Nizhny Novgorod State University

Institute of Information Technologies, Mathematics and Mechanics

Department of Computer software and supercomputer technologies

Educational course
«Modern methods and technologies
of deep learning in computer vision»

Preparing synthetic data based on generative adversarial networks

Supported by Intel

Kustikova Valentina

Content

- ❑ Goals
- ❑ The concept of a generative model
- ❑ The difference between discriminative and generative models
- ❑ Generative adversarial networks
  - – Model scheme
  - – Training problem statement
  - – Training algorithm
- ❑ Classification of generative adversarial networks
- ❑ Applications of generative adversarial networks
- ❑ Conclusion

Goals

- ***The goal*** is to study the general scheme of constructing generative adversarial networks and the algorithm of their training, to consider the classification of generative adversarial networks and their applications

THE CONCEPT OF A GENERATIVE MODEL

Generative model

- ***Generative model*** describes general rules for generating a dataset in terms of a probabilistic model. Sampling data from the constructed probabilistic model allows to generate new data

Generative modelling scheme (1)

Training

Sampling

Train dataset → **Generative model** → **Generated sample**

Random noise ↑

Synthetic vehicle image

Feature is a pixel of the observation

Observation

Generative modelling scheme (2)

- It is assumed there is a dataset containing examples of entities that should to be generated – ***training dataset***
- ***Observation*** is a single element of the training dataset
- Generating images, each observation consists of ***features***; usually, feature is a pixel intensity
- The model is trained to generate ***data*** (images) according to the same rules that the training dataset is constructed
- Generating samples, each pixel is assigned to some intensity value

Generative modelling scheme (3)

- Generative model has to be a probabilistic one
- If the model during sampling data gives the mean value of the training samples as the pixel intensity, then the model is not generative, since it always provides the same result
- The model should include an element of randomness reflecting the individual characteristics of the generated example
- Therefore, ***a probability distribution that explains why some observations (images) are similar to each other in the training dataset, while others are not, has to be found***
- ***The goal of generative modelling is to construct the model that simulates this distribution in the best way***

THE DIFFERENCE BETWEEN DISCRIMINATIVE AND GENERATIVE MODELS

Discriminative modelling scheme (1)

Train dataset → Training → **Discriminative model** → Sampling → **Generated sample**


1

0

0.93 – confidence
of the label 1


1

0

Observation


1

0

Observation label
(1 – «car»,
0 – «not car»)

Discriminative modelling scheme (2)

- It is assumed there is a dataset containing examples of entities – ***training dataset***
- ***Observation*** is a single element of the training dataset
- There is a ***label*** for each observation
- ***The goal of discriminative modelling is to construct a function based on the training dataset that matches observations with labels in the best way***

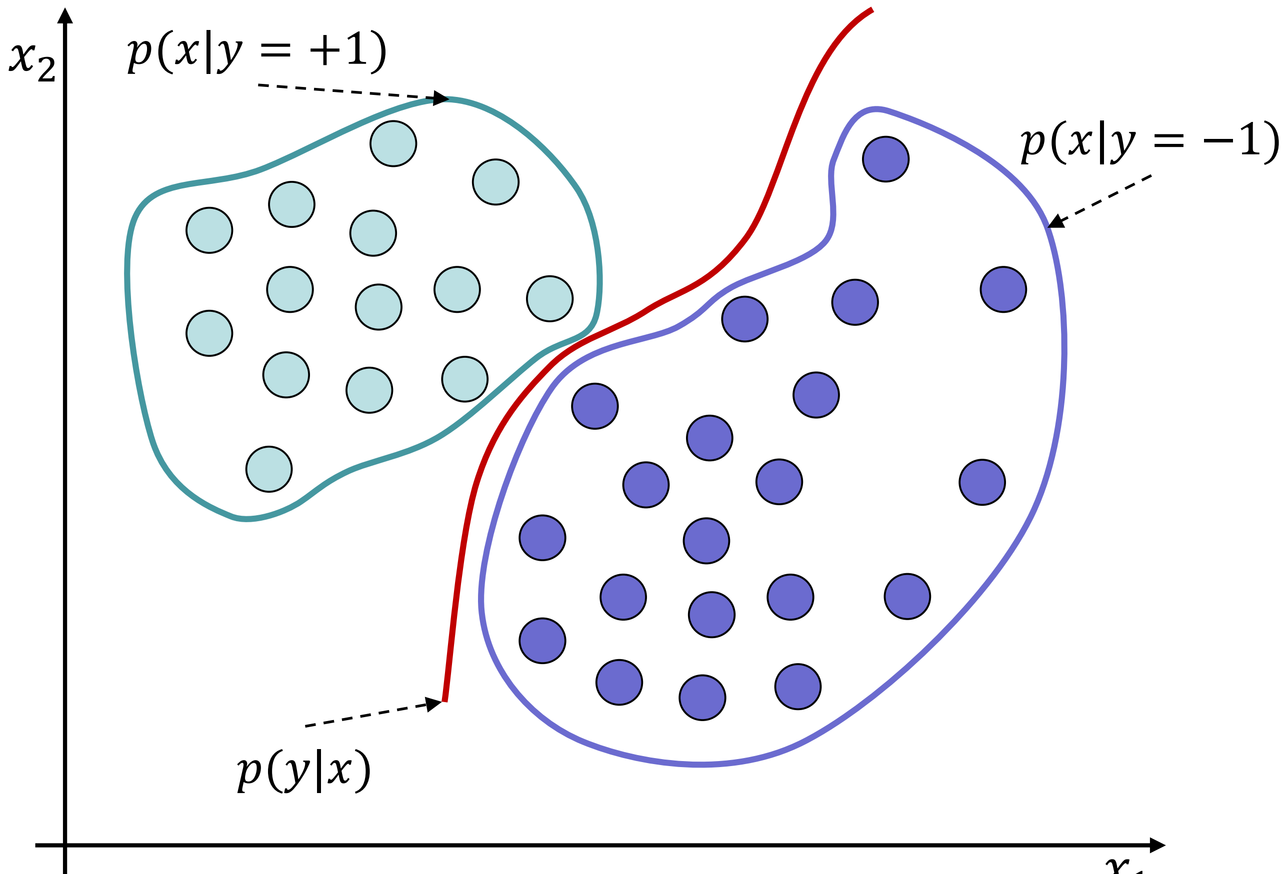
The difference between discriminative and generative modeling (1)

- ***Different modeling goals***
  - ***Generative models*** estimate $p(x, y)$, $p(x, y)$ is a joint probability distribution (it evaluates boundaries of classes)
    - If the dataset is not annotated (unsupervised learning), then model estimates $p(x)$, $p(x)$ is a probability of the observation x
    - If the training dataset is annotated, then the generative model is able to estimate the conditional probability $p(x|y)$ (calculated using the Bayes' theorem), $p(x|y)$ is a probability of the observation x if x belongs to the class y
  - ***Discriminative models*** estimate $p(y|x)$, $p(y|x)$ is a confidence of label y if there is an observation x at the input (it evaluates boundary between classes)

* Ng A.Y., Jordan M.I. On Discriminative vs. Generative Classifiers: A comparison of logistic regression and naïve Bayes // Advances in Neural Information Processing Systems. – 2002. – [http://ai.stanford.edu/~ang/papers/nips01-discriminativegenerative.pdf].

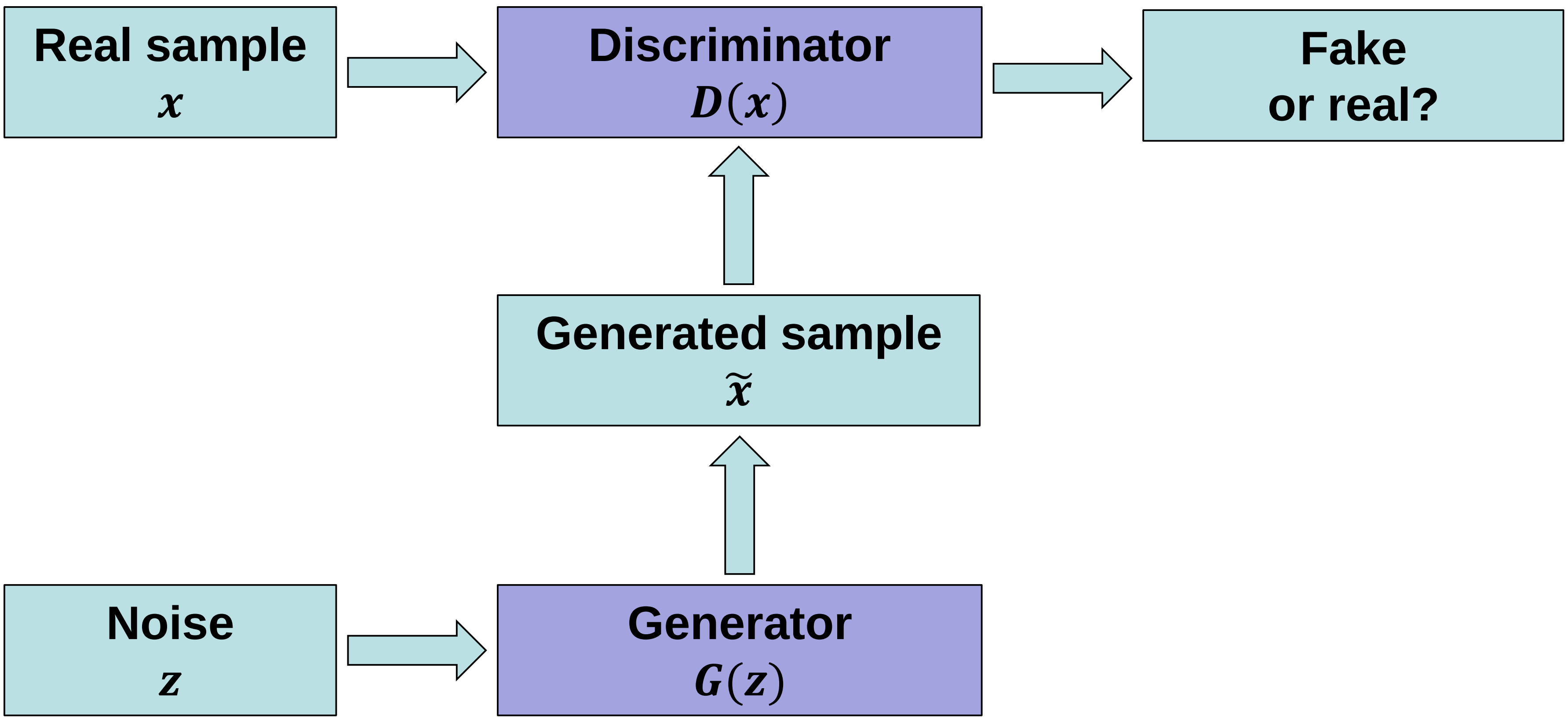
The difference between discriminative and generative modeling (2)

