cape
Campus Asia Plant Environment innovation

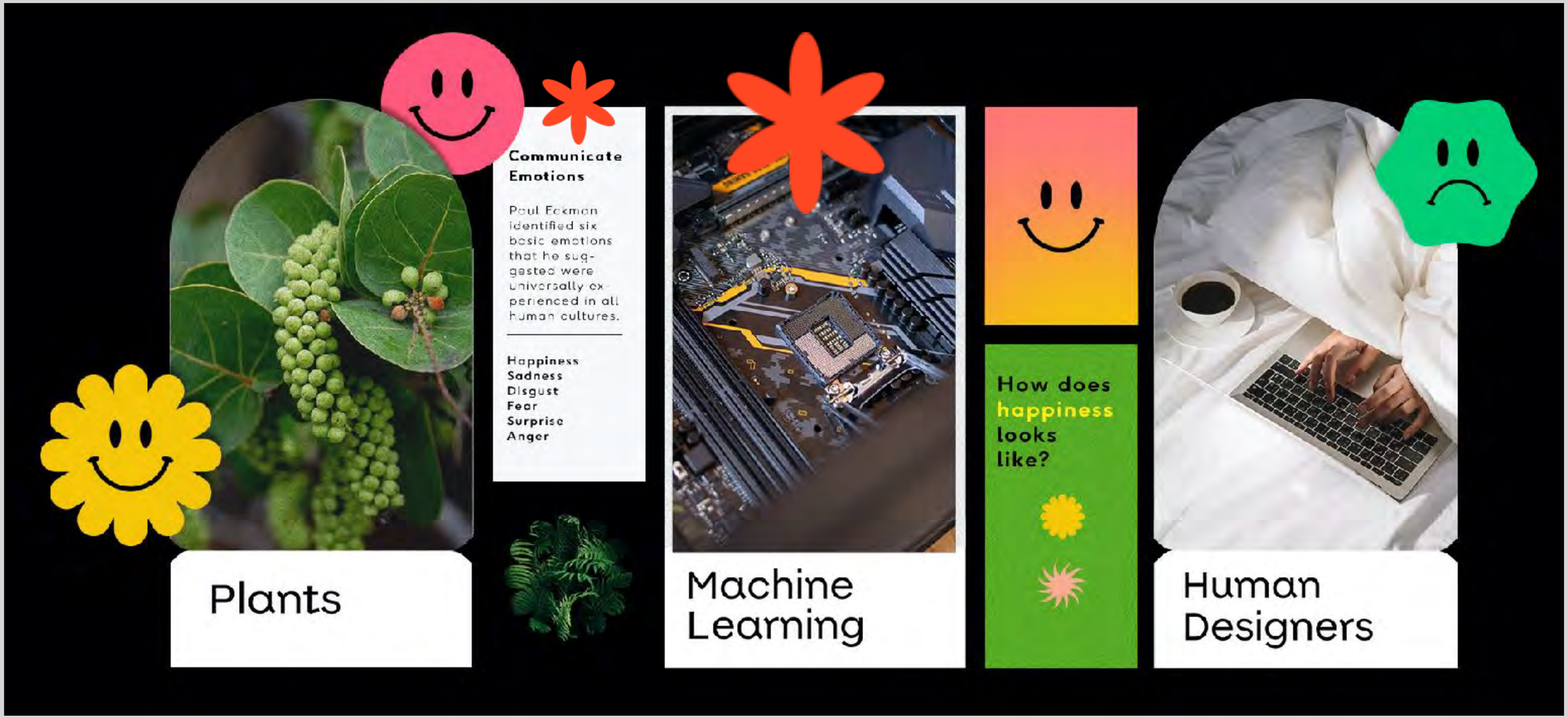
Yonsei University, Zhejiang University & Chiba University

CAPE WS 2021

Generative Design 2

Generative Adversarial Networks

Cape Online Workshop 2021


Communicate Emotions
Paul Ecman identified six basic emotions that he suggested were universally experienced in all human cultures.
Happiness
Sadness
Disgust
Fear
Surprise
Anger
How does happiness looks like?
Plants
Machine Learning
Human Designers

