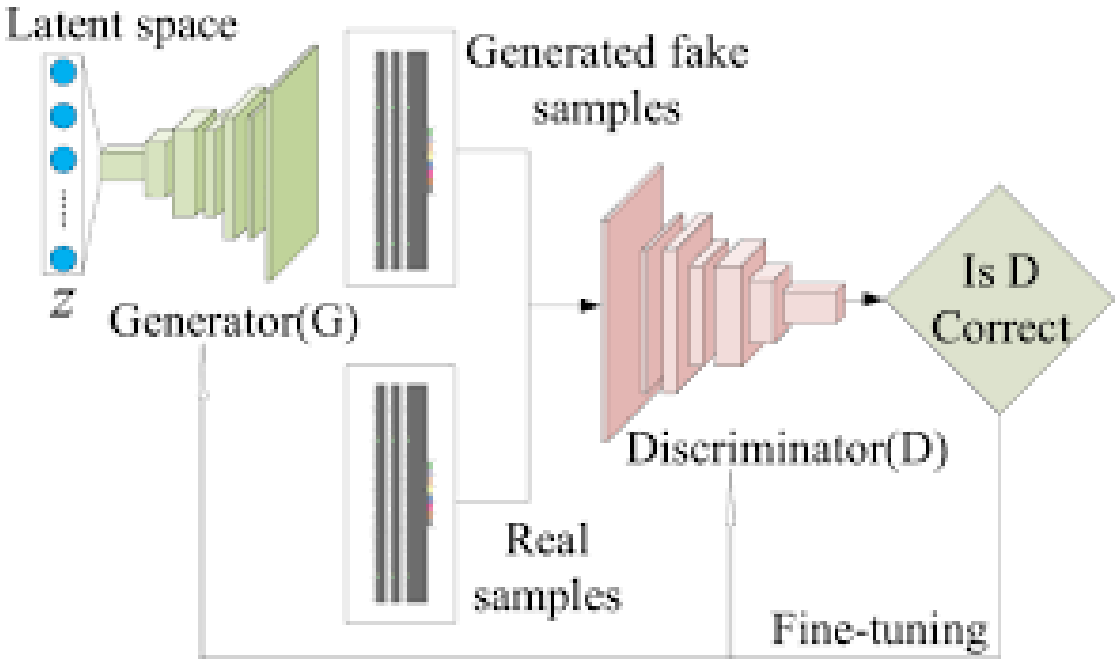
Deep Learning
Wykład 8



Generative Adversary Networks GAN

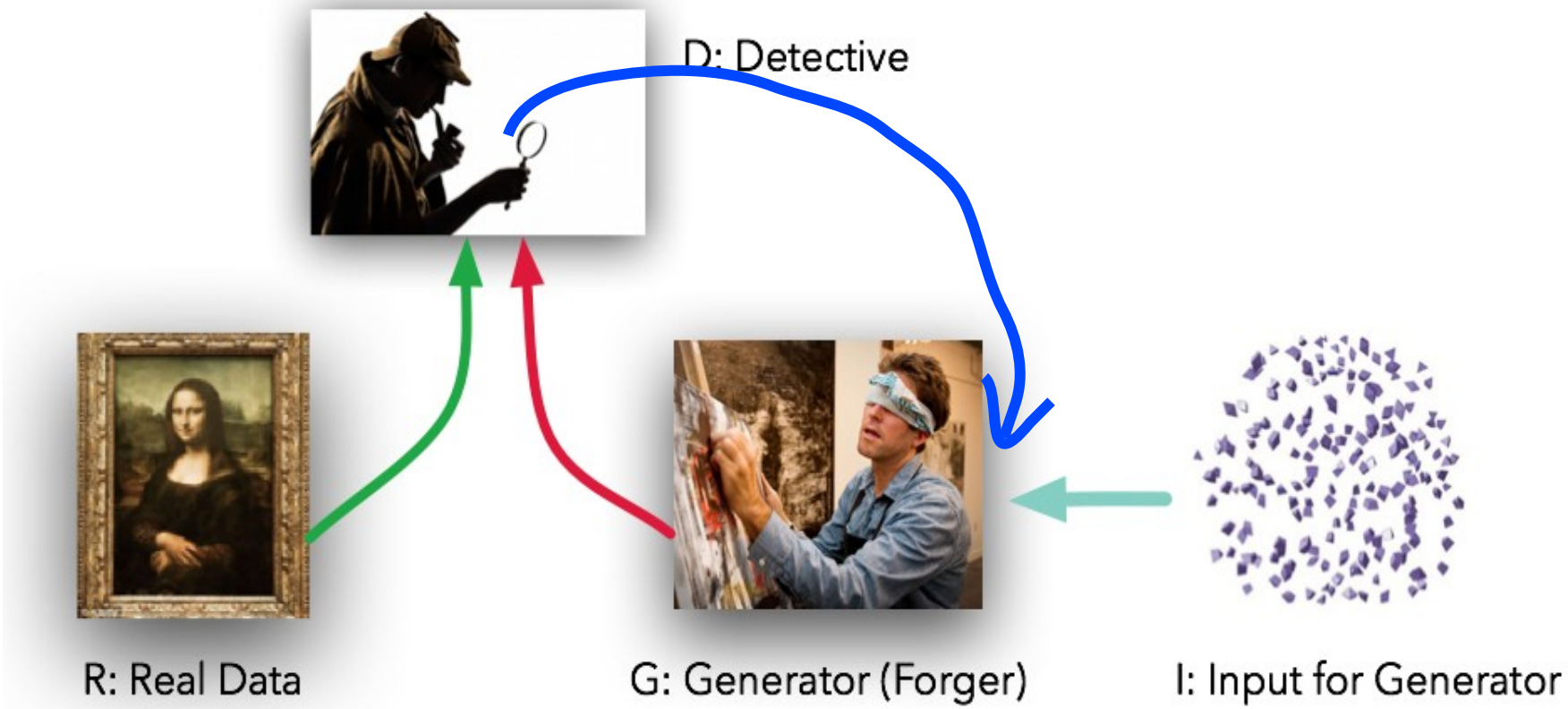
Marcin Wolter, *IFJ PAN, 27 January 2021*