- Interpretation of the difference using the example of two-dimensional observations and two object classes:



GENERATIVE ADVERSARIAL NETWORKS

Model scheme (1)



Model scheme (2)

- Generative adversarial network (GAN) consists of two neural networks:
  - ***Generator*** is a network which generates samples. The goal of the generator is to study to “fool” a discriminator
  - ***Discriminator*** is a network which tries to distinguish real observations from generated samples. The goal of the discriminator is to study recognizing “lie” in the best way
- An example of generating human faces:
  - The generator input is a random noise, the generator output (the discriminator input) is a generated RGB image of a human face
  - The discriminator output is a confidence (from 0 to 1) that the RGB image is an image of a real face

Training problem statement (1)

- Notation:
  - X is a set of observations sampled from the distribution p_{data}
  - Z is a latent space sampled from the distribution p_Z (for example, a set of random vectors sampled from the uniform distribution in the interval $[0; 1]$)
- ***Generator***
  - Mapping $G: Z \to \mathbb{R}^n$, θ is a set of generator parameters
  - The goal is to generate a sample that is maximally similar to the observations from the distribution p_{data}
- ***Discriminator***
  - Mapping $D: \mathbb{R}^n \to [0; 1]$, γ is a set of discriminator parameters
  - The goal is to predict the maximum confidence on the observations of X and the minimum one on the generated samples

Training problem statement (2)

- Notation:
  - θ and γ are parameters of the neural networks corresponding to the generator and discriminator respectively
  - p_{gen} is a distribution of samples, produced by the generator

- ***The training problem*** is to construct the distribution p_{gen}, which simulates the distribution p_{data} in the best way

Training problem statement (3)

- ***The loss function*** is as follows:

$$E_{x\sim p_{data}}[\log D(x)] + E_{\tilde{x}\sim p_{gen}}\left[\log\left(1 - D(\tilde{x})\right)\right],$$

where $E_{\tilde{x}\sim p_{gen}}\left[\log\left(1 - D(\tilde{x})\right)\right] = E_{z\sim p_Z}\left[\log\left(1 - D\left(G(z)\right)\right)\right]$

- ***The training problem*** for a generative adversarial network is the following optimization problem (minimax game)

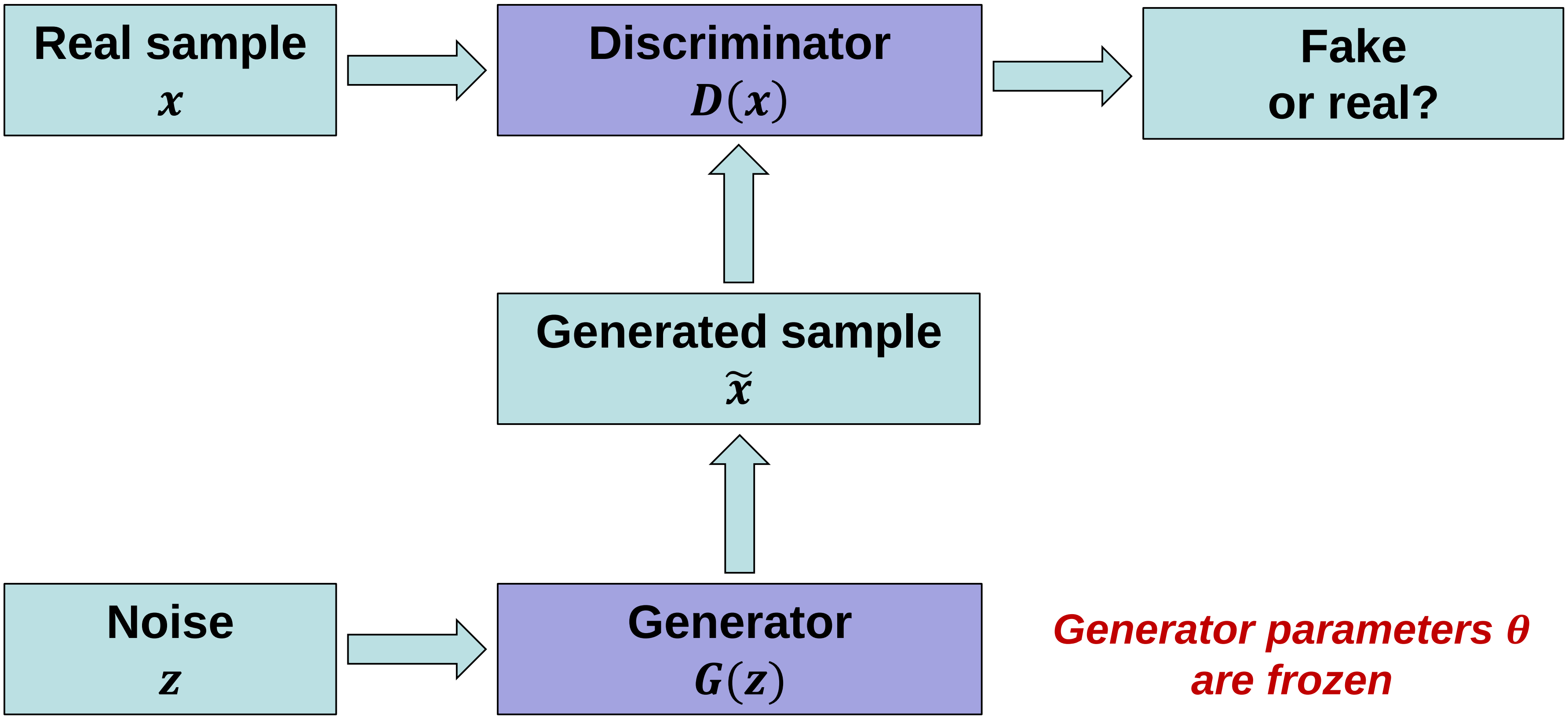
$$\min_G \max_D E_{x\sim p_{data}}[\log D(x)] + E_{z\sim p_Z}\left[\log\left(1 - D\left(G(z)\right)\right)\right]$$

- ***Note***: the proof of converging the distribution p_{gen} to p_{data} is described in the original paper*, in which GANs have been proposed

* Goodfellow I.J., et al. Generative Adversarial Nets. – 2014. – [https://arxiv.org/pdf/1406.2661.pdf].

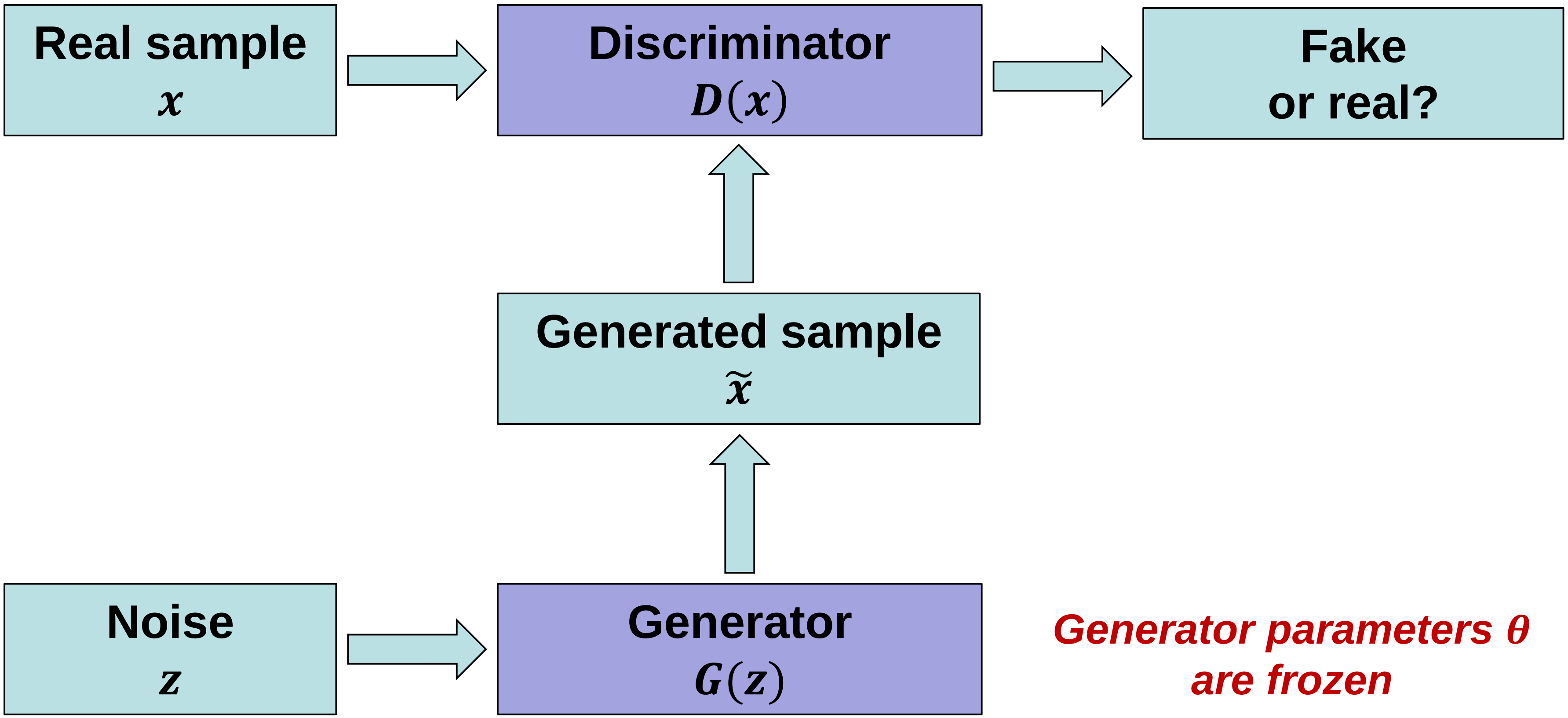
Training algorithm. Stage 1 (1)

