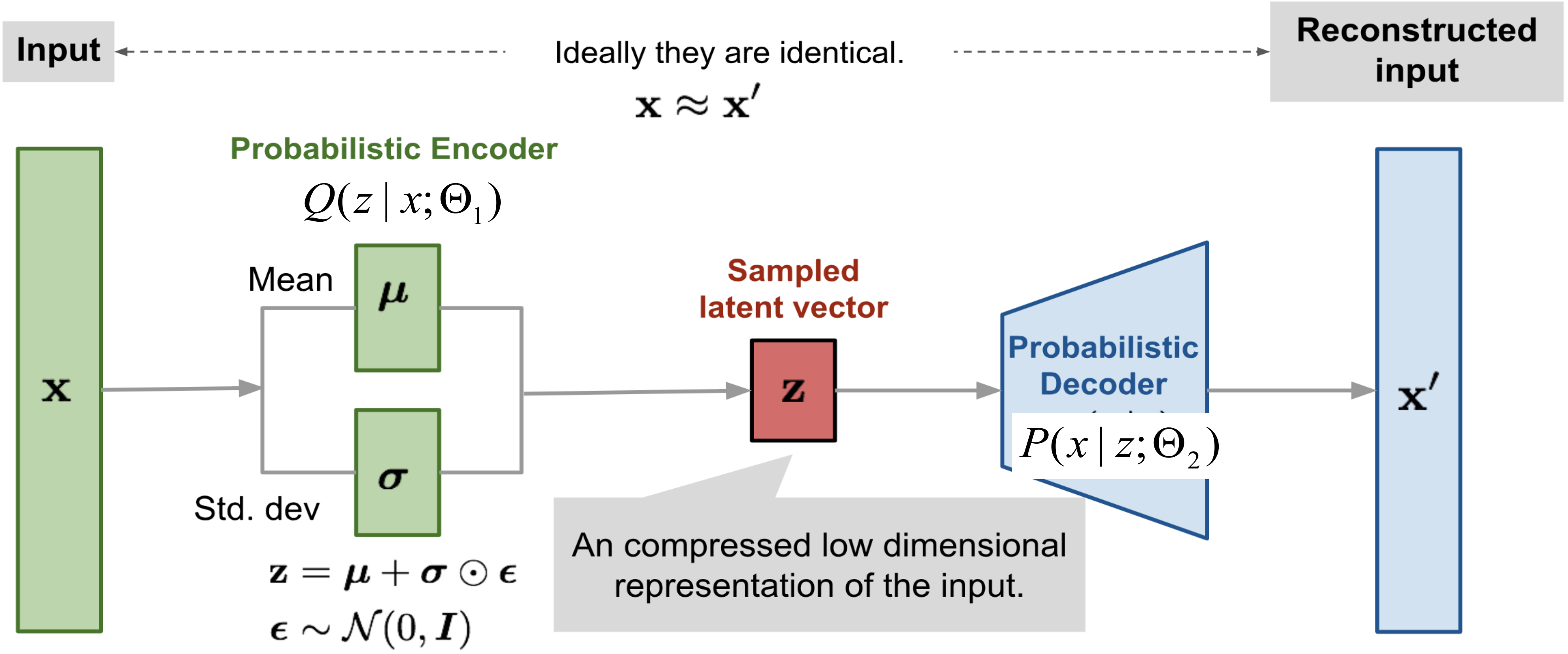
- Discriminator (θ_d) wants to **maximize objective** such that D(x) is close to 1 (real) and D(G(z)) is close to 0 (fake)
- Generator (θ_g) wants to **minimize objective** such that D(G(z)) is close to 1 (discriminator is fooled into thinking generated G(z) is real)

The Nash equilibrium of this particular game is achieved at:

$$P_{data}(x) = P_{gen}(x) \ \ \forall x$$

$$\mathrm{D}(x) = \frac{1}{2} \ \ \forall x$$

Variational autoencoder (VAE)



VAE objective function:

$$\log P(X) - D_{KL}[Q(z|X) \| P(z|X)] = \overbrace{E[\log P(X|z)] - D_{KL}[Q(z|X) \| P(z)]}^{\text{Evidence lower bound (ELBO)}}$$

VAE intuition

The easiest choice for P(z)is N(0,1). Hence, we want to make Q(z|X) to be as close as possible to N(0,1) so that we could sample it easily.