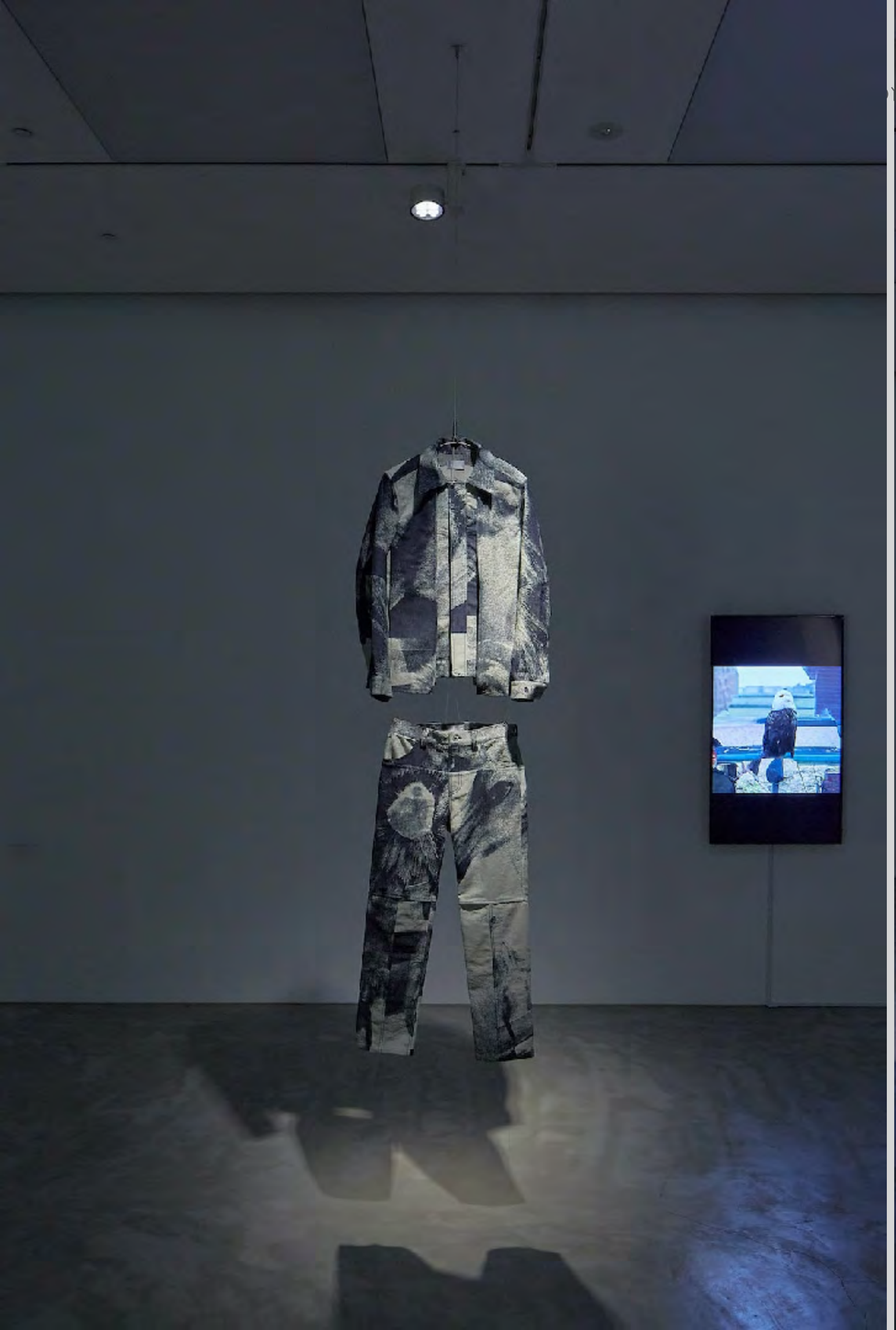
GAN References & uses in Japan

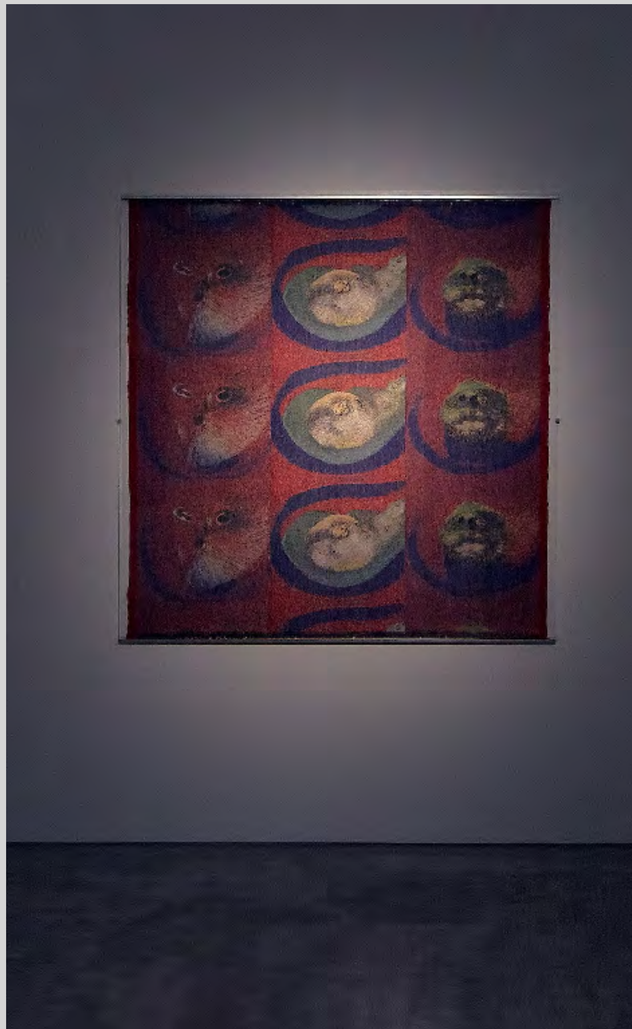
An algorithm-made jacquard denim setup was also created through a fusion of artificial intelligence, fashion and craft.