1. Sample $\{x_1, x_2, \dots, x_m\}$
from the distribution p_{data}



Training algorithm. Stage 1 (2)

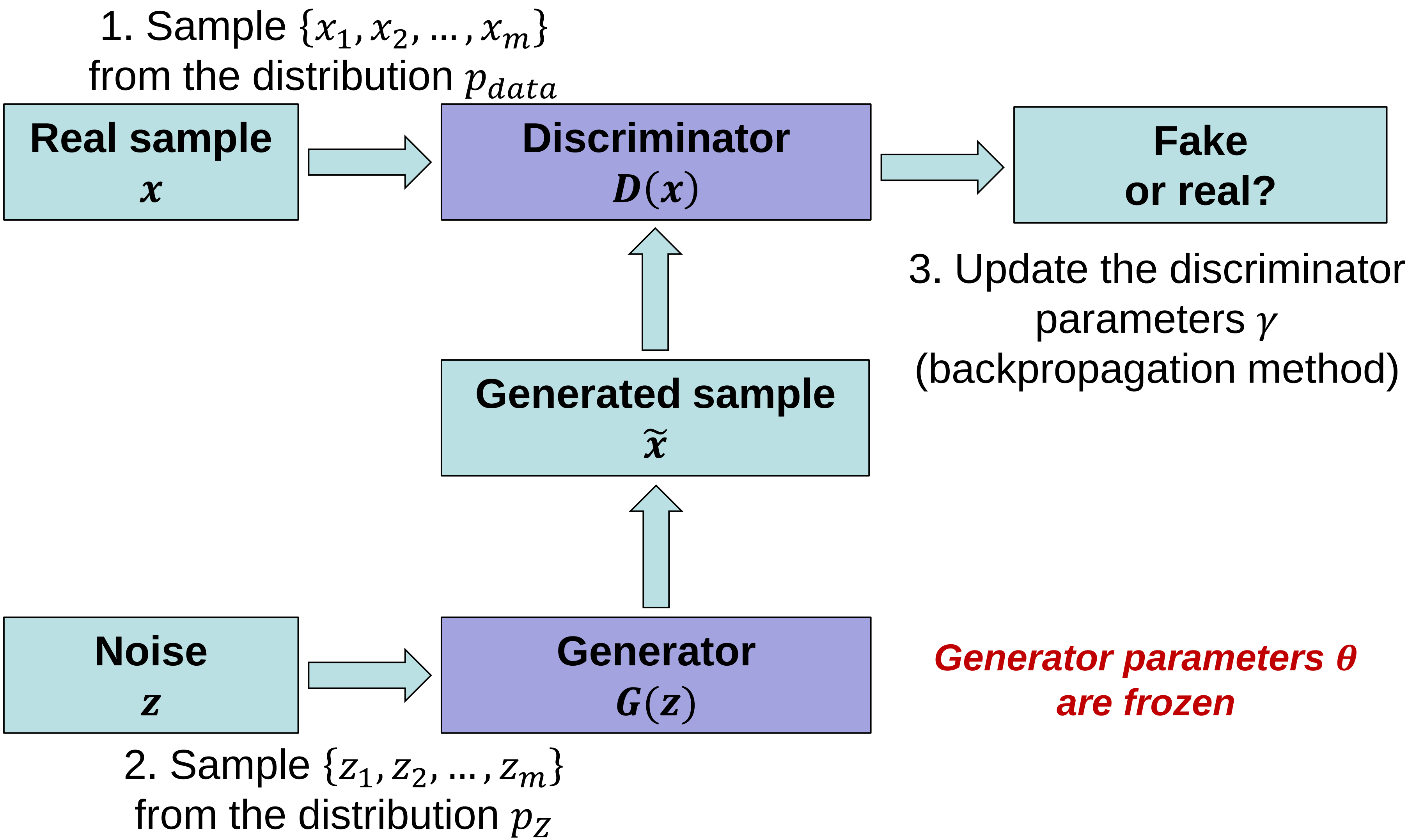
1. Sample $\{x_1, x_2, \ldots, x_m\}$
from the distribution p_{data}



Generator parameters θ are frozen

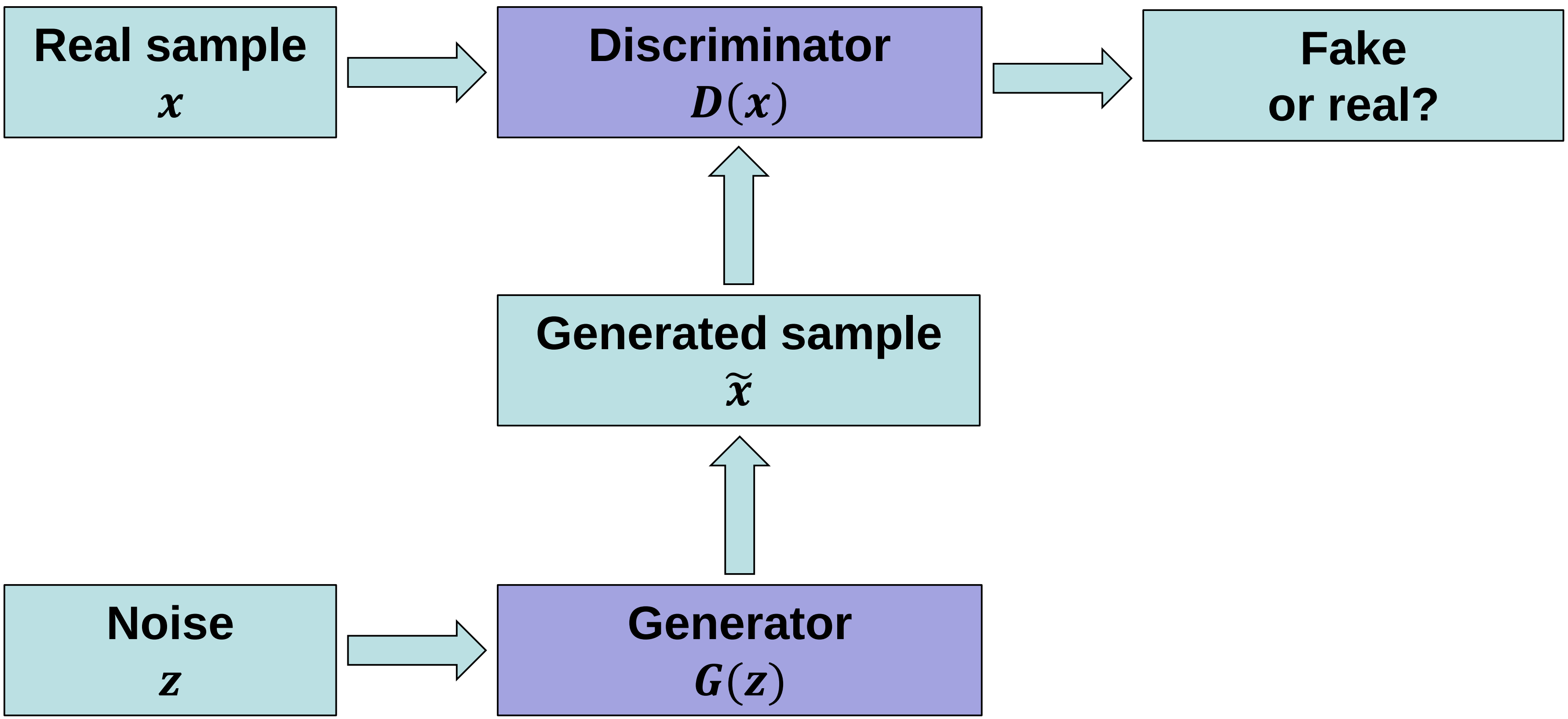
2. Sample $\{z_1, z_2, \ldots, z_m\}$
from the distribution p_Z

Training algorithm. Stage 1 (3)



Training algorithm. Stage 2 (1)

