Technische Hochschule Rosenheim

Deep Learning

Unsupervised Learning – Generative Adversarial Networks (GANs)

Technische Hochschule Rosenheim
Sommer 2023
Prof. Dr. Jochen Schmidt

Acknowledgements

Technische Hochschule Rosenheim

Many of the slides presented here are based on the Deep Learning Slides Summer Semester 2020, courtesy of
A. Maier, V. Christlein, K. Breininger, F. Denzinger, F. Thamm,
Pattern Recognition Lab, Friedrich-Alexander-University Erlangen-Nürnberg.
https://lme.tf.fau.de/

Overview

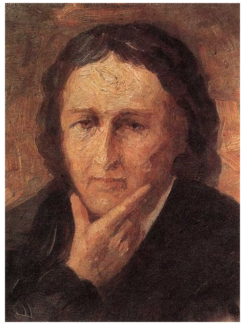
Technische Hochschule Rosenheim

- GAN basics
- Conditional GANs
- Deep Convolutional GANs

GAN Basics

Let’s Play a Game (or the Principle of GANs)

Discriminator



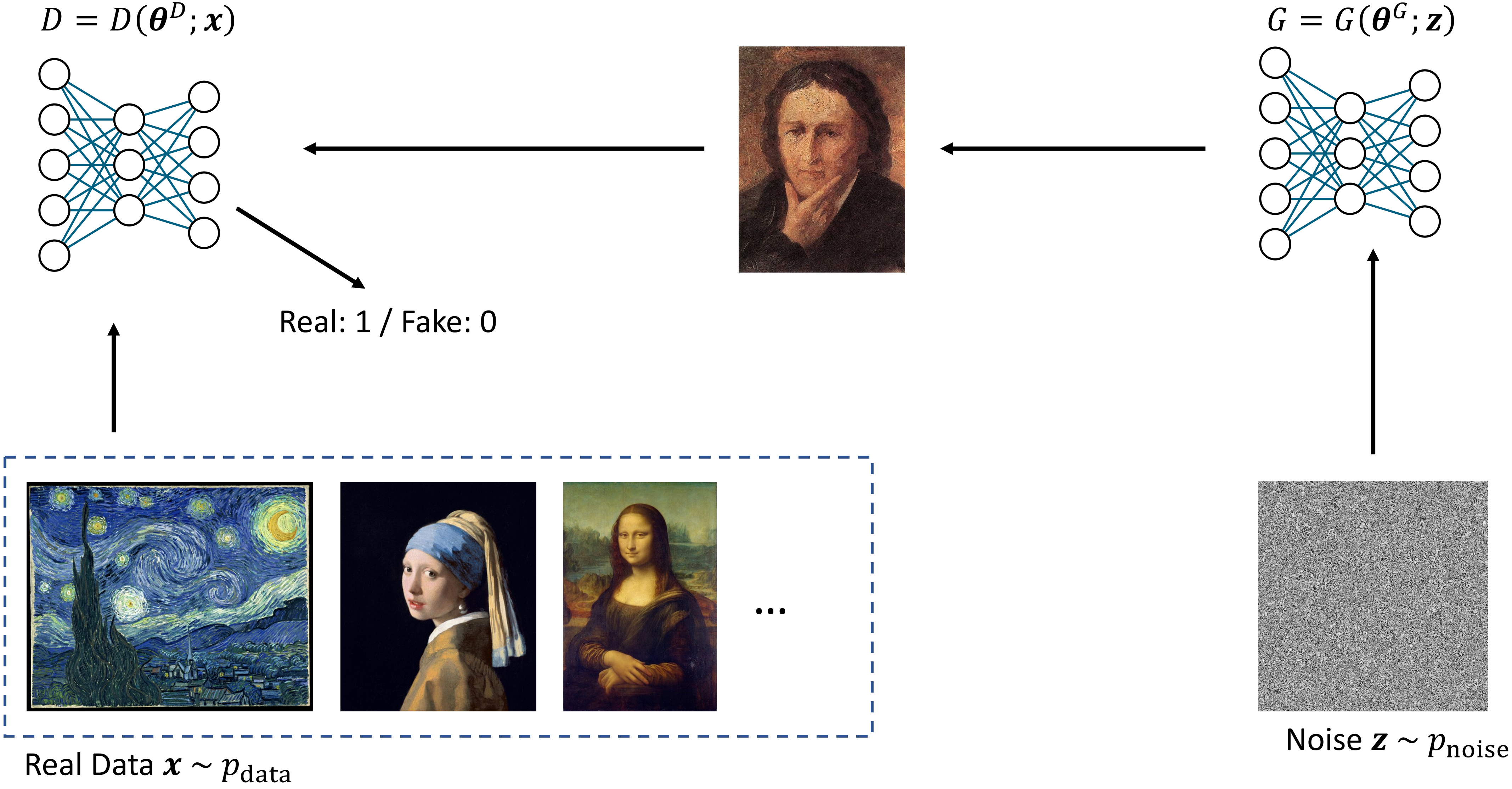
Generator

Real or Fake?

…

Real Data

Let’s Play a Game (or the Principle of GANs)