Informacyjnie

- Zgodnie z informacją na stronie Wirtualna Uczelnia wczoraj były nasze ostatnie cwiczenia. Mamy jeszcze dwa wykłady w przyszłym tygodniu: w poniedziałek i we środę.
- Proszę o projekty!!!

Głębokie uczenie	2018/19 ID stac.	5	Grupa la 02	la 02	26.01.2021 wtorek	08:00	10:15	WM_G_wirtualna	lab	L	Zaliczenie z oceną
Głębokie uczenie	2018/19 ID stac.	5	Grupa la 01	la 01	26.01.2021 wtorek	11:00	13:15	WM_G_wirtualna	lab	Zaj.	Zaliczenie z oceną
Głębokie uczenie	2018/19 ID stac.	5	Grupa wy 01	wy 01	27.01.2021 środa	12:15	14:30	WM_G_wirtualna	wyk	W	Egzamin
Głębokie uczenie	2018/19 ID stac.	5	Grupa wy 01	wy 01	01.02.2021 poniedziałek	12:15	14:30	WM_G_wirtualna	wyk	W	Egzamin
Głębokie uczenie	2018/19 ID stac.	5	Grupa wy 01	wy 01	03.02.2021 środa	12:15	14:30	WM_G_wirtualna	wyk	W	Egzamin