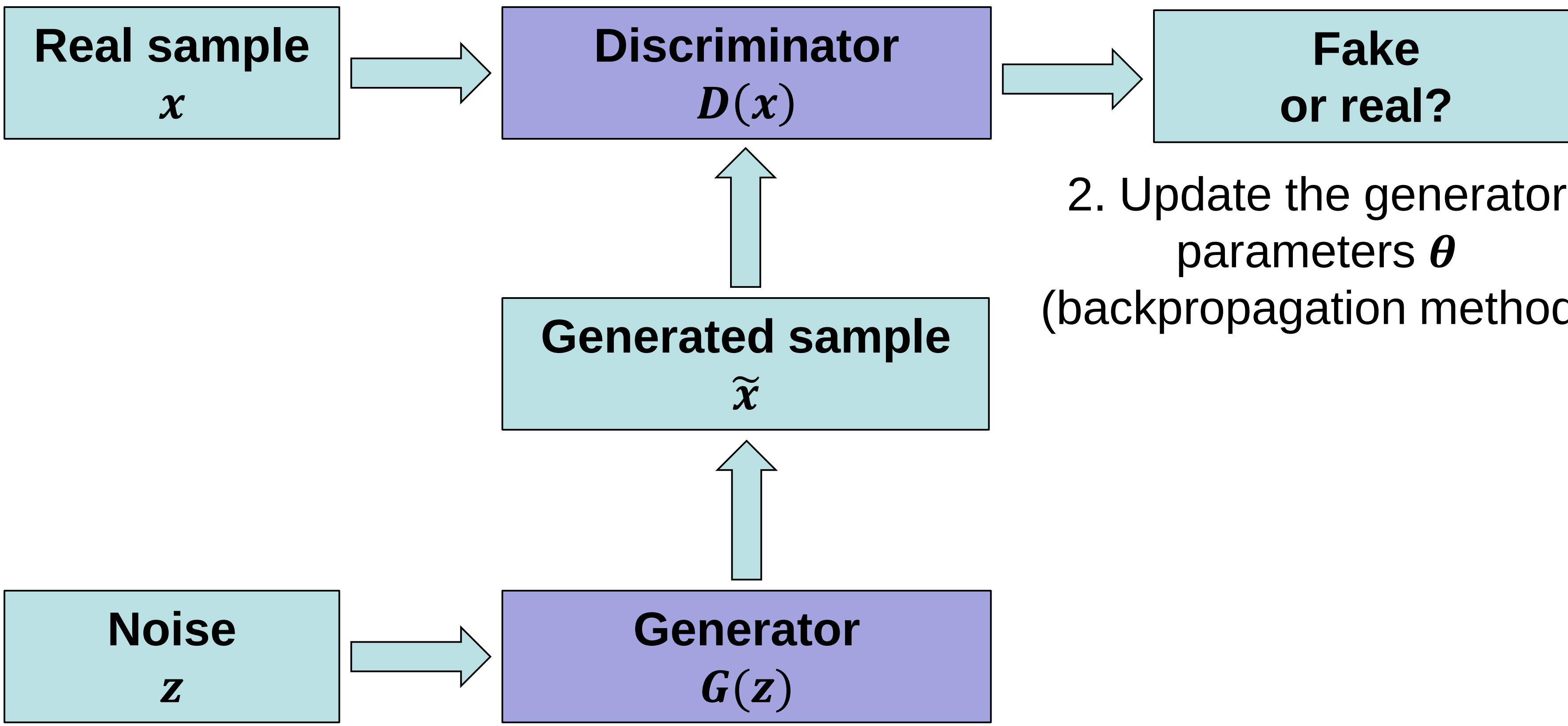
Discriminator parameters γ are frozen



1. Sample $\{z_1, z_2, \ldots, z_m\}$ from the distribution p_Z

Training algorithm. Stage 2 (2)

Discriminator parameters $\boldsymbol{\gamma}$
are frozen



2. Update the generator parameters $\boldsymbol{\theta}$ (backpropagation method)

1. Sample $\{z_1, z_2, \dots, z_m\}$ from the distribution p_Z

Training algorithm. Pseudocode

for i = 1..num_iteration do

  for j = 1..k do

    1.1. Sample $\{x_1, \dots, x_m\}$ from the distribution p_{data}

    1.2. Sample $\{z_1, \dots, z_m\}$ from the distribution p_Z

    1.3. Update the discriminator parameters γ

$$\Delta\gamma \leftarrow \nabla_\gamma \frac{1}{m}\sum_{t=1}^{m}[\log D(x_t)] + \left[\log\left(1 - D\left(G(z_t)\right)\right)\right]$$

  end for

  2.1. Sample $\{z_1, \dots, z_m\}$ from the distribution p_Z

  2.2. Update the generator parameters θ

$$\Delta\theta \leftarrow \nabla_\theta \frac{1}{m}\sum_{t=1}^{m}\left[\log\left(1 - D\left(G(z_t)\right)\right)\right]$$

end for

Training algorithm. Implementation features

- The iteration count **`k`** to update the discriminator weights and **`num_iterations`** to update the generator weights are the parameters of the training algorithm
- Implementing backpropagation algorithm, ***the stochastic gradient descent*** (SGD) was used in the original training algorithm

CLASSIFICATION OF GENERATIVE ADVERSARIAL NETWORKS

Classification of generative adversarial networks (1)

- ***Fully Connected GANs***
- ***Conditional GANs*** (CGAN)
- ***Laplacian Pyramid of Adversarial Networks*** (LAPGAN)
- ***Deep Convolutional GANs*** (DCGAN)
- ***Generative Recurrent Adversarial Networks*** (GRAN)
- ***Information Maximizing GANs*** (InfoGAN)
- ***Bidirectional GANs*** (BiGAN)

* Alqahtani H., Kavakli-Thorne M., Kumar G. Applications of Generative Adversarial Networks (GANs): An Updated Review // Archives of Computational Methods in Engineering. – 2019.

Classification of generative adversarial networks (2)

- The following types of generative adversarial networks will remain outside the lecture:
  - ***Adversarial Autoencoders*** (AAE)
  - ***Variational Autoencoder-GAN*** (VAE-GAN)
  - Other specialized models

* Alqahtani H., Kavakli-Thorne M., Kumar G. Applications of Generative Adversarial Networks (GANs): An Updated Review // Archives of Computational Methods in Engineering. – 2019.

Fully Connected GANs

- ***Fully Connected GANs*** are deep models where the generator and discriminator are fully connected neural networks
- For the first time, fully connected GANs were applied to generate realistic images similar to the images in the following datasets*:
  – MNIST [http://yann.lecun.com/exdb/mnist]
  – CIFAR-10 [https://www.cs.toronto.edu/~kriz/cifar.html]
  – Toronto Face Dataset (TFD)**

* Goodfellow I.J., et al. Generative Adversarial Nets // Advances in neural information processing systems. – 2014. – P. 2672-2680. – [https://arxiv.org/pdf/1406.2661.pdf].

** Susskind J., Anderson A., Hinton G. E. The Toronto face dataset. Technical Report UTML TR 2010-001. – 2010.

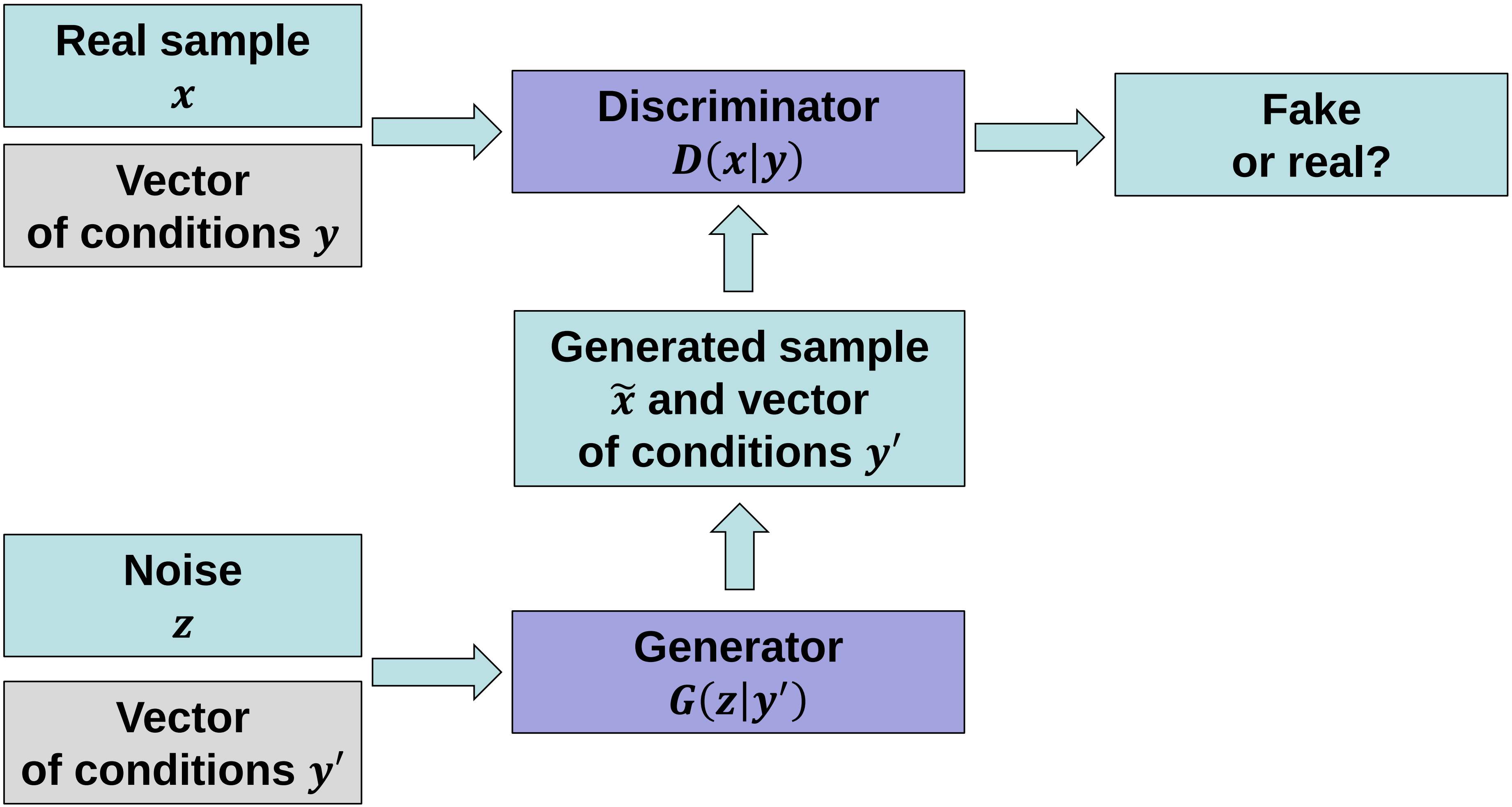
Conditional GANs (1)

- ***Conditional GANs*** (CGAN) are models that allow to generate synthetic images that satisfy certain conditions or have some properties (specific characteristics)

- The generator and discriminator receive auxiliary information
- In the simplest case, the image class (one-hot label vector) or properties of interest are used as auxiliary information

* Mirza M., Osindero S. Conditional generative adversarial nets. – 2014. – [https://arxiv.org/pdf/1411.1784.pdf].