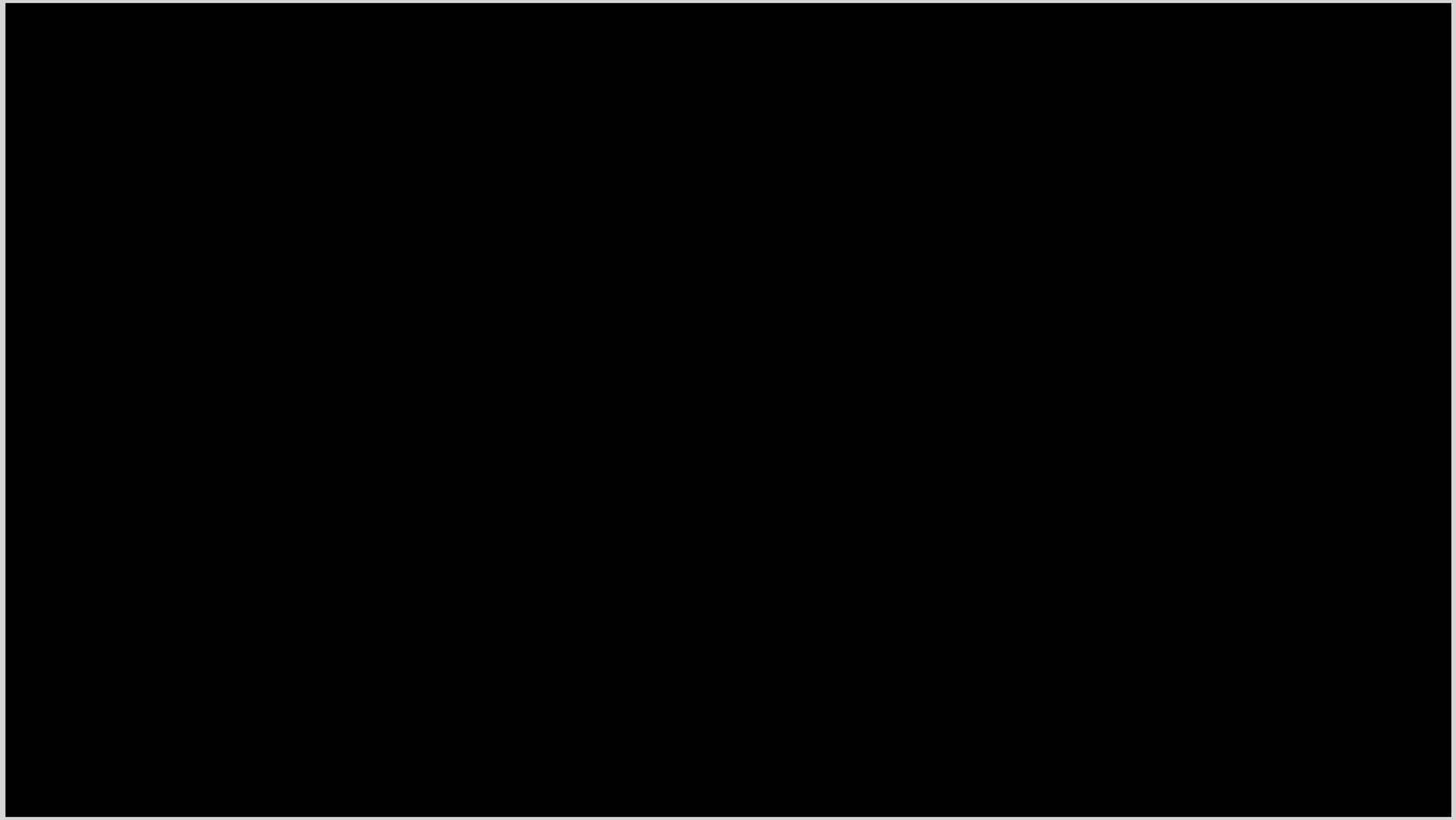
Generative Adversarial Networks (GAN) learned the countless images of animals floating in cyberspace, and the generated data of "imaginary animal patterns"

Credit:

- Project Lead: Kazuya Kawasaki (Synflux)
- Design Lead: Kotaro Sano (Synflux)
- Visual Programing: Ryo Yumoto (Synflux)