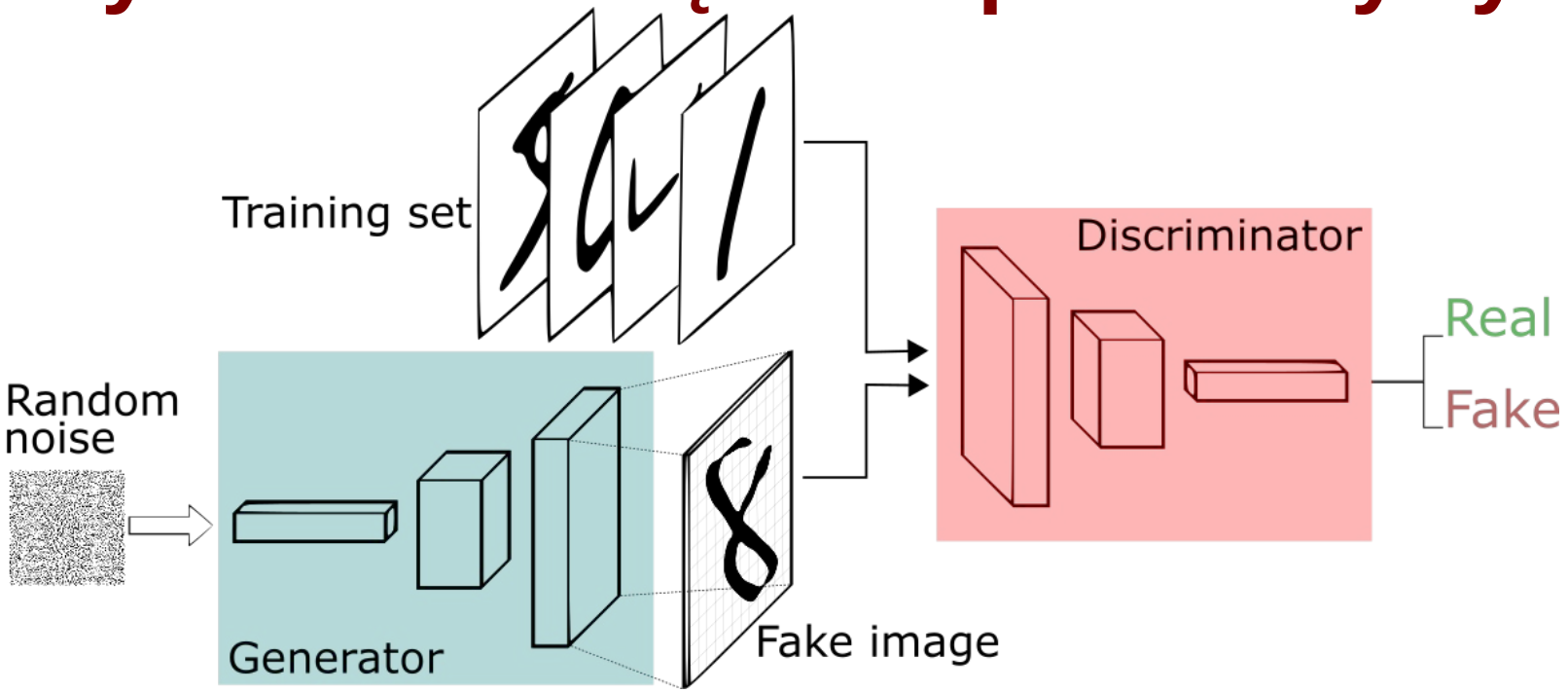
Niewidomy fałszerz i detektyw



Fałszerz nigdy nie widział portretu Mona Lisy, ale dostaje oceny detektywa I stara się, aby jego obraz został przez niego uznany za podobny do oryginału.

Obydwaj muszą uczyć się równolegle. Jest to ważne, gdyż zbyt dobry detektyw nie pozwoli namalować fałszerzowi niczego akceptowalnego, a zbyt słaby zaakceptuje wszystko.

Przykład GAN- ręcznie pisane cyfry

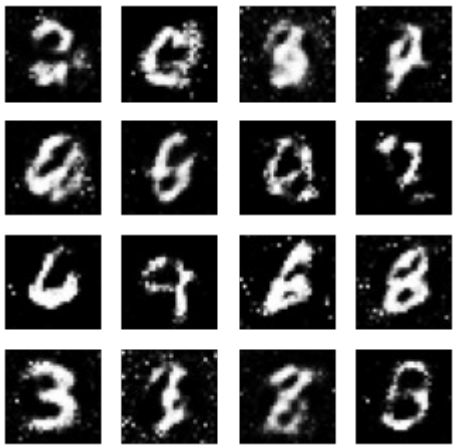


- **Zbiór treningowy** – MNIST: ręcznie pisane cyfry od US Post.
- **Dyskryminator/klasyfikator**– konwolucyjna sieć neuronowa rozpoznająca cyfry jako “prawdziwe” lub “fałszywe”.
- **Generator** – odwrócona sieć konwolucyjna (lub w pełni połączona “Dense”):

  Implementacja: kod w języku Python z użyciem Keras i TensorFlow.

Generowanie cyfr MNIST zwykła sieć GAN i DCGAN

- Przykład z repozytorium programów Tensorflow:
  - https://www.tensorflow.org/tutorials
- Przykład z użyciem zwykłej sieci Dense:
  - https://github.com/marcinwolter/DeepLearning_2020/blob/main/tensorflow_gan.ipynb
- I z użyciem sieci konwolucyjnej:
  - https://github.com/marcinwolter/DeepLearning_2020/blob/main/tensorflow_dcgan.ipynb
- Ten sam szkielet programu i dane, zmieniamy tylko sieć „dense” na konwolucyjną.