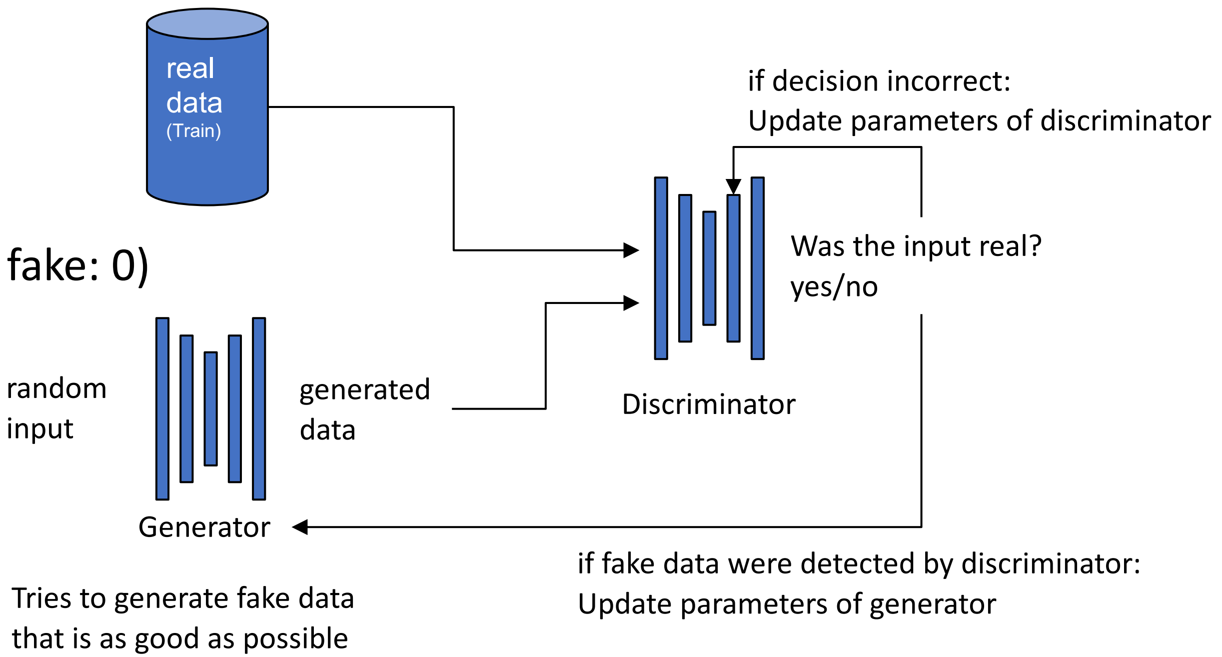


GANs – A Two Player Game (Minimax)

Technische Hochschule Rosenheim

- Objectives
  - **Discriminator** tries to distinguish real from fake data
  - **Generator** tries to generate data such that discriminator is fooled

- **Training**: Alternate between
  1. Training of **discriminator**
     - use real as well as fake data with labels (real: 1, fake: 0)
     - train using cross entropy
     - keep generator weights frozen
  2. Training of **generator**
     - update weights if fake was not good enough
     - keep discriminator weights frozen



Training GANs – Minimax

Alternate between

1. Train D: Maximize $$E_{\boldsymbol{x}\sim p_{\text{data}}} \log \overbrace{D(\boldsymbol{x})}^{\text{prop. real data}} + E_{\boldsymbol{z}\sim p_{\text{noise}}} \log \overbrace{\left(1 - D\left(G(\boldsymbol{z})\right)\right)}^{\text{prop. fake data}}$$

   $\longrightarrow$ Minimize $$L^D(\boldsymbol{\theta}^D, \boldsymbol{\theta}^G) = -E_{\boldsymbol{x}\sim p_{\text{data}}} \log D(\boldsymbol{x}) - E_{\boldsymbol{z}\sim p_{\text{noise}}} \log\left(1 - D\left(G(\boldsymbol{z})\right)\right)$$

   $\longrightarrow$ Trained to distinguish real data samples from fake ones

2. Train G: Maximize L^D $\longrightarrow$ Minimize $L^G = -L^D$

   Generator minimizes log-probability of the discriminator being correct
   $\longrightarrow$ Trained to generate data domain images and fool D

- Optional: run *k* steps of one player for every step of the other player
- Equilibrium is a saddle point of the discriminator loss

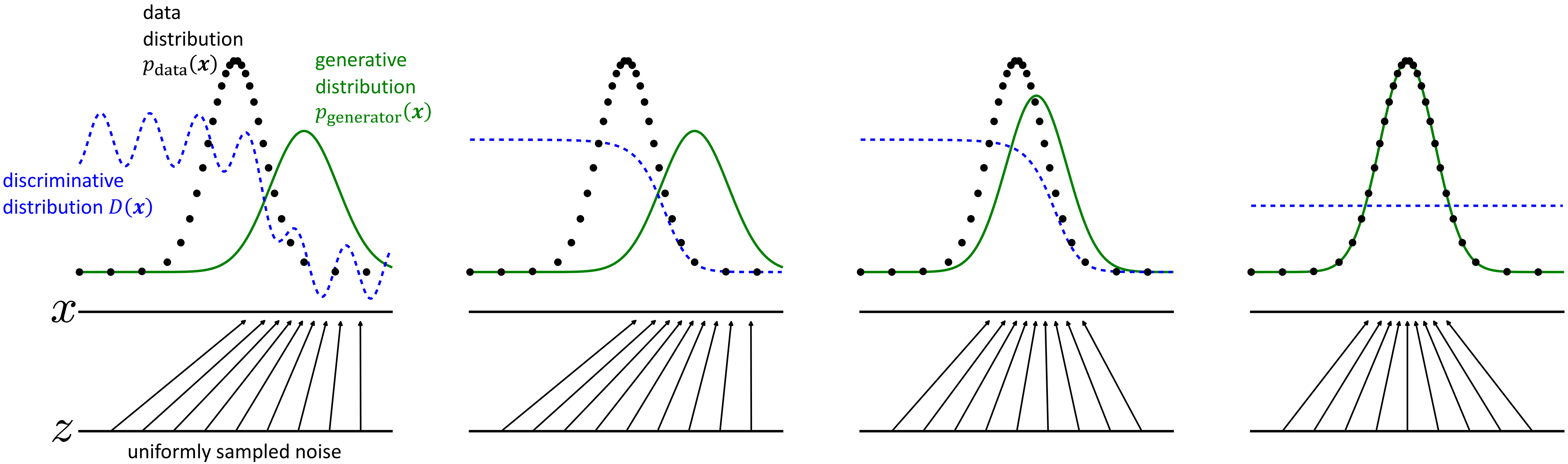
Training GANs – Optimal Discriminator

- Assumption: both densities are nonzero everywhere
  - Otherwise, some input values are never trained → some $D(\boldsymbol{x})$ have undetermined behavior
- For fixed generator, solve for $\frac{\partial L^D}{\partial D(\boldsymbol{x})} = 0$
- Optimal $D^*(\boldsymbol{x})$ for any $p_{\text{data}}(\boldsymbol{x})$ and $p_{\text{generator}}(\boldsymbol{x})$:

$$D^*(\boldsymbol{x}) = \frac{p_{\text{data}}(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x}) + p_{\text{generator}}(\boldsymbol{x})}$$

- The optimum is reached for $p_{\text{generator}} = p_{\text{data}}$, i.e., $D^*(\boldsymbol{x}) = \frac{1}{2}$
- GANs use supervised learning to estimate this ratio
  - Underfitting / Overfitting