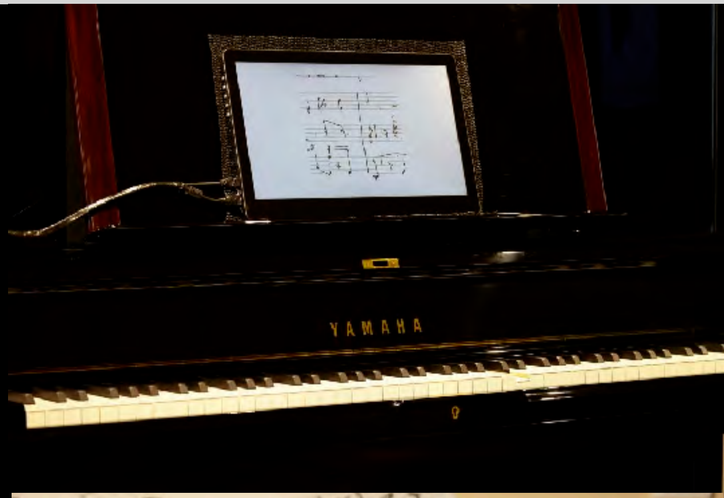


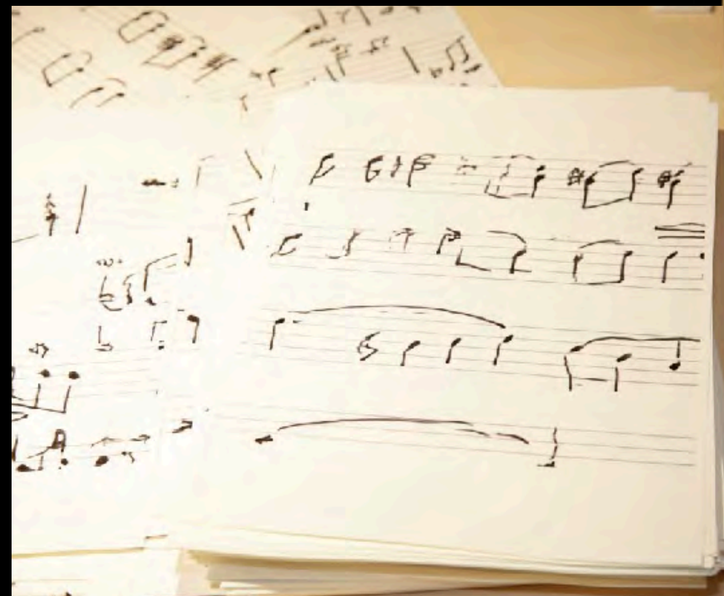


Graphic Design & Co Creation with Artificial Intelligence Exhibition - Design Hub Tokyo Midtown

《作曲をする – 楽譜からうまれるメロディ》Qosmo

https://qosmo.jp/projects/jagda-exhibition/





Manga

Paidon

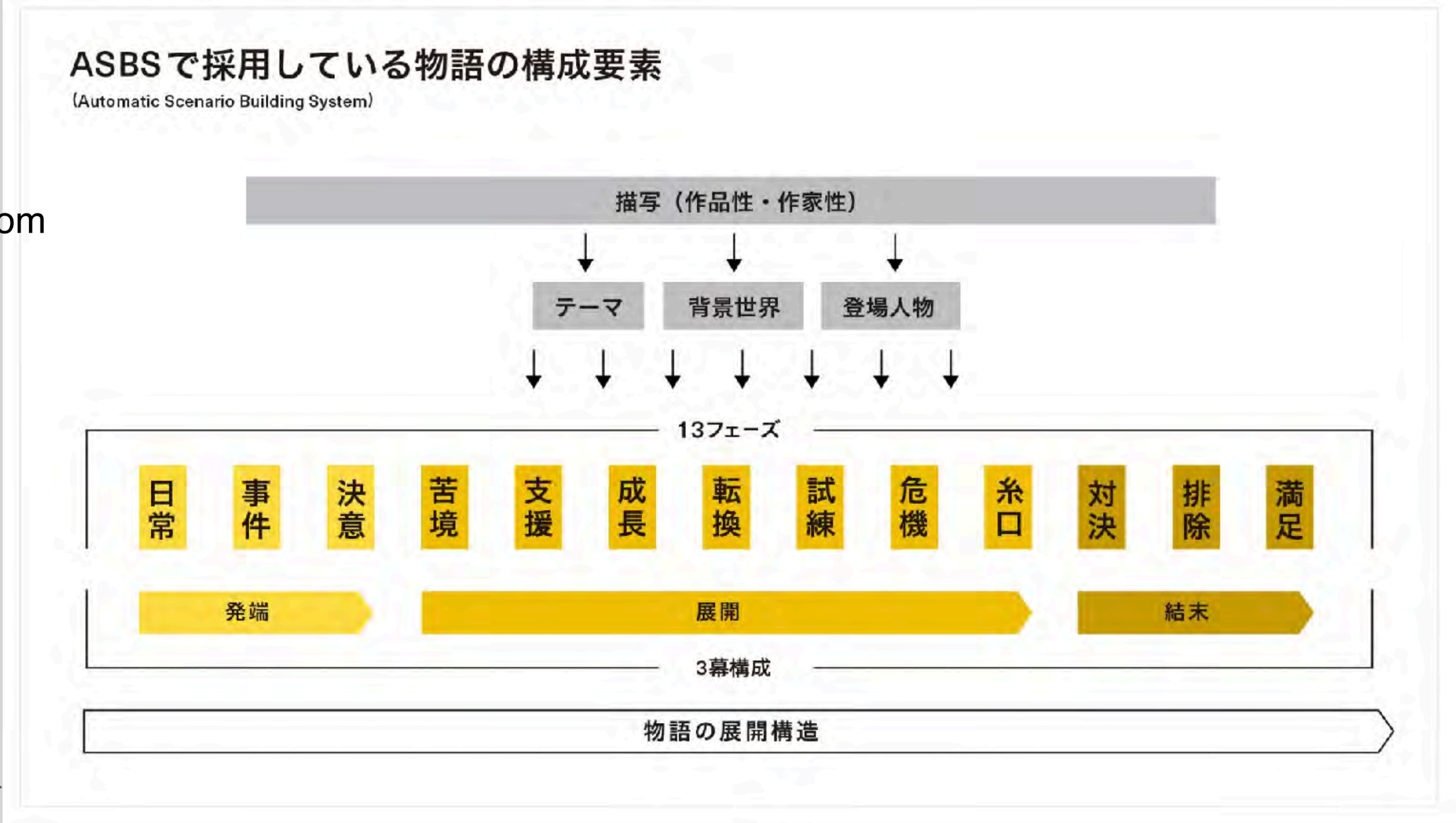
Tezuka2020 Project.
Establish the future of Manga Creation.