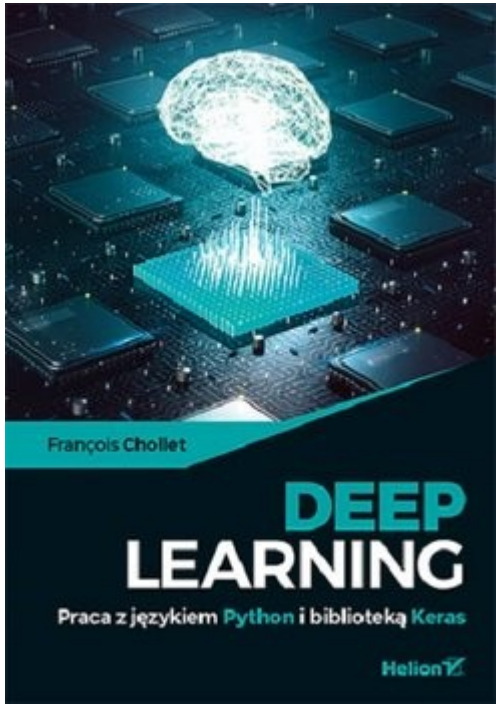


dense

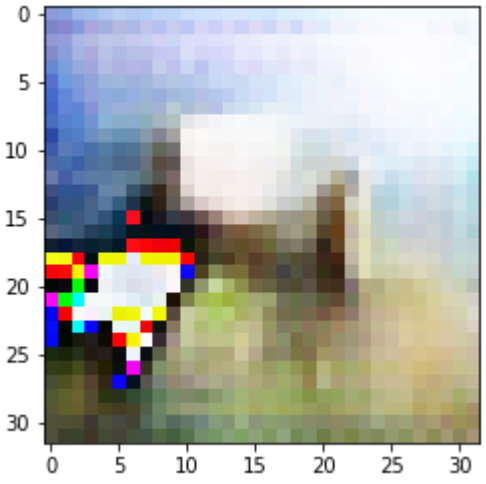
dcgan

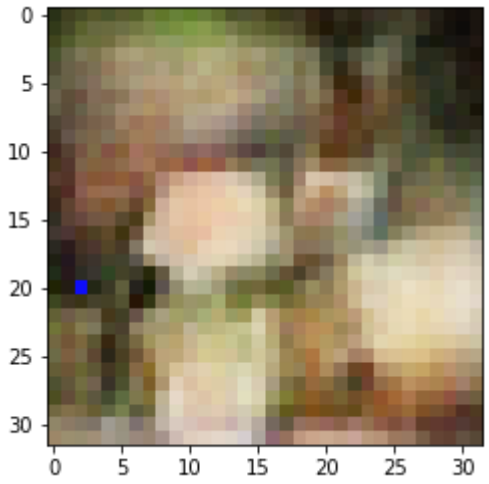
Tworzenie nowych obrazków z CIFAR10

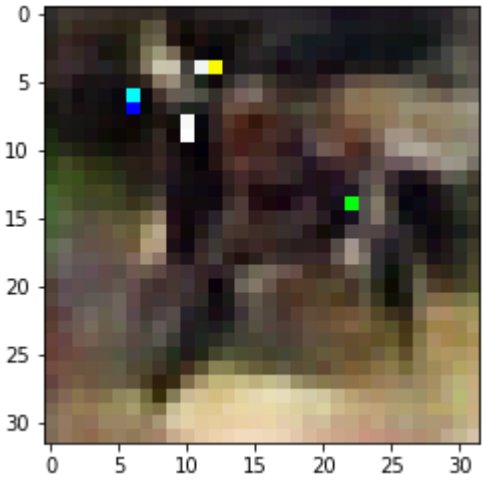
- Przykład z książki: Francois Chollet „Deep Learning”: https://www.ceneo.pl/75394698;pla?se=YxWbm1iqQxdyrhZALD2q02WnsAqEsNg5&gclid=CjwKCAiA9bmABhBbEiwASb35V-5QGgESxuZHwPQWqiAmyV-Q0dYUDDBwBgtMnUhhO3Z3mjTAym09jRoCJ7MQAvD_BwE
  - https://github.com/marcinwolter/DeepLearning_2020/blob/main/8_5_introduction_to_gans.ipynb



- Program używa danych CIFAR10 i stara się wygenerować obrazek „koń”







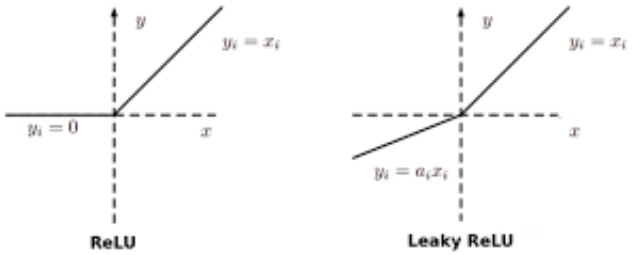
GAN

- Sieć generatora (generator) mapuje wektory o kształcie (latent_dim,) na obrazy o kształcie (32, 32, 3).
- Sieć dyskryminatora (discriminator) mapuje obrazy o kształcie (32, 32, 3) na binarną wartość określającą prawdopodobieństwo tego, że obraz jest prawdziwy,
- Sieć gan tworzy łańcuch składający się z generatora i dyskryminatora: gan(x)

  =discriminator(generator (x)). Sieć gan mapuje wektory niejawnej przestrzeni na oceny realizmu wystawiane przez dyskryminator.
- Trenujemy dyskryminator przy użyciu przykładów prawdziwych i sztucznych obrazów oznaczonych etykietami, tak jakbyśmy trenowali zwykły model klasyfikacji obrazów.
- W celu wytrenowania generatora korzystamy z gradientów wag generatora w odniesieniu do straty modelu gan. W związku z tym każdy krok trenowania ma modyfikować wagi generatora tak, aby zwiększyć prawdopodobieństwo zaklasyfikowania wygenerowanych obrazów jako prawdziwych. Innymi słowy, trenujemy generator tak, aby był w stanie oszukać dyskryminator.