Conditional GANs (2)



Conditional GANs (3)

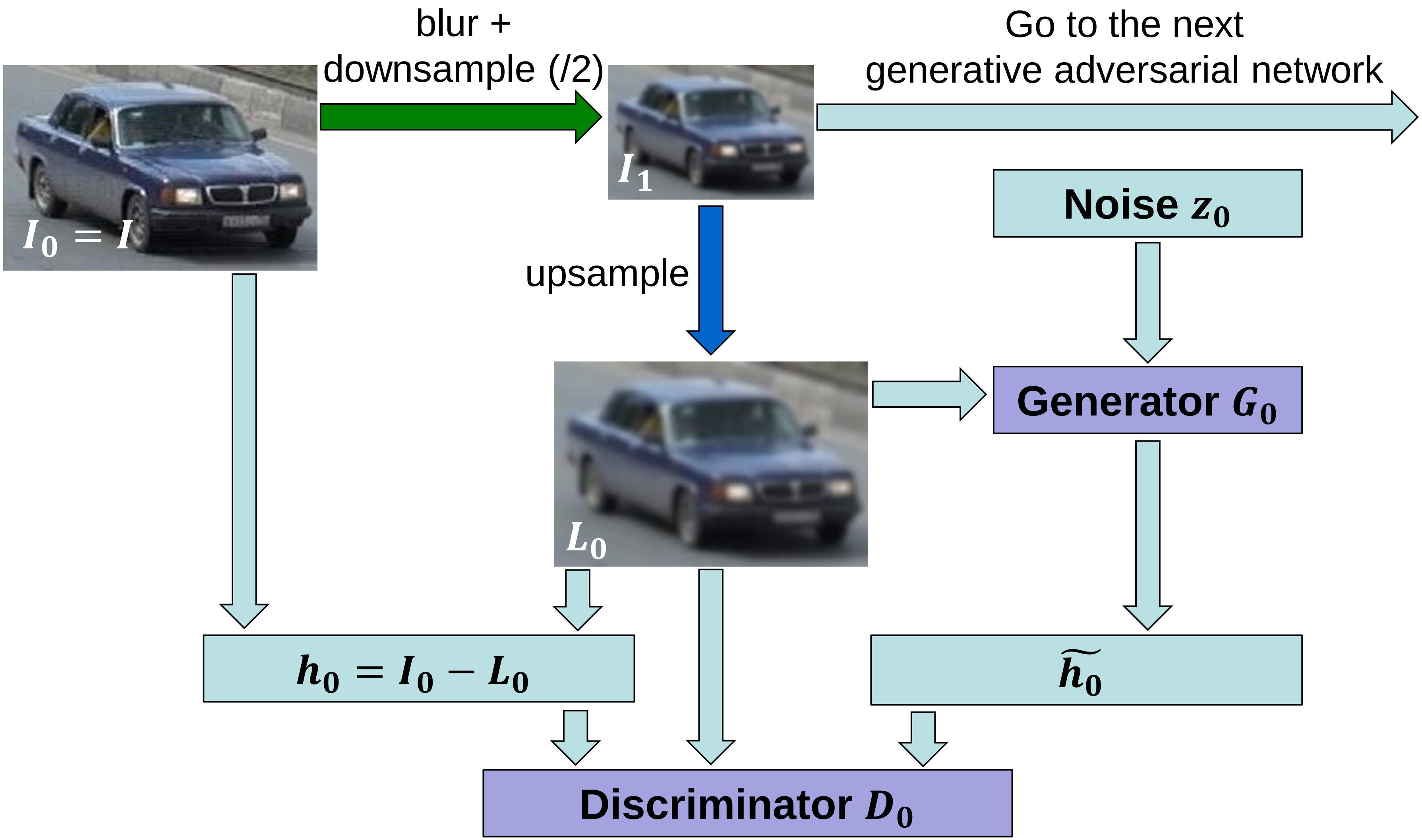
- ***Generator***
  - The input of the generator ceases to be completely random when supplementing a vector of conditions to the input
  - A vector of conditions helps the generator to understand how to generate samples in the best way
- ***Discriminator***
  - The discriminator makes a decision that the sample is a real one or no using the auxiliary information
- ***The training algorithm is similar to the considered one, the difference is the loss function, that depends on vectors of conditions***

Laplacian Pyramid of Adversarial Networks (1)

- ***Laplacian Pyramid of Adversarial Networks*** (LAPGAN) is a model based on a cascade of convolutional networks that form the Laplacian pyramid
- LAPGAN allows to generate high-resolution images in a coarse-to-fine fashion
- The Laplacian pyramid is based on the Gaussian one using downsampling and upsampling operations
- An element of the Laplacian pyramid is defined by the difference between adjacent levels of the Gaussian pyramid

* Denton E.L., et al. Deep generative image models using a Laplacian pyramid of adversarial networks // Advances in neural information processing systems. – 2015. – P. 1486-1494. – [https://arxiv.org/pdf/1506.05751.pdf].

Laplacian Pyramid of Adversarial Networks (2)

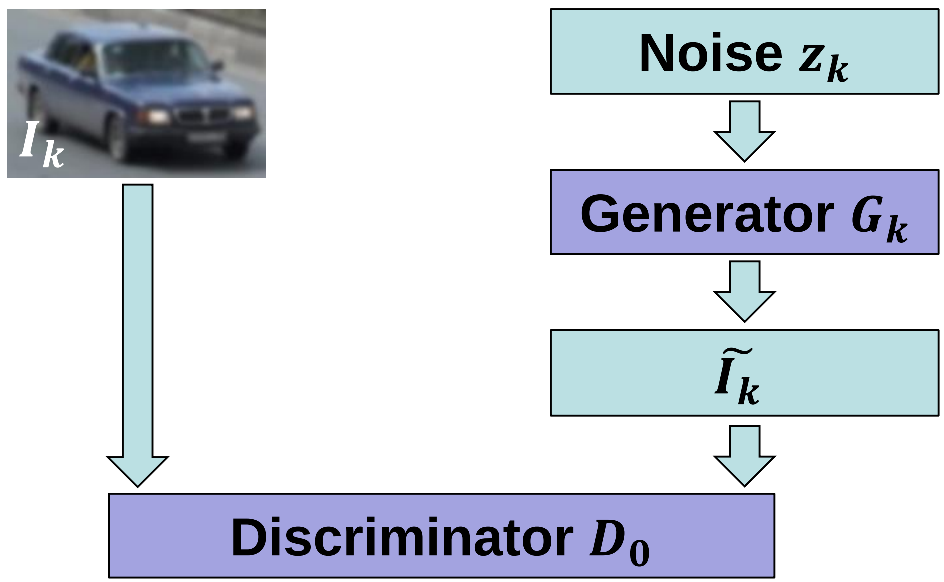


Laplacian Pyramid of Adversarial Networks (3)

- The first adversarial network in the framework receives the input image during the training
- The image is blurring and downsampling by 2 times
- The image resolution is increased to the resolution of the original one using upsampling
- ***Generator*** receives the upsampled image and noise and tries to predict the difference between the upsampled and original images
- ***Discriminator*** decides whether the difference is real
- ***The training of LAPGAN is the training of a conditional generative adversarial network***

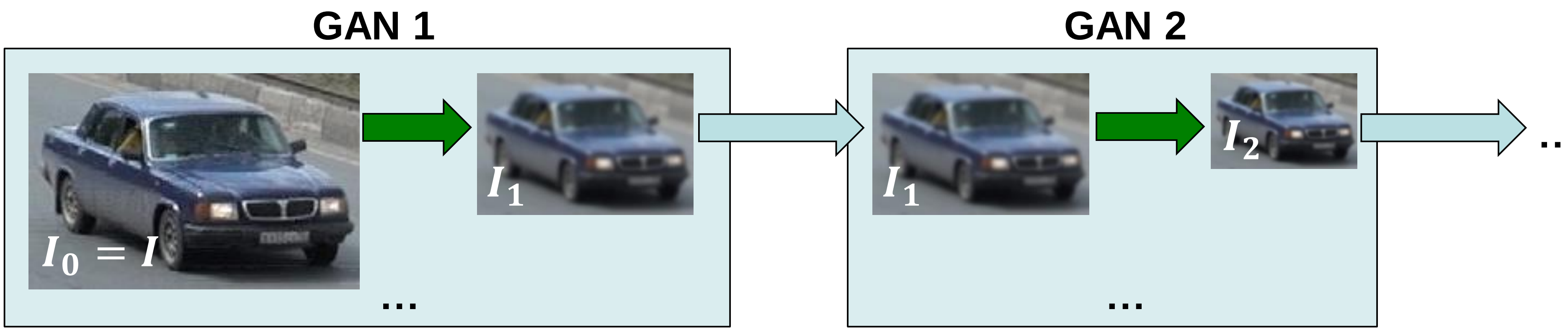
Laplacian Pyramid of Adversarial Networks (4)

- The scheme of each subsequent generative adversarial network in the framework is similar to the represented one, excepting the last network
- ***The last network provides image reconstruction based on the noise***, i.e. it is not a conditional generative adversarial network



Laplacian Pyramid of Adversarial Networks (5)

- Training the next model, the image blurred and downsampled for the previous network is used as the input for the next one



- GANs in the pyramid can be trained independently
- The use of the trained framework is a backward pass through the constructed sequence of models; a low-resolution image is the input of the last model

Deep Convolutional GANs (1)

- ***Deep Convolutional GANs*** (DCGAN) are models where generator and discriminator are deep convolutional networks with the following limitations:
  - No fully connected layers
  - Discriminator contains strided convolutions instead of pooling layers, and generator contains fractional-strided convolutions
  - Discriminator and generator contain batch normalization
  - …

* Radford A., Metz L., Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks. – 2015. – [https://arxiv.org/pdf/1511.06434.pdf].