Kioxia and Tezuka Productions.
Launched 2020.

https://tezuka2020.kioxia.com

Plot

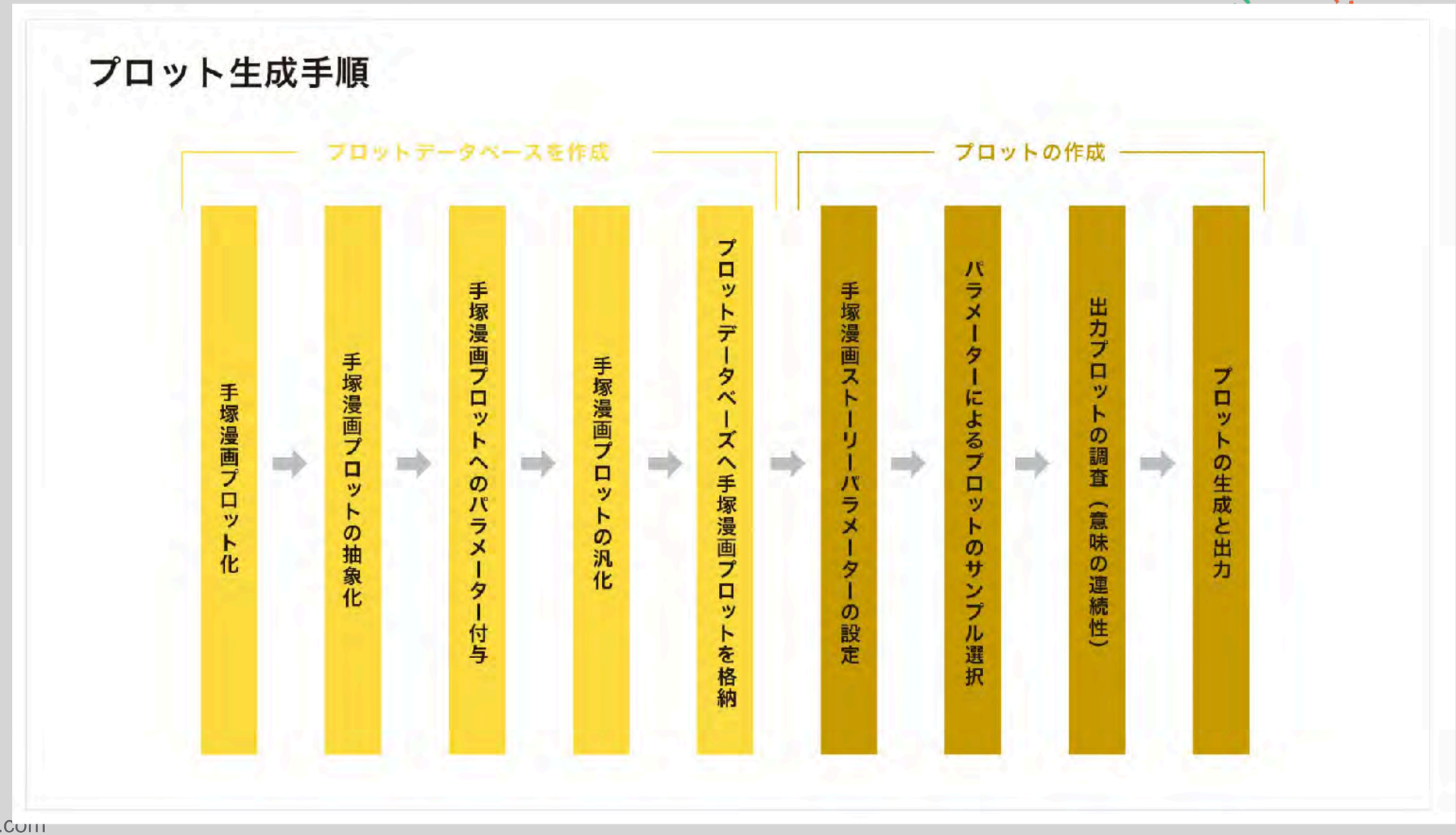
RNN Trained from
130 escenarios



https://tezuka2020.kioxia.