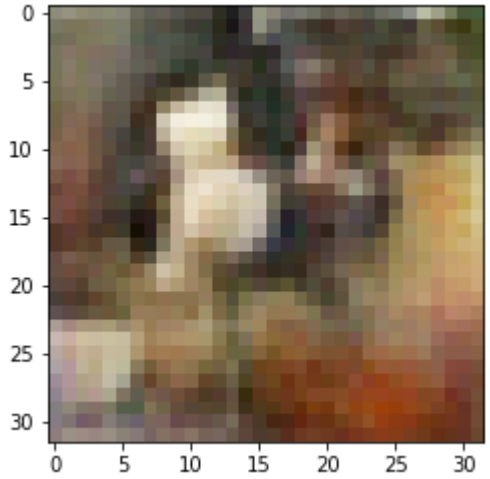
Zbiór przydatnych rozwiązań

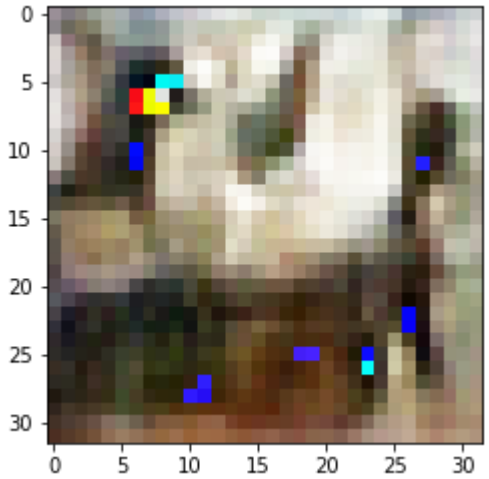
- Ostatnią warstwą aktywacji modelu jest funkcja *tanh*, a nie *sigmoid* spotykana w większości innych innych modeli.
- Stochastyczność przyczynia się do uzyskania bardziej solidnego modelu. Trenowanie sieci GAN przyczynia się do powstania dynamicznej równowagi, a więc proces ten może utknąć w wielu punktach. Wprowadzanie losowości do procesu trenowania pomaga temu zapobiec. Losowość wprowadzamy na dwa sposoby: korzystanie z mechanizmu odrzucania zaimplementowanego w dyskryminatorze oraz poprzez dodanie losowego szumu do etykiet przetwarzanych przez dyskryminator.
- Rzadkie gradienty mogą zaszkodzić. Dlatego nie korzystamy z operacji *maxpooling* i aktywacji *Relu* – używamy aktywacji *LeakyRelu*

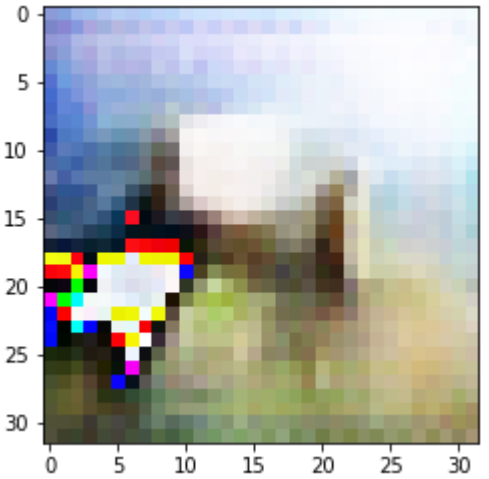


- Aby w generowanych obrazach unikać artefaktów przypominających szachownicę korzystając z Conv2DTransposed / Conv2D stosujemy rozmiar jądra podzielny przez rozmiar kroku.

Wyniki







- Dopatrzenie się konia wymaga trochę wyobraźni...
- Prawdopodobnie dłuższy trening dałby lepsze wyniki
- Problem jest bardzo złożony

Generacja twarzy - DCGAN

- DCGAN do generacji twarzy na podstawie danych ze zbioru CELEBA.

  https://github.com/marcinwolter/DeepLearning_2020/blob/main/dcgan_overriding_train_step.ipynb

- Zamiast liter generujemy twarze (bardzo zredukowana rozdzielczość dla zwiększenia szybkości).