Training GANs – Illustration



Source: [Goo14]

Non-Saturating Games – Modify Generator's Loss

Technische Hochschule Rosenheim

$$L^D = -E_{\boldsymbol{x}\sim p_{\text{data}}} \log \underbrace{D(\boldsymbol{x})}_{\text{prop. real data}} - E_{\boldsymbol{z}\sim p_{\text{noise}}} \log \Big(\underbrace{1 - D\big(G(\boldsymbol{z})\big)}_{\text{prop. fake data}}\Big)$$

(same as before)

$$L^G = -E_{\boldsymbol{z}\sim p_{\text{noise}}} \log D\big(G(\boldsymbol{z})\big)$$

(new)

- Objective: Improve generator when sample is recognized as being fake
- Particularly in the beginning, the discriminator will usually be better than the generator
  - leads to vanishing gradient of G for original loss (G minimizes log-probability of D being correct)
  - model cannot be trained, saturation
  - generator has lost the game before it really started
- Now: G maximizes log-probability of D being mistaken
  - Disadvantage: Equilibrium no longer describable with single loss

Other Losses: Feature Matching Loss / Perceptual Loss

- G trained to match expected value of features $f(\boldsymbol{x})$ of intermediate layer of D:

$$L^G = \left\| E_{\boldsymbol{x} \sim p_{\mathrm{data}}} f(\boldsymbol{x}) - E_{\boldsymbol{z} \sim p_{\mathrm{noise}}} f\big(G(\boldsymbol{z})\big) \right\|_2^2$$

- prevents “overtraining” of G on current D

- Many more loss functions exist, e.g.,
  - Wasserstein Loss (helps to counter vanishing gradients in D)
  - KL divergence
- But: the approximation strategy matters more than the loss

How to Evaluate GANs – Inception Score

Technische Hochschule Rosenheim

- Objectives:
  1. Generated images should be recognizable
     - feed image $\boldsymbol{x}$ through neural network for classification to obtain probabilities for labels y
       - standard: Inception v-3 pre-trained on Imagenet
       - results in conditional label distribution $p(y \mid \boldsymbol{x})$
     - meaningful images have a distribution where one class dominates, i.e.,
     - image-wise class distribution $p(y \mid \boldsymbol{x})$ should have low entropy
  2. Generated images should be diverse
     - Class distribution over all generated images $p(y) = \int p(y \mid \boldsymbol{x} = G(z))\, dz$ should be close to uniform, i.e.,
     - entropy should be high

- Based on KL divergence between distributions (higher = better):

$$\exp\left(E_x(KL(p(y \mid \boldsymbol{x}), p(y)))\right)$$

- see [Sal16] for details

How to Evaluate GANs – Fréchet Inception Distance (FID)

- Use intermediate layer (last pooling layer of Inception-v3 pre-trained on ImageNet)
- Model data distribution by multivariate Gaussians ($\boldsymbol{\mu}, \boldsymbol{\Sigma}$)
- FID score between real images $\boldsymbol{x}$ and generated images $\boldsymbol{g}$ (lower = better):

$$\left\|\boldsymbol{\mu}_x - \boldsymbol{\mu}_g\right\|_2^2 + \operatorname{tr}\left(\boldsymbol{\Sigma}_x + \boldsymbol{\Sigma}_g - 2\sqrt{\boldsymbol{\Sigma}_x \boldsymbol{\Sigma}_g}\right)$$

  - More robust to noise than Inception Score
  - No class concept needed

- see [Heu17] for details

GANs in Comparison to Other Generative Models

Technische Hochschule Rosenheim

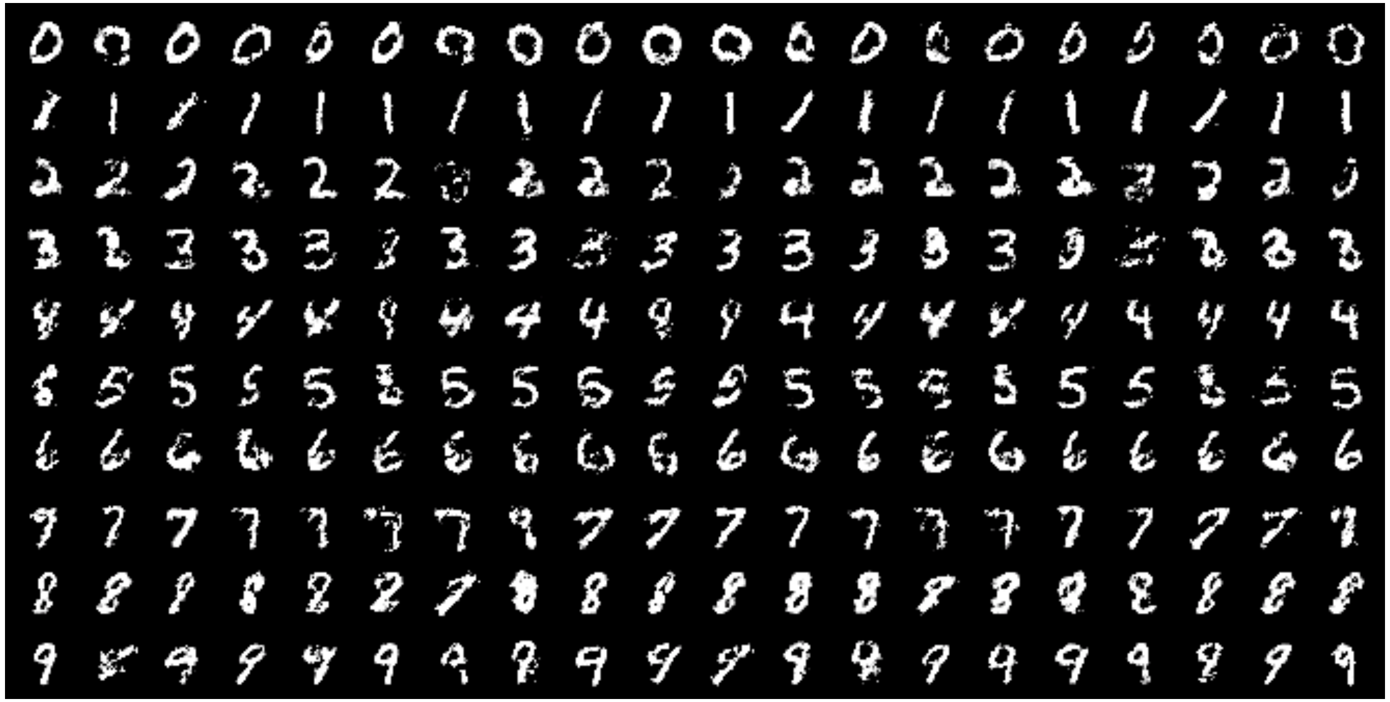
- Ability to generate samples in parallel
- Very few restrictions (e.g., compared to Boltzmann machines)
  - No Markov chain needed!
- No variational bound is needed
  - GANs known to be asymptotically consistent since the model families are universal function approximators