Deep Convolutional GANs (2)

- ***Deep Convolutional GANs*** (DCGAN) are models where generator and discriminator are deep convolutional networks with the following limitations:
  - For all layers of the generator, except the last one, ReLU is used as the activation function. Hyperbolic tangent (Tanh) was used in the original paper*
  - Leaky ReLU is used for all layers of the discriminator

* Radford A., Metz L., Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks. – 2015. – [https://arxiv.org/pdf/1511.06434.pdf].

Generative Recurrent Adversarial Networks (1)

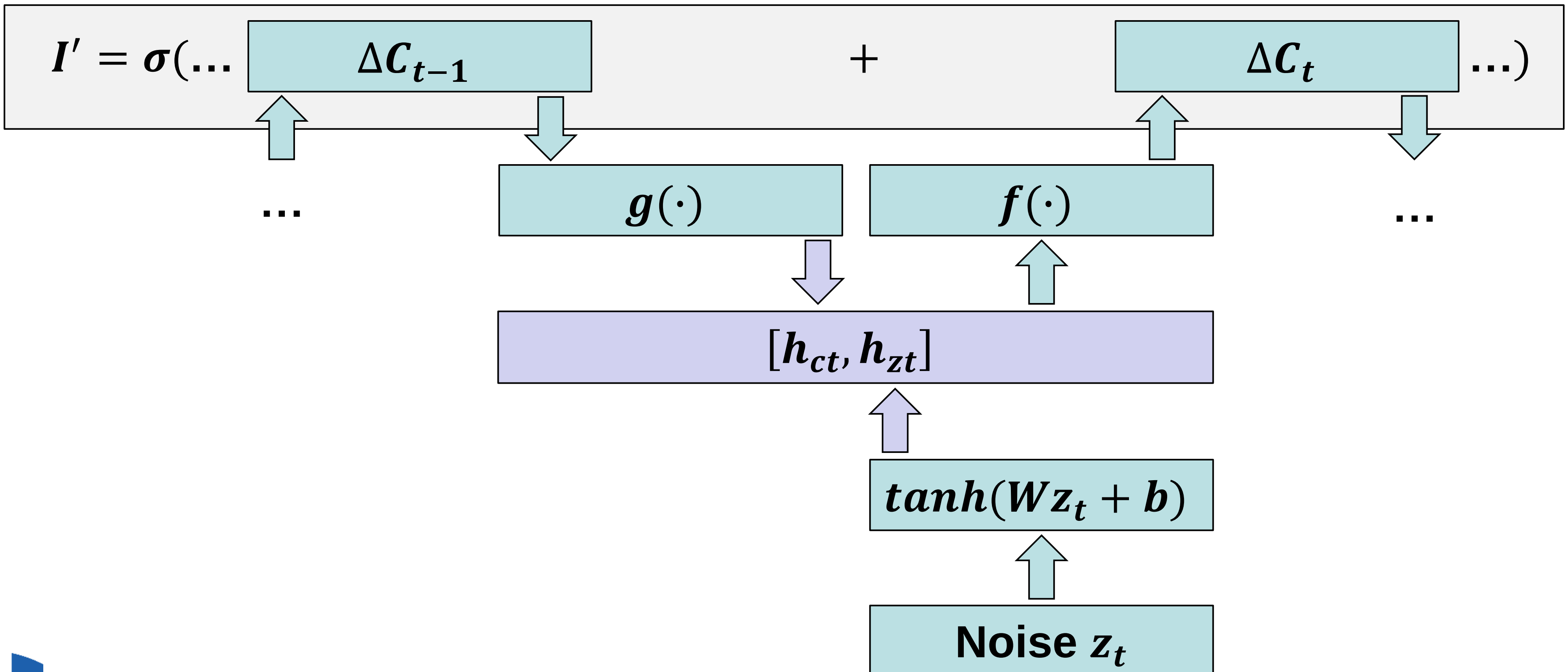
- ***Generative Recurrent Adversarial Networks*** (GRAN) are models where the generator is a recurrent neural network

- Generator receives a sequence of noise samples and generates a sequence of images
- The next image of the constructed sequence accumulates updates, and it leads to the final sample
- Discriminator determines whether the final image is a real or fake one
- Further, we consider the generator structure

* Im D.J., Kim C.D., Jiang H., Memisevic R. Generating images with recurrent adversarial networks. – 2016. – [https://arxiv.org/pdf/1602.05110.pdf].

Generative Recurrent Adversarial Networks (2)

- $\{z_t, t = \overline{1,T}\}, z_t \sim p(Z)$ is a sequence of noise samples
- $\Delta C_1, \Delta C_2, \dots, \Delta C_T$ is a sequence of generated images



Generative Recurrent Adversarial Networks (3)

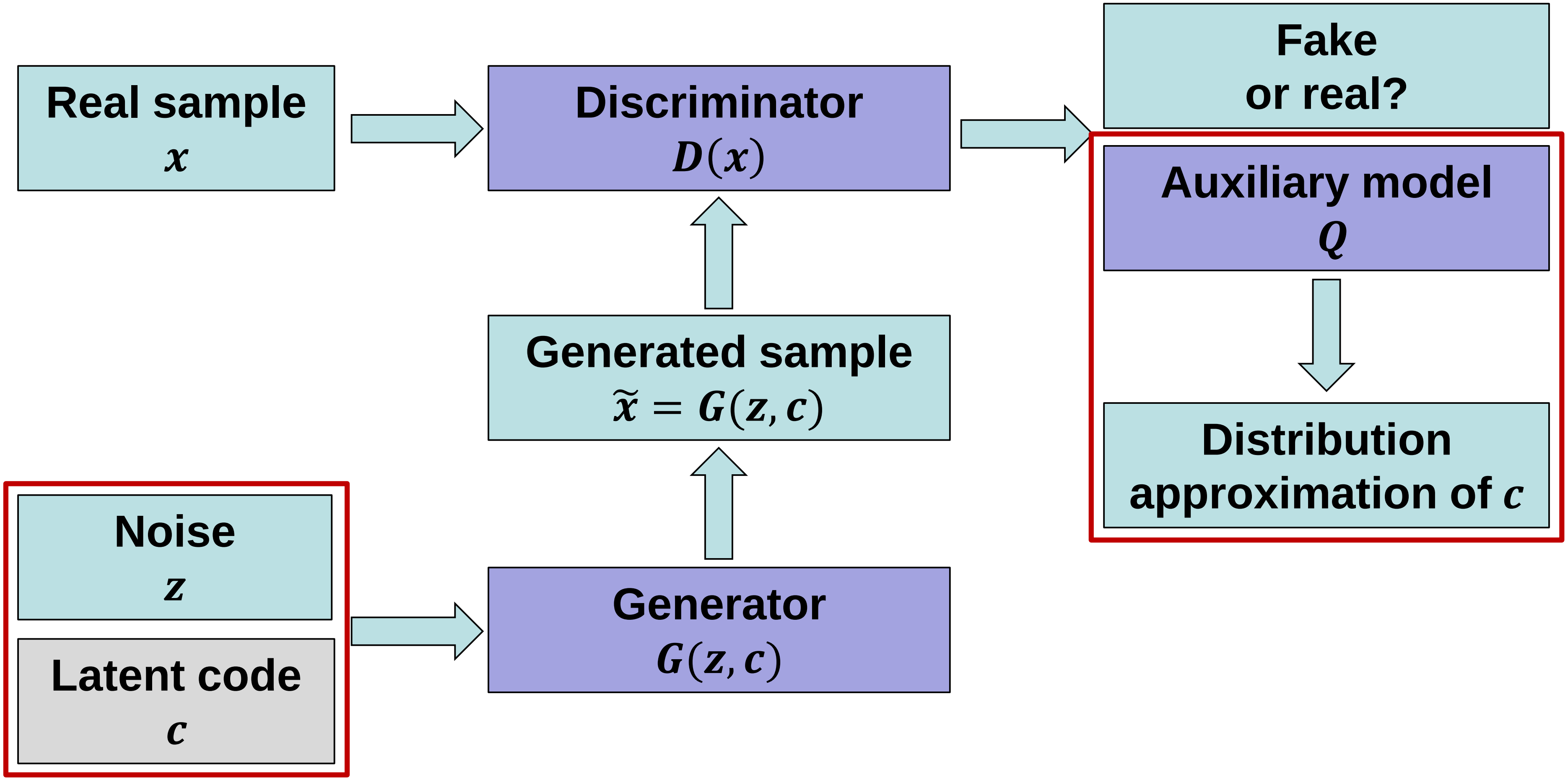
- $g(\cdot)$ is a network represented by a sequence of convolutional layers and a fully connected layer, it works as an encoder
- $f(\cdot)$ is a reverse copy of the network $g(\cdot)$ (a fully connected layer and a sequence of convolutional layers), it works as a decoder
- h_{ct} is an encoded representation of the image generated in the step $t-1$
- h_{zt} is a hypothesis about required updates
- $[h_{ct}, h_{zt}]$ is a concatenation
- The generator output is a sum of all generated images
- ***Backpropagation through time*** is used as the model training algorithm

Information Maximizing GANs (1)

- ***Information Maximizing GANs*** (InfoGAN) is an information-theoretic extension of the generative adversarial network that is able to learn disentangled representations in a completely unsupervised manner
- Disentangled representation is a collection of features that represent the characteristics of a data sample and can be useful for a wide range of tasks
- The goal of these models is to construct meaningful representations by maximizing mutual information between a small subset of noise and observation variables

* Chen X., et al. InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets // Advances in Neural Information Processing Systems. – 2016. – P. 2172-2180. – [https://arxiv.org/pdf/1606.03657.pdf].