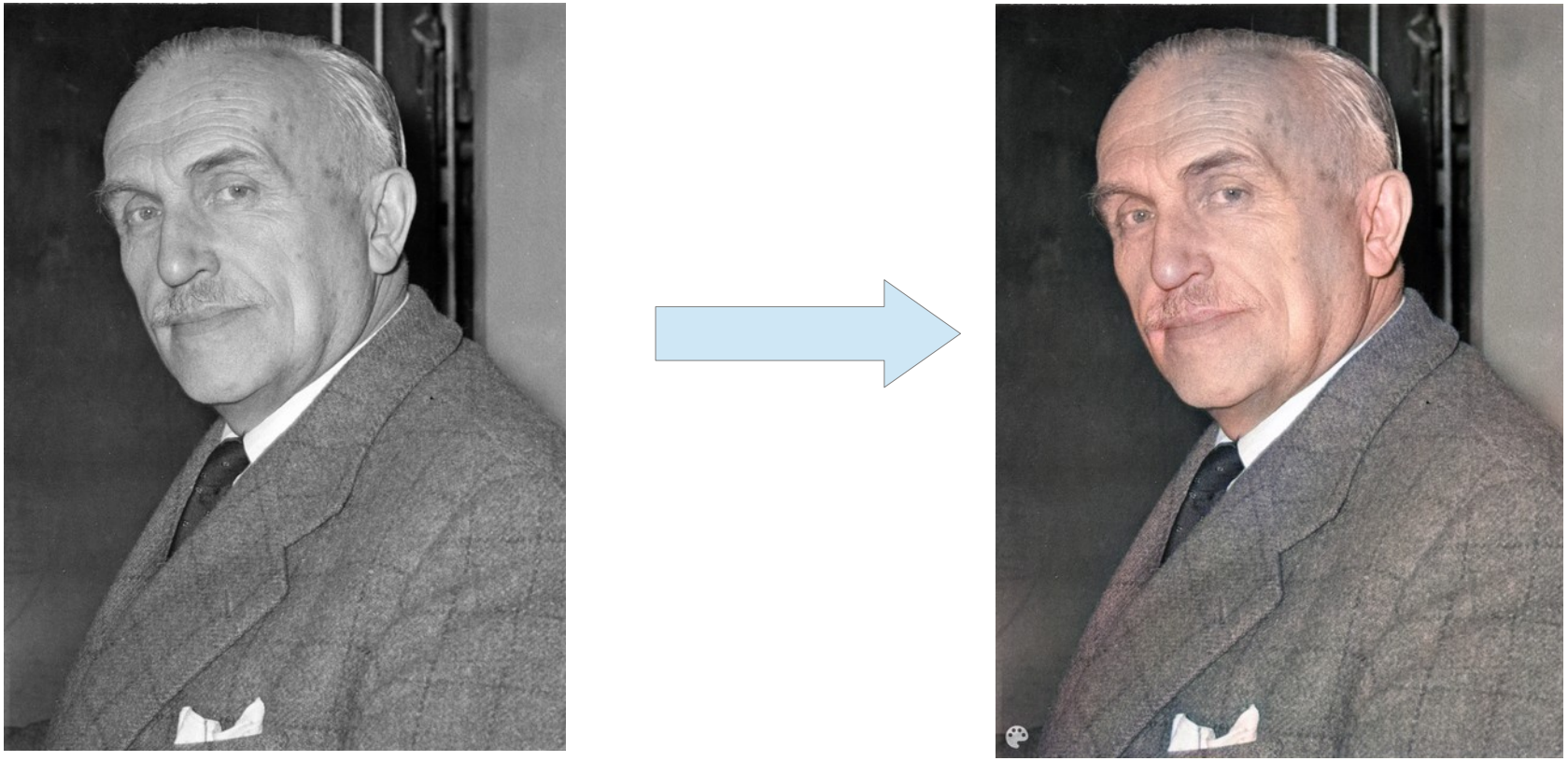
Oryginalne zdjęcie

Generowane twarze po 20 epokach (potrzebujemy więcej epok, żeby wygenerować znośne twarze).

pix2pix

- https://github.com/jantic/DeOldify

  Kolorował zdjęcia.



Pix2Pix

- Pix2Pix demonstruje transformację obrazu w obraz z użyciem Conditional Adversarial Networks. Z użyciem tej techniki możemy dodawać kolor do zdjęć, transformować mapy google'a do „google earth" itp. Technika ta jest opisana w https://arxiv.org/abs/1611.07004
  - https://www.tensorflow.org/tutorials/generative/pix2pix
  - https://github.com/marcinwolter/DeepLearning_2020/blob/main/tensorflow_pix2pix.ipynb