Conditional GANs (CGANs)

Conditional GANs – Idea

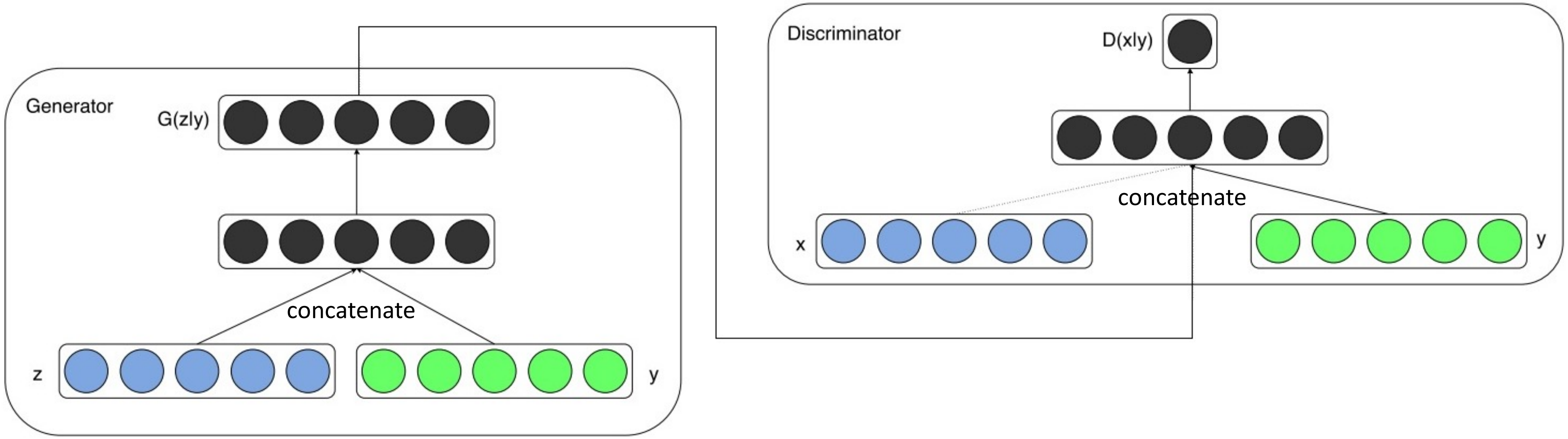
- Problem: Generator creates a “fake” generic image → not specific for a certain condition/characteristic
- Example: **text to image generation** – image should depend on the text
- Idea: Provide additional vector y to networks to encode **conditioning** [Mirza14]

Generated samples conditioned on one label (digit)



Conditional GANs – Overview

Source: [Mirza14]



- Generator G receives the noise vector $\boldsymbol{z}$ as well as a conditioning vector $\boldsymbol{y}$
- Discriminator D receives image $\boldsymbol{x}$ and also $\boldsymbol{y}$
- Loss functions change to

$$L^D = -E_{\boldsymbol{x}\sim p_{\text{data}}} \log D(\boldsymbol{x} \mid \boldsymbol{y}) - E_{\boldsymbol{z}\sim p_{\text{noise}}} \log\Big(1 - D\big(G(\boldsymbol{z} \mid \boldsymbol{y})\big)\Big)$$
$$L^G = -L^D$$

Example: Conditional GANs for Face Generation

- Add conditional feature (e.g., smiling, gender, old age, ...)
- Generator/Discriminator learn to operate in **modes**:
  - Generator learns to generate a face with a certain attribute
  - Discriminator learns to decide whether the face contains attribute