Plot

From Theme to
Plot Estructure



https://tezuka2020.kioxia.com

Script

<物語設定>
【いつ?】現代
【どこで?】街の中
【どんな世界?】現実と架空の狭間の暗いところも明るいところもあるアクションの世界

<主人公設定>
【年齢】少年期
【性別】男
【性格その1】少し明るい
【性格その2】少し強い
【属性】
所属・種族：海人神
属性：偽き
テーマ・職種：二頭の蛇の毛

<基本プロット>
第1幕
主人公は姫の中で人間の生き残りをして生活をしている.
そして、主人公に天界からの贈り物の依頼が出る.
主人公は、この事件に立ち向かうことを決める.
第2幕
事件に足を踏み入れたことにより、医師の試練の指揮を行うこととなる.
様々な動作をもらいこの苦境を乗り越える.
一時的に休息をする主人公.
少しだけ成長した主人公は、偽の心を持ったロボットの改造を行うが、自身が危険な状況となる.
主人公は、解決の最終要を見つけ、擬似的な死を克服する.
第3幕
主人公は、患者の死と対決する、主人公は、この障害を排除する.
主人公は、難題がすべて解決し出産する.

<物語設定>
【いつ?】現代
【どこで?】魔法が使える国
【どんな世界?】現実と架空の狭間の暗いところも明るいところもあるアクションの世界

<主人公設定>
【年齢】少年期
【性別】男
【性格その1】少し明るい
【性格その2】少し強い
【属性】
所属・種族：孔雀
属性：大変
テーマ・職種：海皮膚医＋本業＋医師＋新婚医

<基本プロット>
第1幕
魔法が使える国で暮らす小鳥をしている主人公.
近隣の住む医者との対決が起きる.
その中に足を踏み入れる主人公.
第2幕
父からの依頼で協むことになるが.
様々なヒントをもらい苦境を乗り越える.
主人公は、つかの間の休息をえる.
主人公は、協師としてライバルから子供を正し出す要求に立ち向かう、主人公は、この試練により破滅的な危機に陥る.
試練をどうにか乗り越える主人公.
第3幕
主人公は、協会乗りに挑む、主人公は、この障害を排除することに成功する.
主人公は、難題がすべて解決し宮崎する.

<物語設定>
【いつ?】現代
【どこで?】アフリカ
【どんな世界?】現実と架空の狭間の暗いところも明るいところもあるアクションの世界

<主人公設定>
【年齢】少年期
【性別】男
【性格その1】少し明るい
【性格その2】少し強い
【属性】
所属・種族：教師
属性：やさしい
テーマ・職種：サイボーグ＋妖怪退治

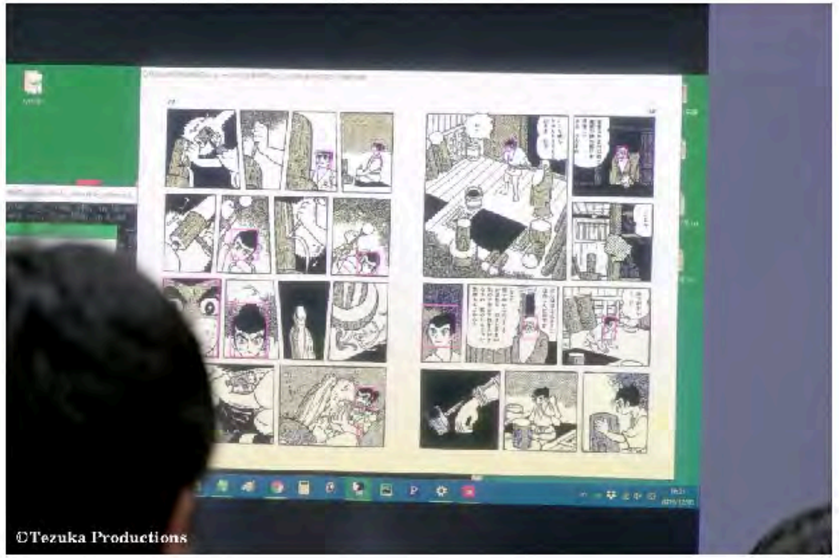
<基本プロット>
第1幕
アフリカで主人公は、進路に悩む学生をしている.
主人公は、悪魔の医への誘惑に巻き込まれる.
主人公は、これからこのことに立ち向かうこととなる.
第2幕
修行で遊む改善の夢を追いふれ、興したが・・・
様々な支援でこの苦境を乗り越える.
主人公は妖気により問題解決の糸口を見つける.
主人公は、安易な解決策の提示チャレンジする、しかし、主人公は、危険的な状況に陥る.
なんとか、機械を倒して、
第3幕
主人公は勇敢に協会である、交通事故の患者の修復と対決する、主人公は、この障害を乗り越えることに成功する.
主人公は、すべての困難を乗り越えて完定を得る