Pix2Pix

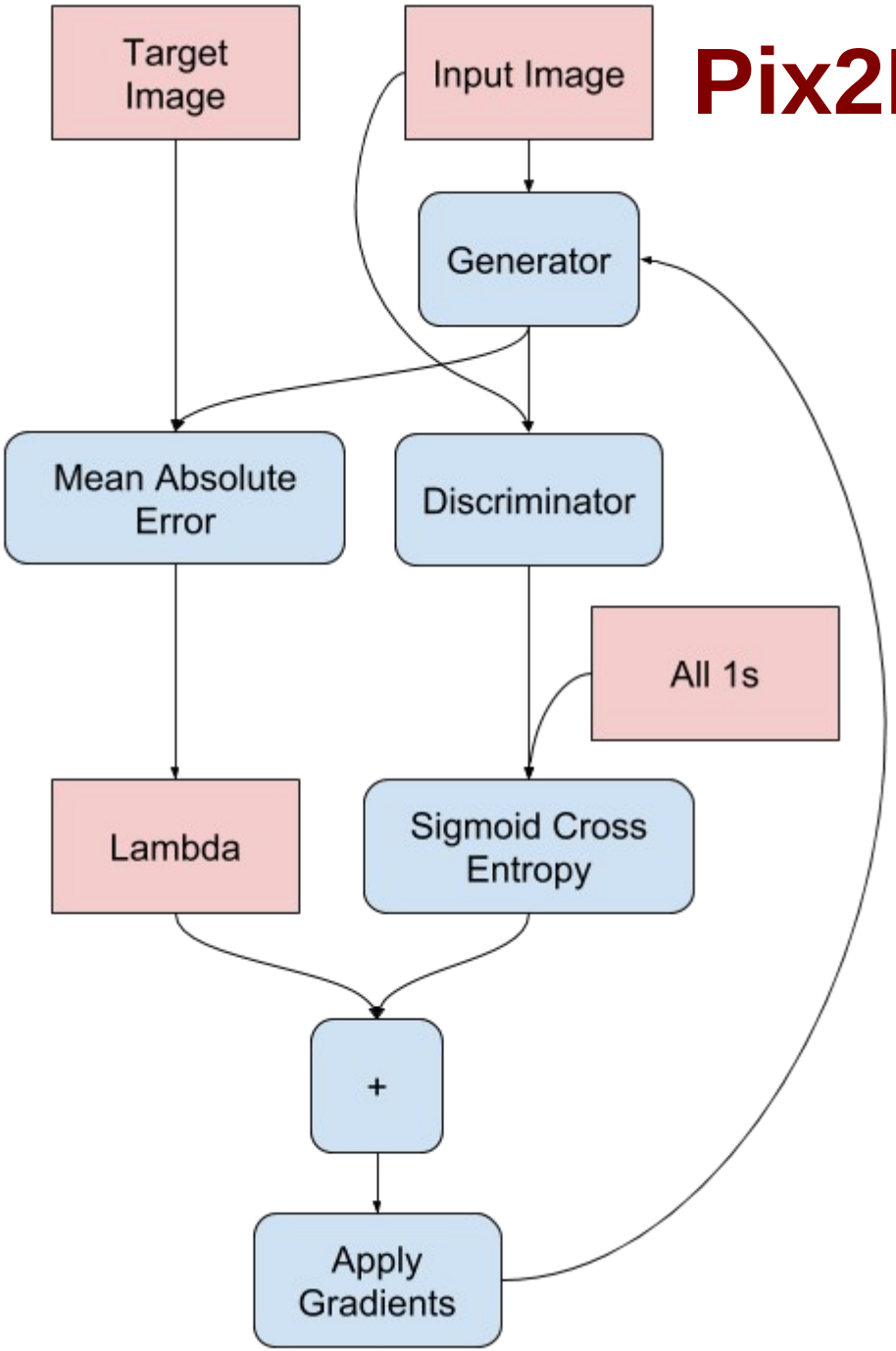


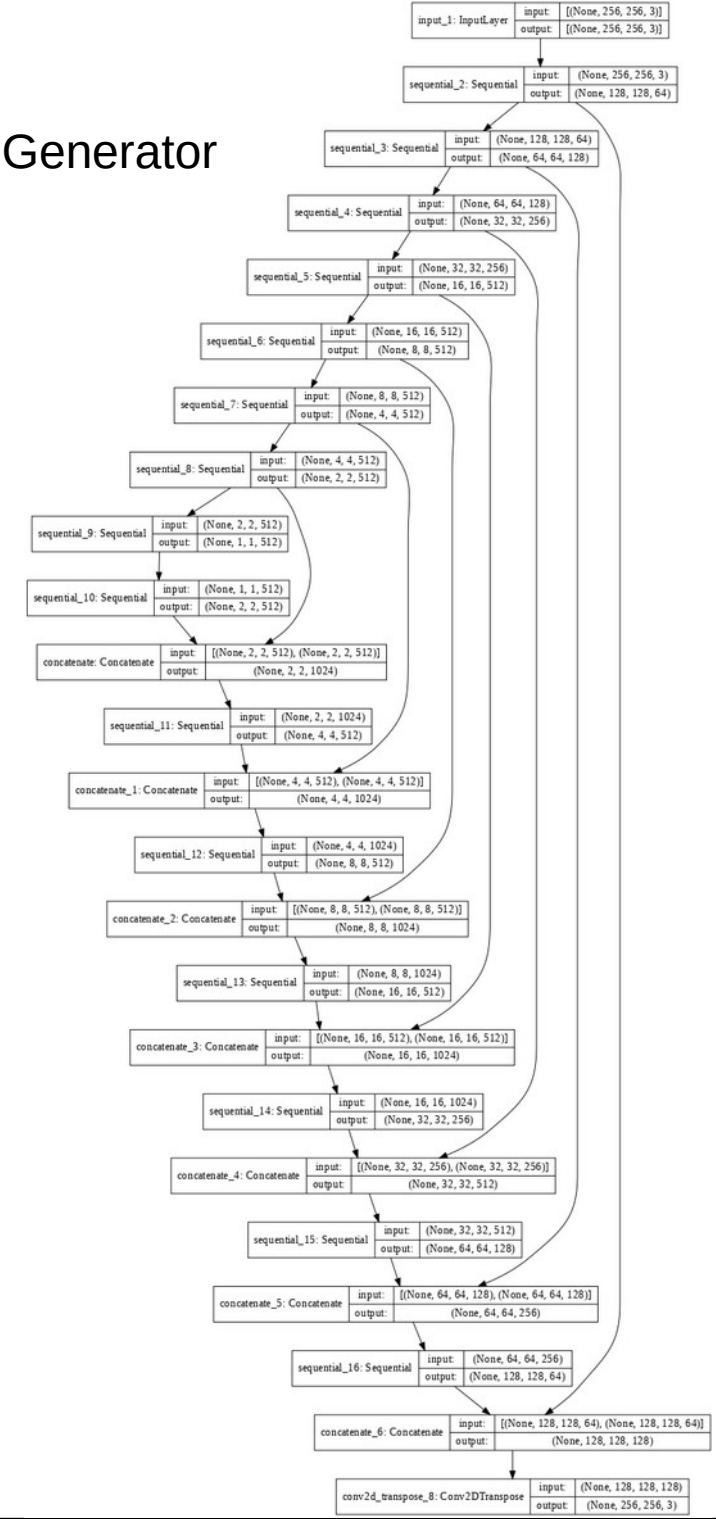
Image-to-Image Translation with Conditional Adversarial Networks