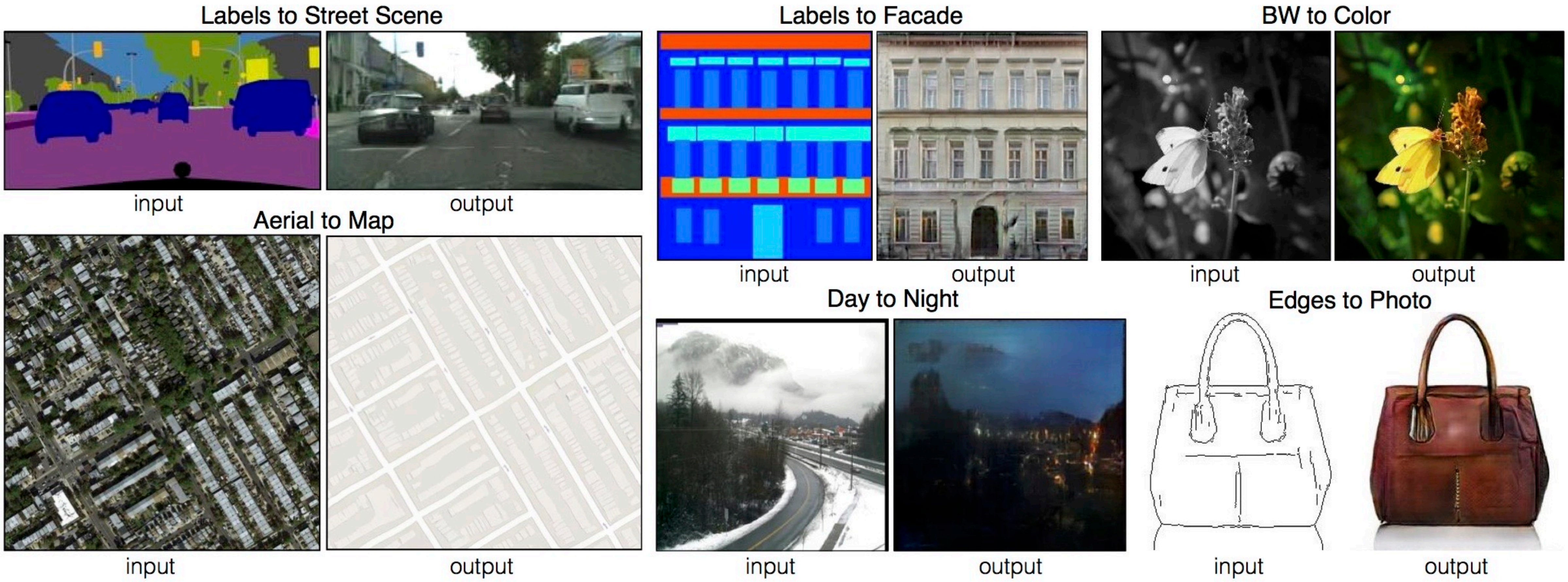
random samples

$\boldsymbol{y} \sim$ old age

$\boldsymbol{y} \sim$ old age + smiling

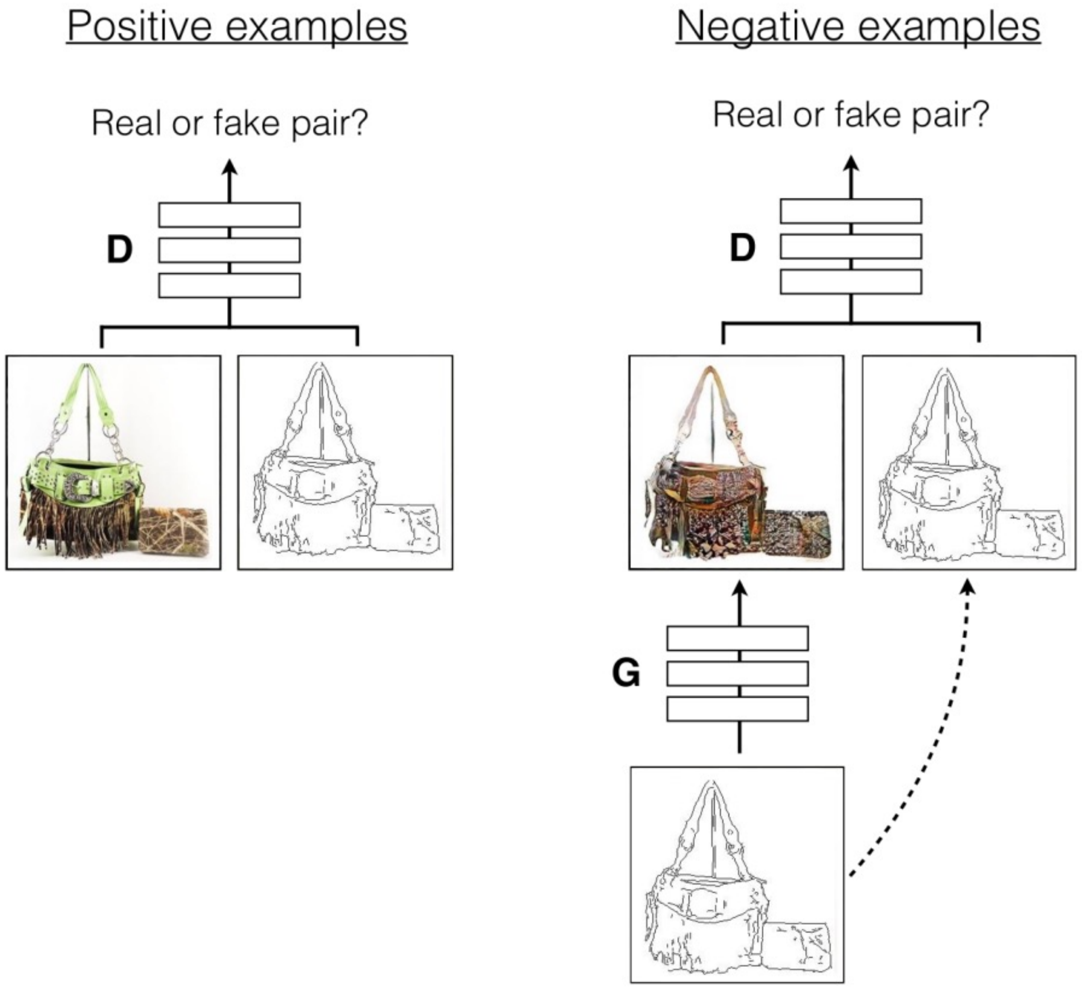
Source: [Gau15]

Image To Image Translation



Source: [Iso16]

Image To Image – Just a Conditional GAN!