Information Maximizing GANs (2)



Information Maximizing GANs (3)

- The input noise vector is decomposed into two parts:
  - z is treated as source of incompressible noise
  - c is the latent code which will target the salient structured semantic features of the data distribution
- $G(z,c)$ is a generated data sample
- The mutual information $I\big(c;G(z,c)\big)$ between latent codes c and generator distribution $G(z,c)$ should be high
- The mutual information can be expressed as the difference of two entropy terms:

$$I\big(c;G(z,c)\big) = H(c) - H(c|G(z,c)) = H\big(G(z,c)\big) - H(G(z,c)|c)$$

- InfoGAN is defined as the following minimax game:

$$E_{x\sim p_{data}}[\log D(x)] + E_{z\sim p_Z, c\sim p_C}\left[\log\Big(1 - D\big(G(z,c)\big)\Big)\right] - \lambda I\big(c;G(z,c)\big)$$

Information Maximizing GANs (4)

- In practice, the mutual information term $I\big(c; G(z,c)\big)$ is hard to maximize directly as it requires access to the posterior $P(c|\tilde{x})$
- We can obtain a lower bound of it by defining an auxiliary distribution $Q(c|\tilde{x})$ to approximate $P(c|\tilde{x})$
- $Q(c|\tilde{x})$ is modeled by a neural network
- Based on $Q(c|\tilde{x})$, we can obtain a lower bound for the value of mutual information

$$I\big(c; G(z,c)\big) = H(c) - H(c|G(z,c))$$

$$= E_{\tilde{x}\sim G(z,c)}\left[E_{c'\sim P(c|\tilde{x})}[\log P(c'|\tilde{x})]\right] + H(c)$$

$$= E_{\tilde{x}\sim G(z,c)}\left[\underbrace{\boxed{D_{KL}\big(P(\cdot\,|\tilde{x})||Q(\cdot\,|\tilde{x})\big)}}_{\geq 0} + E_{c'\sim P(c|\tilde{x})}[\log Q(c'|\tilde{x})]\right] + H(c)$$

$$\geq E_{\tilde{x}\sim G(z,c)}\left[E_{c'\sim P(c|\tilde{x})}[\log Q(c'|\tilde{x})]\right] + \underbrace{\boxed{H(c)}}_{const}$$

Information Maximizing GANs (5)

- The following equality was proved in the original paper:

$$E_{\tilde{x}\sim G(z,c)}\left[E_{c'\sim P(c|\tilde{x})}[\log Q(c'|\tilde{x})]\right] = E_{c\sim p_C,\,\tilde{x}\sim G(z,c)}[\log Q(c|\tilde{x})]$$

- Therefore, the regularization term can be calculated using the following algorithm:
  – Sample the latent code c from the distribution p_C
  – Sample the noise vector z from the distribution p_Z
  – Generate the data sample $\tilde{x} = G(z, c)$
  – Calculate the probability $Q\big(c|\tilde{x} = G(z, c)\big)$
- The final loss function is as follows:

$$E_{x\sim p_{data}}[\log D(x)] + E_{z\sim p_Z, c\sim p_C}\left[\log\Big(1 - D\big(G(z,c)\big)\Big)\right] - \lambda\Big(E_{c\sim p_C,\,\tilde{x}\sim G(z,c)}[\log Q(c|\tilde{x})] + H(c)\Big)$$

Information Maximizing GANs (6)

- As a rule, the neural networks D and Q have common convolutional layers, and there is only one fully connected layer for deriving the distribution parameters $Q(c|x)$

- Latent codes can be either categorical or continuous
  - A typical categorical code is the class of the generated data sample (for example, a digit from 0 to 9 in the task of generating handwritten digits)
  - A typical continuous code is the value of the distribution parameter, which corresponds to any feature of the generated data samples (for example, the inclination angle of the generated handwritten digit)

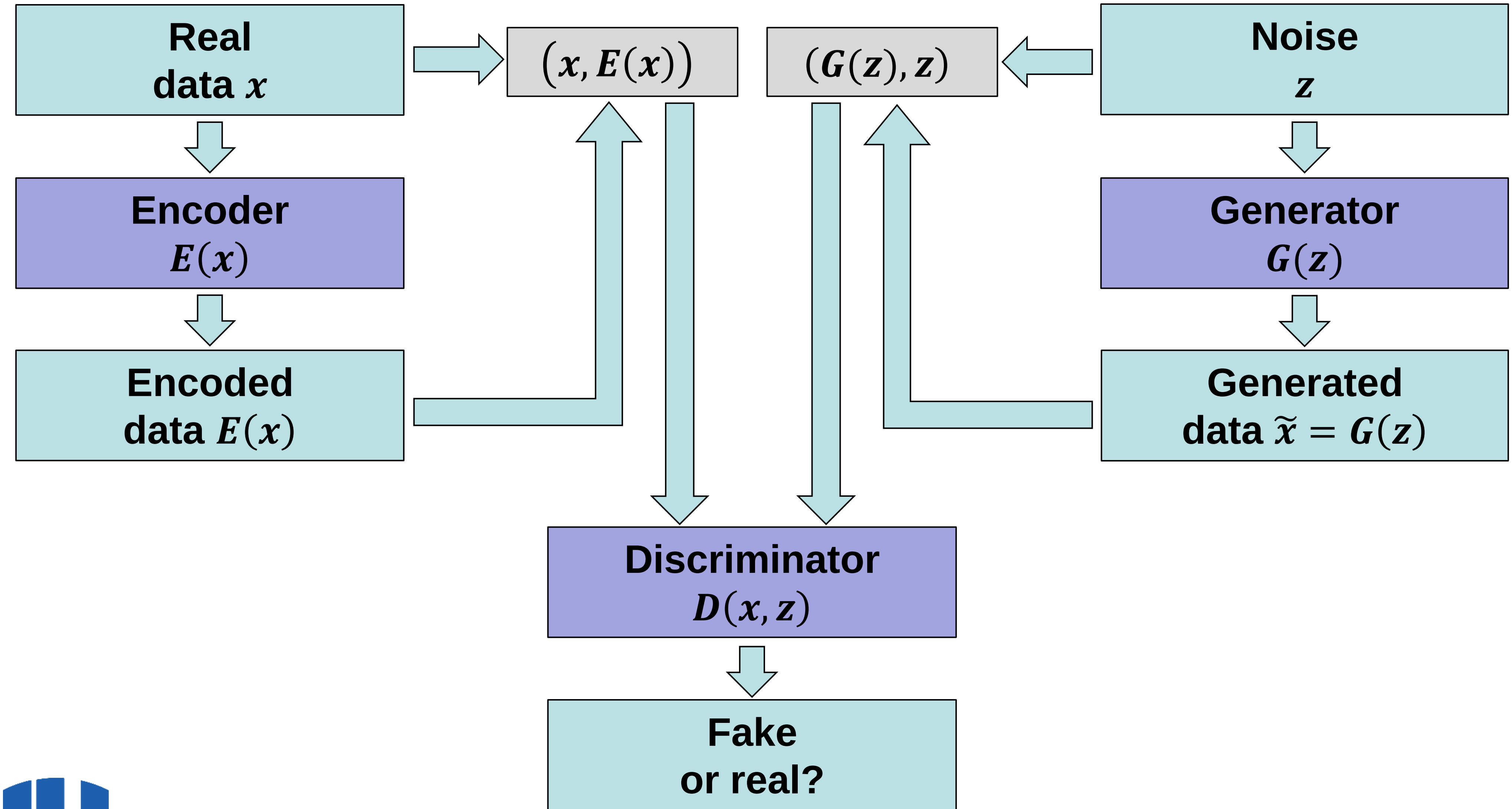
Bidirectional GANs (1)

- ***Bidirectional GANs*** (BiGAN) are models in which the discriminator for deciding whether the data is real uses, in addition to the data itself, their representation in latent space

- The input data is encoded into a representation in the latent space by a special block called ***the encoder***
- The discriminator receives an input pair (sample data, representation in the latent space) in the case of real data and (generated sample, noise) processing the generator output

* Donahue J., Krahenbuhl P., Darrell T. Adversarial feature learning. – 2017. – [https://arxiv.org/pdf/1605.09782.pdf].

Bidirectional GANs (2)



Bidirectional GANs (3)

- ***The loss function*** is as follows:

$$E_{x\sim p_{data}}\left[\log D\big(x,E(x)\big)\right]+E_{z\sim p_Z}\left[\log\big(1-D(G(z),z)\big)\right]$$
$$=E_{x\sim p_{data}}\left[E_{\tilde{z}\sim p_E(\cdot|x)}[D(x,\tilde{z})]\right]+E_{z\sim p_Z}\left[E_{\tilde{x}\sim p_G(\cdot|z)}[1-D(\tilde{x},z)]\right],$$

  where $p_E(\cdot\,|x)$ is a distribution modeled by the encoder, $p_G(\cdot\,|z)$ is a distribution modeled by the generator

- In BiGANs, the encoder is trained at the same time as the generator, and it models the distribution $p_E(z|x)=\delta\big(z-E(x)\big)$
- The discriminator estimates the probability $p_D(y|x,z)$, where $y=1$, if x is a real data sample, and $y=0$, if x is a generated sample

Other types of generative adversarial networks

- Generative adversarial networks are not limited to those listed in the represented classification
- The considered types are widely used, and there are various modifications of these models
- There are specialized generative adversarial networks that solve specific problems
- Further, we consider applications of generative adversarial networks

APPLICATIONS OF GENERATIVE ADVERSARIAL NETWORKS

Data augmentation

- ***Data augmentation*** means generating of synthetic data similar to the data in some existing dataset, but containing various transformations. As a rule, data augmentation is required to extend the train dataset