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros

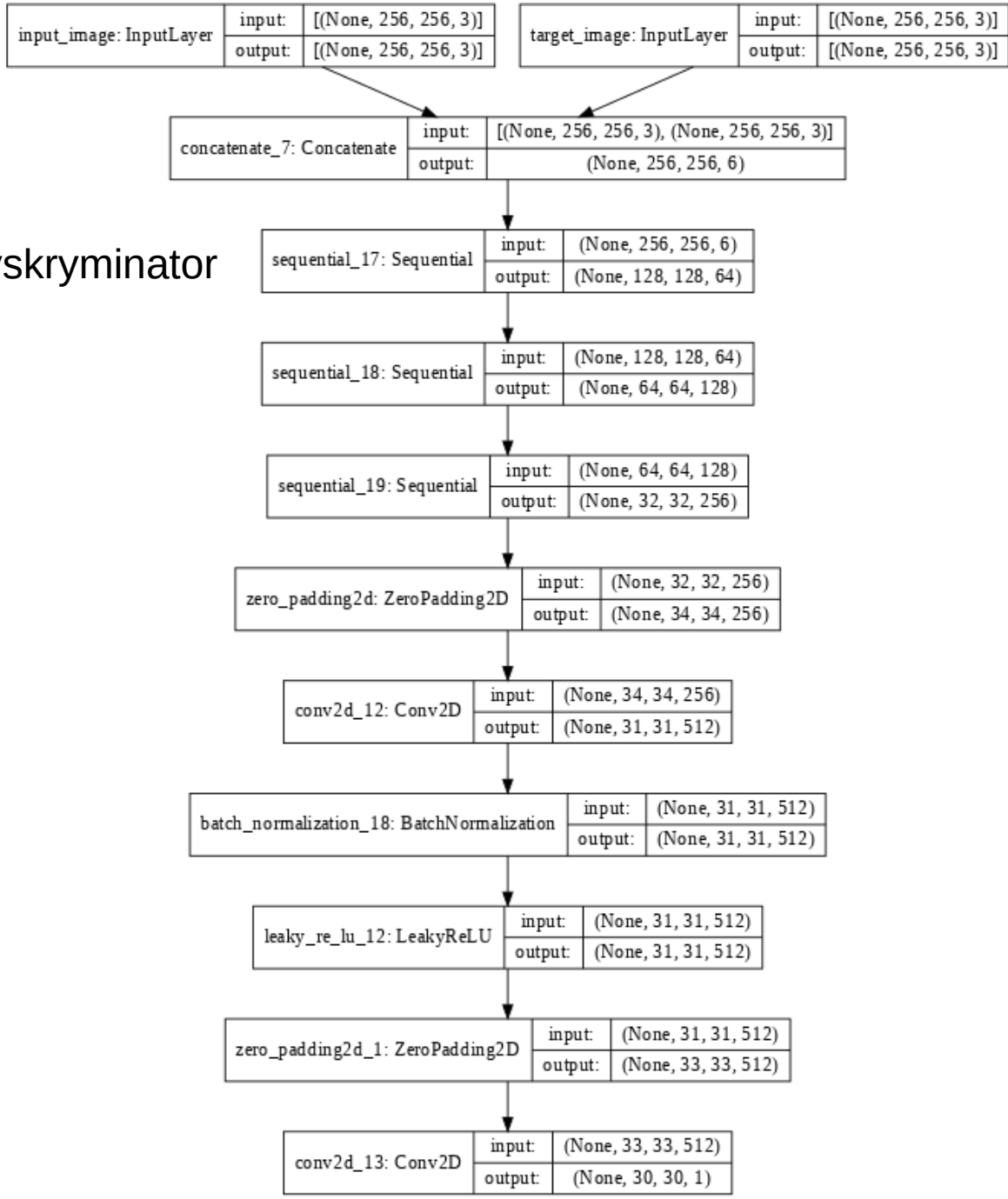
https://arxiv.org/abs/1611.07004

Nice presentation:
https://www.slideshare.net/xavigiro/imagetoimage-translation-with-conditional-adversarial-nets-upc-reading-group

Generator



Dyskryminator



Podsumowanie

- GANs – obiecujące narzędzie do produkcji obrazów / dźwięków / danych podobnych do oryginałów.
- Można nawet zbudować „automatycznego artystę” malującego prawie jak Rembrandt.
- Conditional Adversarial Networks – sprawdza się do zadań typu transformacja obrazu w inny obraz (np. dodanie koloru).
- Stosowany także do celów artystycznych.