We also want Q(z|X) to be Gaussian with parameters μ(X) and Σ(X). The KL divergence between those two distribution could be computed in closed form!

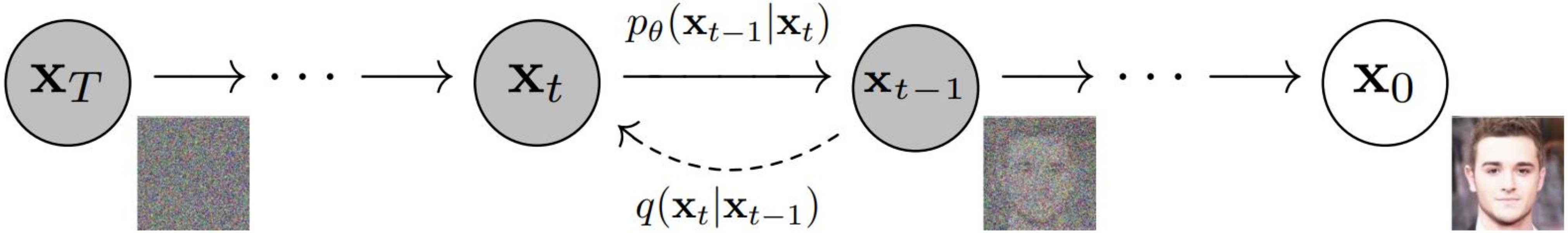
$$D_{KL}[N(\mu(X),\Sigma(X))\|N(0,1)] = \frac{1}{2}\sum_k \left(\Sigma(X) + \mu^2(X) - 1 - \log\Sigma(X)\right)$$

In practice, however, it's better to model $\Sigma(X)$ as $\log\Sigma(X)$, as it is more numerically stable to take exponent compared to computing log. Hence, our final KL divergence term is:

$$D_{KL}[N(\mu(X),\Sigma(X))\|N(0,1)] = \frac{1}{2}\sum_k \left(\exp(\Sigma(X)) + \mu^2(X) - 1 - \Sigma(X)\right)$$

https://wiseodd.github.io/techblog/2016/12/10/variational-autoencoder/

Denoising Diffusion Probabilistic Models



The forward process: $q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$

$$q(x_t|x_0) = \mathcal{N}(\sqrt{\bar{\alpha}_t}x_0, \sqrt{1-\bar{\alpha}_t}I) = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \qquad \alpha_t = 1-\beta_t \qquad \bar{\alpha}_t = \prod_{s=1}^{t} a_s$$

$$\epsilon \sim \mathcal{N}(0,1)$$

Algorithm 1 Training

```
repeat
  x_0 ~ q(x_0)
  t ~ Uniform({1, ..., T})
  ε ~ N(0, I)
  Take gradient descent step on
      ∇_θ ||ε − ε_θ(√ᾱ_t x_0 + √(1 − ᾱ_t) ε, t)||^2
until converged
```

$$\nabla_\theta \left\| \epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t) \right\|^2$$

Algorithm 2 Sampling

```
x_T ~ N(0, I)
for t = T, ..., 1 do
  z ~ N(0, I) if t > 1, else z = 0
  x_{t-1} = 1/√α_t (x_t − (1−α_t)/√(1−ᾱ_t) ε_θ(x_t, t)) + σ_t z
end for
return x_0
```

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t)\right) + \sigma_t\mathbf{z}$$