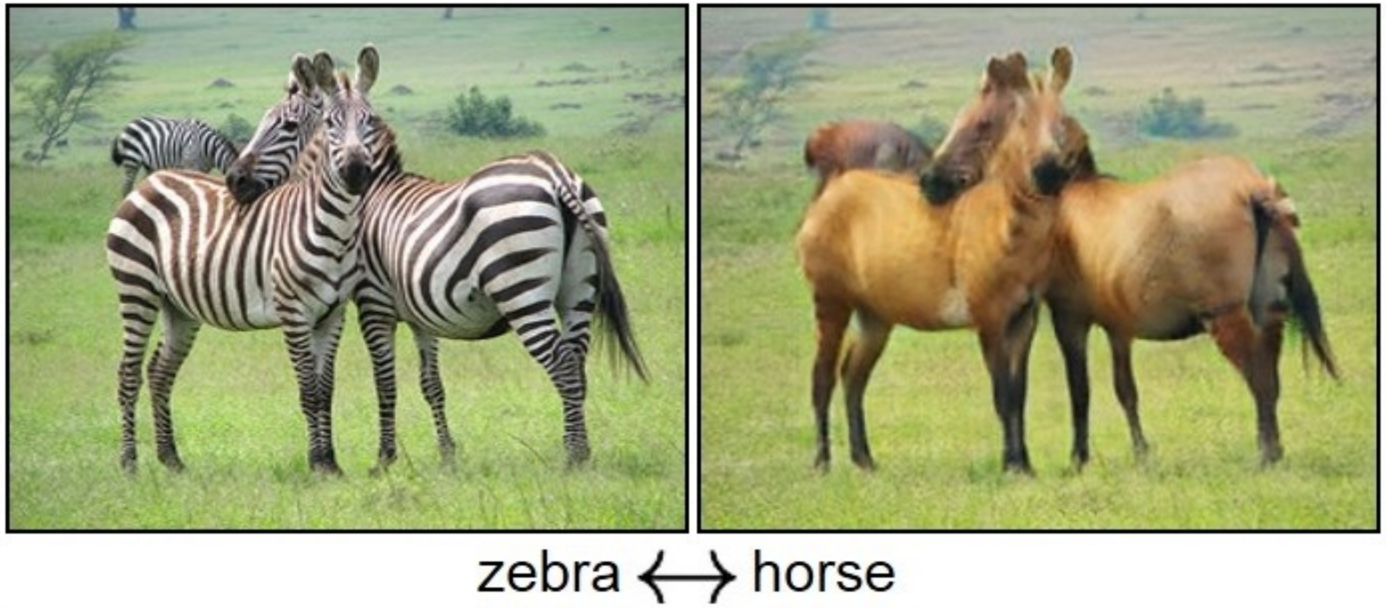


Source: [Iso16]

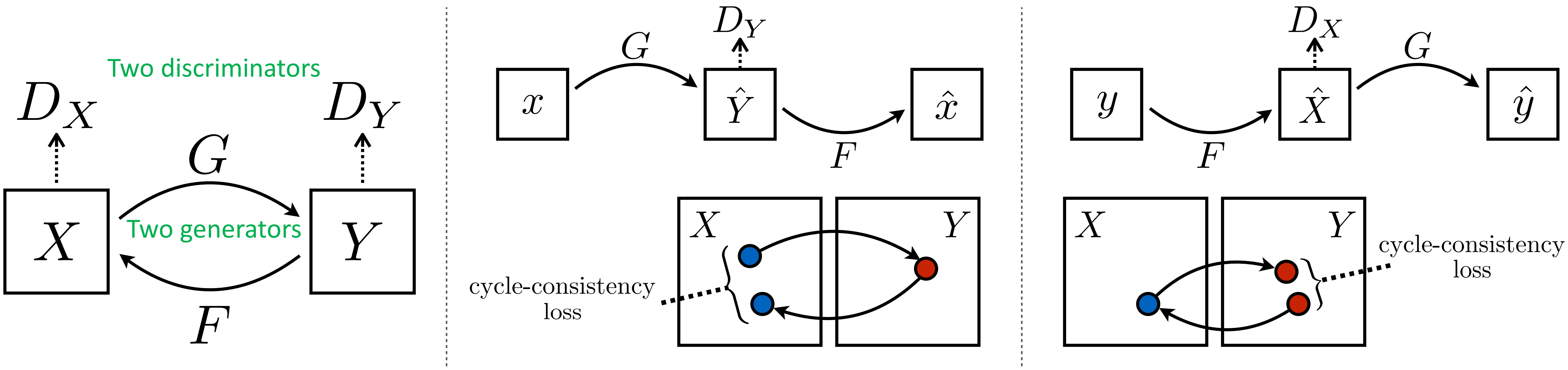
Cycle Consistent GANs

- Image to Image GAN should generate plausible results w.r.t. input
- **Paired data** difficult/impossible to obtain
- Cycle consistency loss [Zhu17]: Couple GAN with trainable **inverse** mapping F such that

$$F\big(G(\boldsymbol{x})\big) \approx \boldsymbol{x} \quad \text{and} \quad G\big(F(\boldsymbol{y})\big) \approx \boldsymbol{y}$$



Cycle Consistency Loss



- Two discriminators D_X and D_Y
- Cycle consistency loss for two generators G, F:

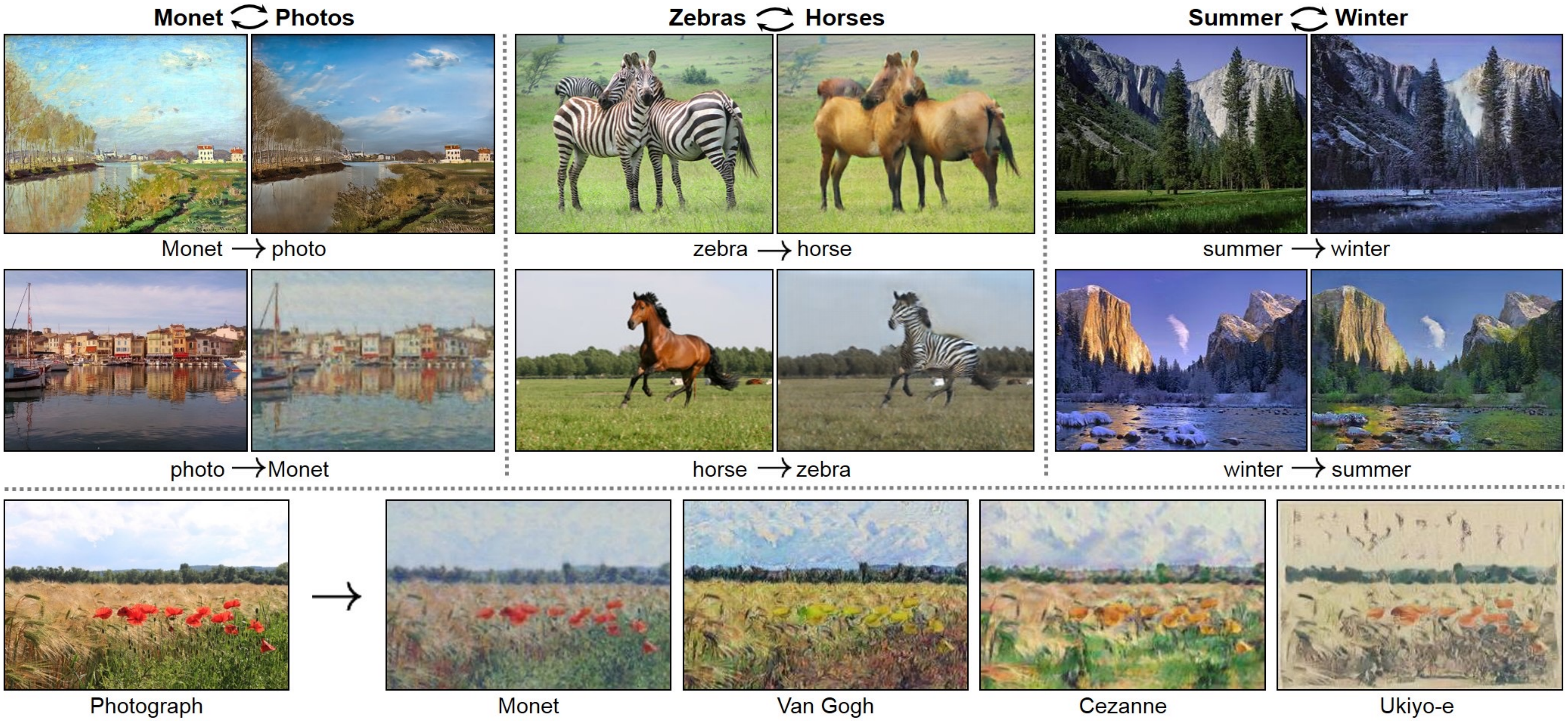
$$L_{\text{cyc}} = E_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}\left(\left\|F(G(\boldsymbol{x})) - \boldsymbol{x}\right\|_1\right) + E_{\boldsymbol{y} \sim p_{\text{data}}(\boldsymbol{y})}\left(\left\|F(G(\boldsymbol{y})) - \boldsymbol{y}\right\|_1\right)$$