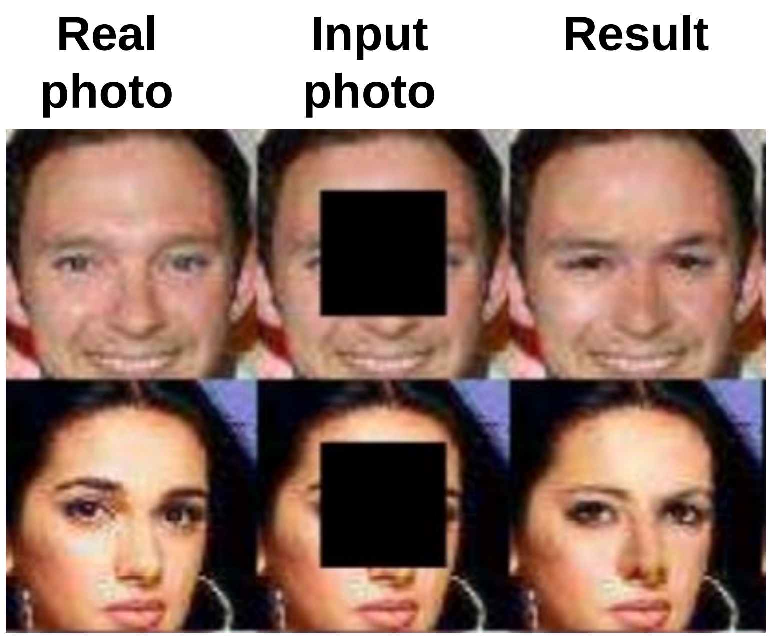
- ***CycleGAN*** [https://www.nature.com/articles/s41598-019-52737-x] (2019) – computer tomography (CT) segmentation tasks
- ***Data Augmentation GAN (DAGAN)*** [https://arxiv.org/pdf/1711.04340.pdf] (2018) – data augmentation based on transformations
- ***Balancing GAN (BAGAN)*** [https://arxiv.org/pdf/1803.09655.pdf] (2018) – data augmentation with balancing

Generating high-resolution images

- ***Laplacian Pyramid of Adversarial Networks (LAPGAN)*** [https://arxiv.org/pdf/1506.05751.pdf] (2015) – increasing image resolution
- ***Generative Adversarial What-Where Network (GAWWN)*** [https://arxiv.org/pdf/1610.02454.pdf] (2016) – generating images using auxiliary description and information about object displacement
- ***Generative Adversarial Network for image Super-Resolution (SRGAN)*** [https://arxiv.org/pdf/1609.04802.pdf] (2017) – increasing image resolution
- ***Self-Attention Generative Adversarial Network (SAGAN)*** [https://arxiv.org/pdf/1805.08318.pdf] (2019) – modeling dependencies between the individual parts of the image to generate high-resolution images

Image inpainting (1)

- ***Image inpainting*** means removing unwanted objects in the image or restoring damaged fragments of old photos



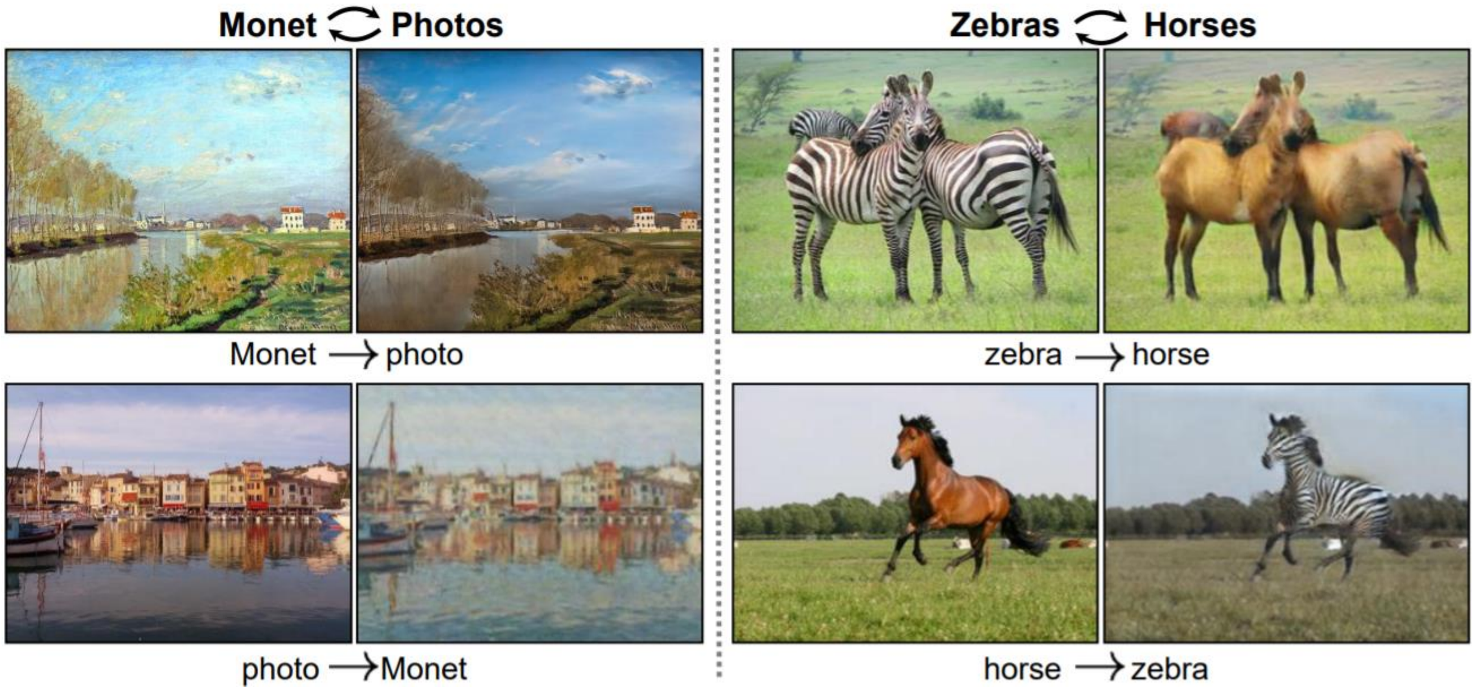
* Yeh R.A., et al. Semantic Image Inpainting With Deep Generative Models // In the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – 2017. – P. 5485-5493. – [http://openaccess.thecvf.com/content_cvpr_2017/html/Yeh_Semantic_Image_Inpainting_CVPR_2017_paper.html].

Image inpainting (2)

- ***Semantic Image Inpainting with Deep Generative Models*** [http://openaccess.thecvf.com/content_cvpr_2017/html/Yeh_Semantic_Image_Inpainting_CVPR_2017_paper.html] (2017) – restoring damaged fragments of photos (faces, cars, etc.)
- ***EdgeConnect*** [https://arxiv.org/pdf/1901.00212.pdf] (2019) – restoring damaged fragments of photos (faces, real-life images)
- ***PEPSI*** [http://openaccess.thecvf.com/content_CVPR_2019/html/Sagong_PEPSI__Fast_Image_Inpainting_With_Parallel_Decoding_Network_CVPR_2019_paper.html] (2019) – reducing the number of calculations when restoring damaged fragments of photos using generative adversarial networks

Style transfer (1)

- ***Style transfer*** means transferring the style of one image to others, for example, transferring the style of drawing pictures by a painter in a photo



* Zhu J.-Y., Park T., Isola P., Efros A.A. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. – 2018. – [https://arxiv.org/pdf/1703.10593.pdf].

Style transfer (2)

- ***CycleGAN*** [https://arxiv.org/pdf/1703.10593.pdf] (2018) – unpaired image-to-image translation (transferring the style of drawing pictures by a painter in a photo, transferring winter effects to the summer photo, etc.)
- ***StyleGAN*** [https://arxiv.org/pdf/1812.04948.pdf] (2019) – generating human faces and style transfer (glasses, long hair, etc.) from one face to another one
- ***BigGAN*** [https://arxiv.org/pdf/1809.11096.pdf] (2019) – generating real-life images

Conclusion

- Generative adversarial networks are deep models that are widely used to generate synthetic data for solving various problems
- The range of problems covers both entertainment and practically significant areas
- Generative adversarial networks are actively developing. New modifications of these networks appear, these models contain elements of other deep neural networks

Literature

- Goodfellow I.J., et al. Generative Adversarial Nets // Advances in neural information processing systems. – 2014. – P. 2672-2680. – [https://arxiv.org/pdf/1406.2661.pdf].
- Goodfellow I.J. NIPS 2016 Tutorial: Generative Adversarial Networks. – 2016. – [https://arxiv.org/pdf/1701.00160.pdf].
- Alqahtani H., Kavakli-Thorne M., Kumar G. Applications of Generative Adversarial Networks (GANs): An Updated Review // Archives of Computational Methods in Engineering. – 2019.
- Foster D. Generative Deep Learning. – O'Reilly. – 2019.
- Sandfort V., Yan K., Pickhardt P.J., Summers R.M. Data augmentation using generative adversarial networks (CycleGAN) to improve generalizability in CT segmentation tasks // Scientific Reports. – 2019. – [https://www.nature.com/articles/s41598-019-52737-x].

Authors

- **Turlapov Vadim Evgenievich,** Dr., Prof., department of computer software and supercomputer technologies
  vadim.turlapov@itmm.unn.ru
- **Vasiliev Engeny Pavlovich,** lecturer, department of computer software and supercomputer technologies
  evgeny.vasiliev@itmm.unn.ru
- **Getmanskaya Alexandra Alexandrovna,** lecturer, department of computer software and supercomputer technologies
  alexandra.getmanskaya@itmm.unn.ru
- **Kustikova Valentina Dmitrievna**
  Phd, assistant professor, department of computer software and supercomputer technologies
  valentina.kustikova@itmm.unn.ru