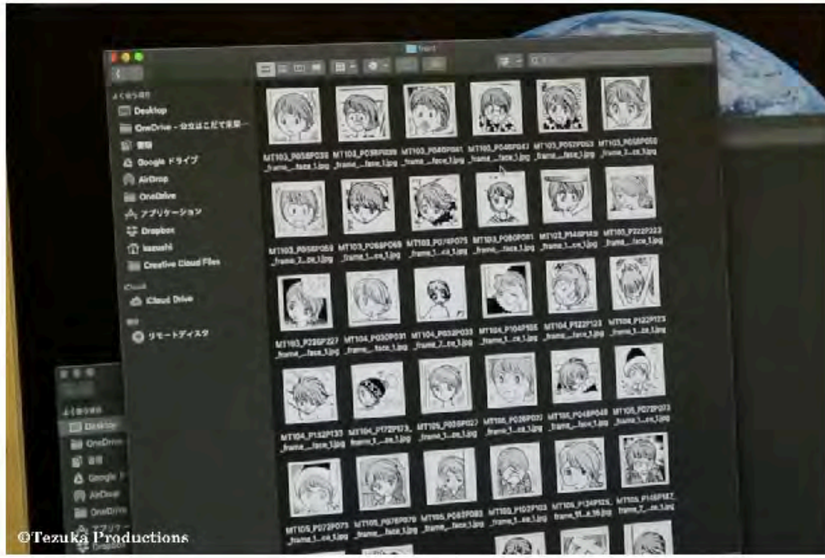
https://tezuka2020.kioxia.com

Character

Generative Adversarial Network

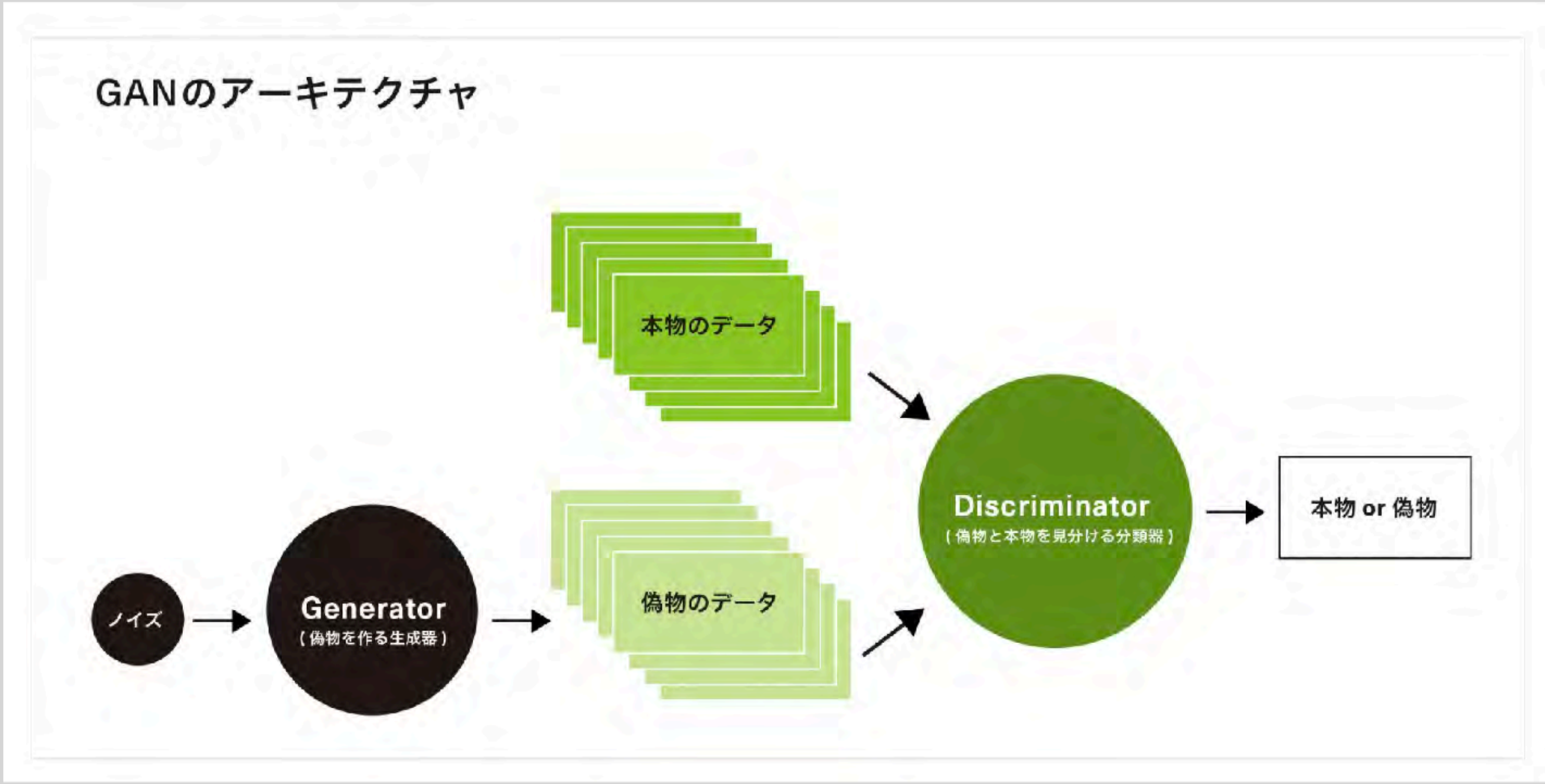
人のサポートなくして成し得ない、
AIの画像認識とは？





Character

GAN Structure