- Total loss:

$$L = L_{\text{GAN}}(G, D_X, X, Y) + L_{\text{GAN}}(F, D_Y, X, Y) + \lambda L_{\text{cyc}}(G, F)$$

adapted from [Zhu17]

CycleGAN – Examples



Source: [Zhu17]

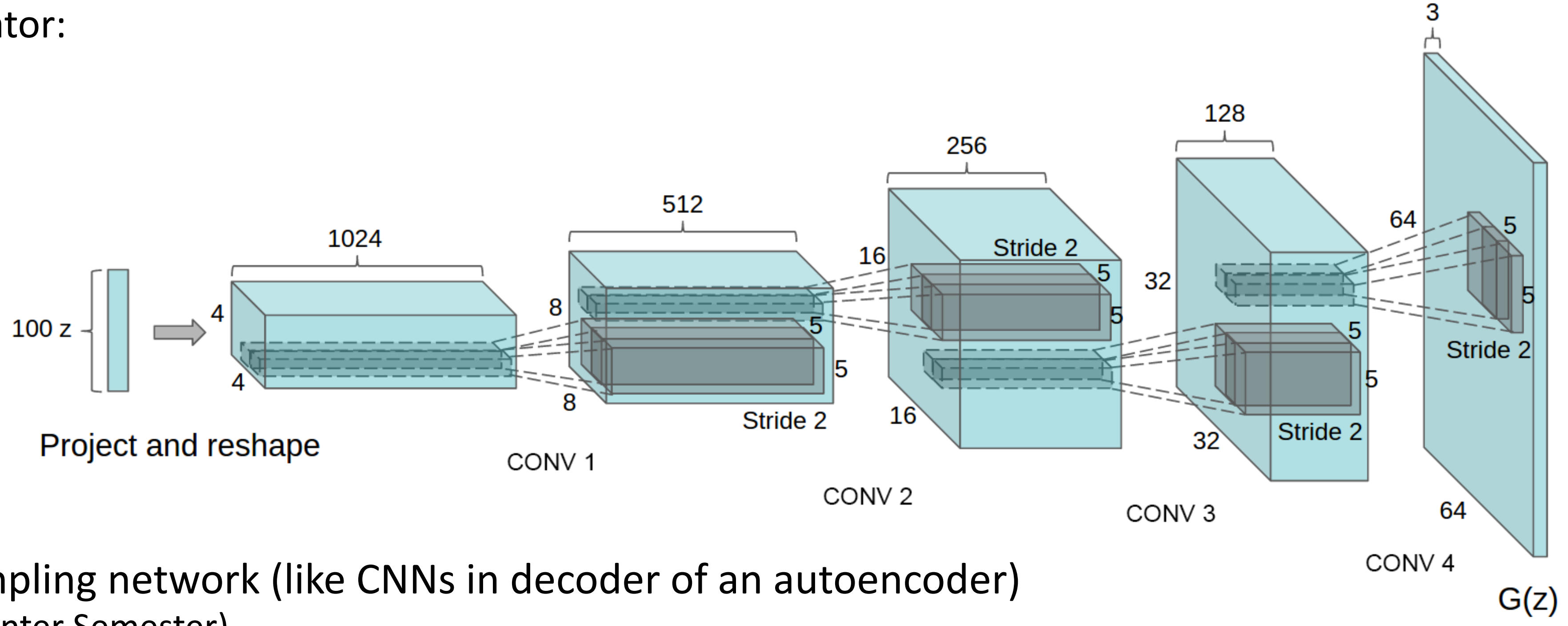
CycleGAN – Examples

Source: [Zhu17]

Deep Convolutional GANs (DCGAN)

Deep Convolutional GANs (DCGAN)

Generator:



- Discriminator is a CNN
- Generator is an upsampling network (like CNNs in decoder of an autoencoder)
  → Computer Vision (Winter Semester)
- Architecture Guidelines
  - Replace any pooling layer with strided convolutions (D) and transposed convolution (G)
  - Remove fully connected hidden layers for deeper architectures
  - G: Use ReLU activation except for output layer which uses tanh
  - D: Leaky ReLU activation for all layers
  - Use batch normalization

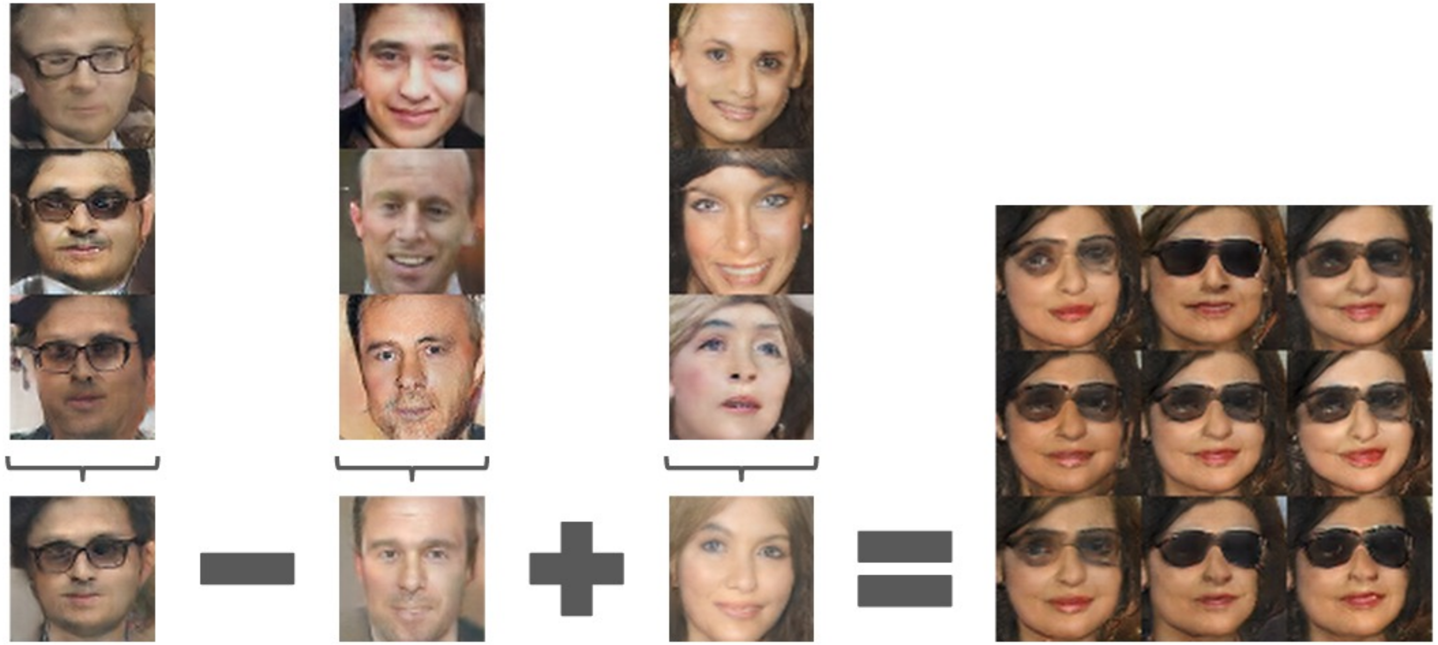
Source: [Rad15]

DCGAN – Examples

Bedrooms after 1 epoch

Source: [Rad15]

Vector Arithmetic



man with glasses | man without glasses | woman without glasses | woman with glasses

- Average three latent z codes and apply operation
  - GANs learn a distributed representation that disentangles the concept of gender from the concept of wearing glasses
- See also “InfoGAN” [Chen16]

Remarks

One-sided Label Smoothing

- Replace targets of the real samples with a smoothed version → replace 1 by 0.9
- Do **not** do the same for fake samples (don’t change 0 label)
  - Otherwise, D will reinforce incorrect behavior
  - G will produce samples that resemble the data or samples it already makes
- Benefits
  - Prevents D from giving very large gradient signal to G
  - Prevents extrapolating to encourage extreme samples

Is Balancing G and D necessary?

No.

- GANs work by estimating ratio of data and model density
  - Ratio estimated correctly only when D is optimal
  - Fine if D overpowers G

But when D gets too good

- G's gradient may vanish → use non-saturating loss
- G's gradient may get too large → use label smoothing

Summary

GANs

- are generative models that use supervised learning to approximate an intractable cost function
- can simulate many cost functions
- hard to find equilibrium between D and G
- cannot generate discrete data
- can also be used for
  - (semi-)supervised classification
  - transfer learning
  - multi-modal outputs
  - ...

- there are many more models out there, e.g.,
  - StackedGANs: Given some text, generate a fitting image [Zha16]
  - Style-Based Generator Architecture (StyleGAN) [Kar18]
    - check out https://www.whichfaceisreal.com/

References

[Boc20] T. Bocklet: Deep Learning Slides Winter Semester 2020/21. Technische Hochschule Nürnberg.

[Chen16] Xi Chen, Xi Chen, Yan Duan, et al. “InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets”. In: Advances in Neural Information Processing Systems 29. Curran Associates, Inc., 2016, pp. 2172–2180.

[Gau15] John Gauthier. Conditional generative adversarial networks for face generation. Mar. 17, 2015. URL: http://www.foldl.me/2015/conditional-gans-face-generation/ (visited on 21/06/2021).

[Goo14] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, Yoshua Bengio. „Generative Adversarial Nets“. *Annual Conference on Neural Information Processing Systems* 2014. Pages 2672-2680. https://arxiv.org/abs/1406.2661

[Goo16] Ian Goodfellow. NIPS 2016 Tutorial: Generative Adversarial Networks. 2016. eprint: arXiv:1701.00160.

[Heu17] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, et al. “GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium”. In: Advances in Neural Information Processing Systems 30. Curran Associates, Inc., 2017, pp. 6626–6637.

[Iso16] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, et al. “Image-to-Image Translation with Conditional Adversarial Networks”. In: (2016). eprint: arXiv:1611.07004.

[Kar18] Tero Karras, Samuli Laine, Timo Aila. “A Style-Based Generator Architecture for Generative Adversarial Networks. In: arXiv:1812.04948. 2018.

[Kingma13] Diederik P Kingma and Max Welling. “Auto-Encoding Variational Bayes”. In: arXiv e-prints, arXiv:1312.6114 (Dec. 2013), arXiv:1312.6114.

[Mirza14] Mehdi Mirza and Simon Osindero. “Conditional Generative Adversarial Nets”. In: CoRR abs/1411.1784 (2014). arXiv: 1411.1784.

[Ng11] Andrew Ng. “CS294A Lecture notes”. In: 2011.

[Rad15] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks, 2015. eprint: arXiv:1511.06434.

[Sal16] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, et al. “Improved Techniques for Training GANs”. In: Advances in Neural Information Processing Systems 29. Curran Associates, Inc., 2016, pp. 2234–2242.

[Zha16] Han Zhang, Tao Xu, Hongsheng Li, et al. “StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks”. In: CoRR abs/1612.03242 (2016). arXiv: 1612.03242.

[Zhou16] Bolei Zhou, Aditya Khosla, Agata Lapedriza, et al. “Learning Deep Features for Discriminative Localization”. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, June 2016, pp. 2921–2929. arXiv: 1512.04150.

[Zhu17] Jun-Yan Zhu. “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks”. In: CoRR abs/1703.10593 (2017). arXiv: 1703.10593.