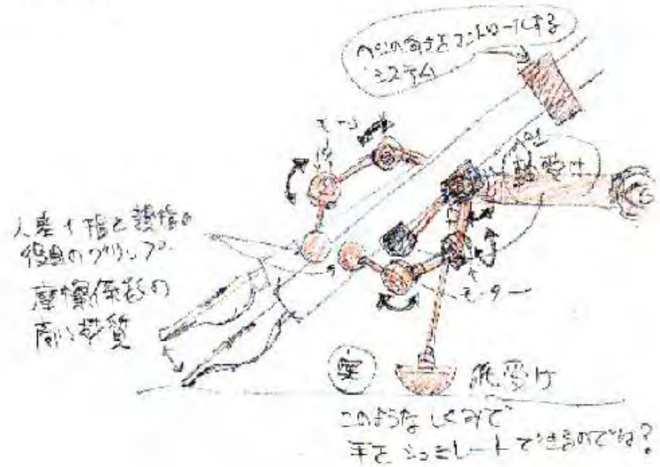
Character

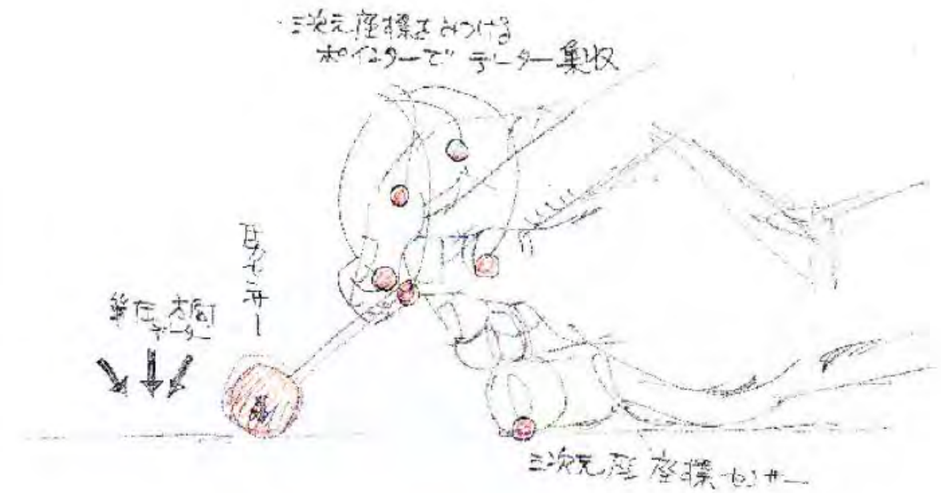
候補画像

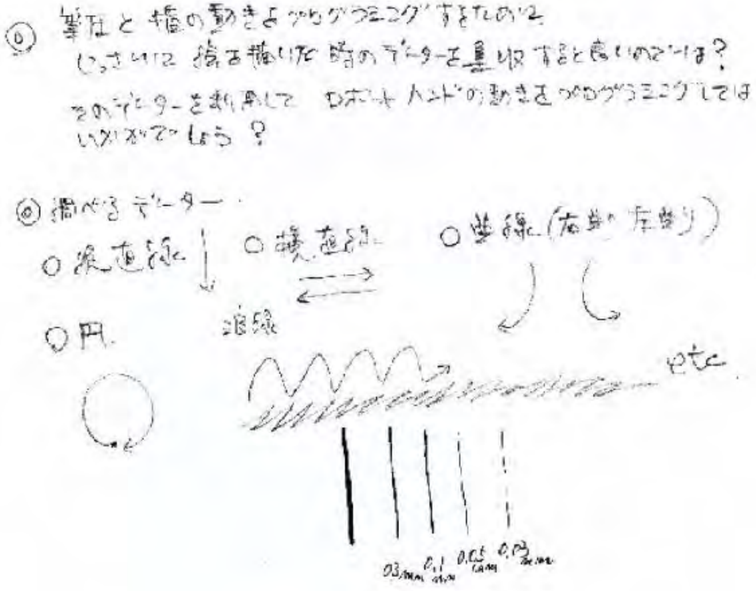
cape
Campus Asia Plant Environment innovation

Drawing

Robot Arm
Pen Drawing





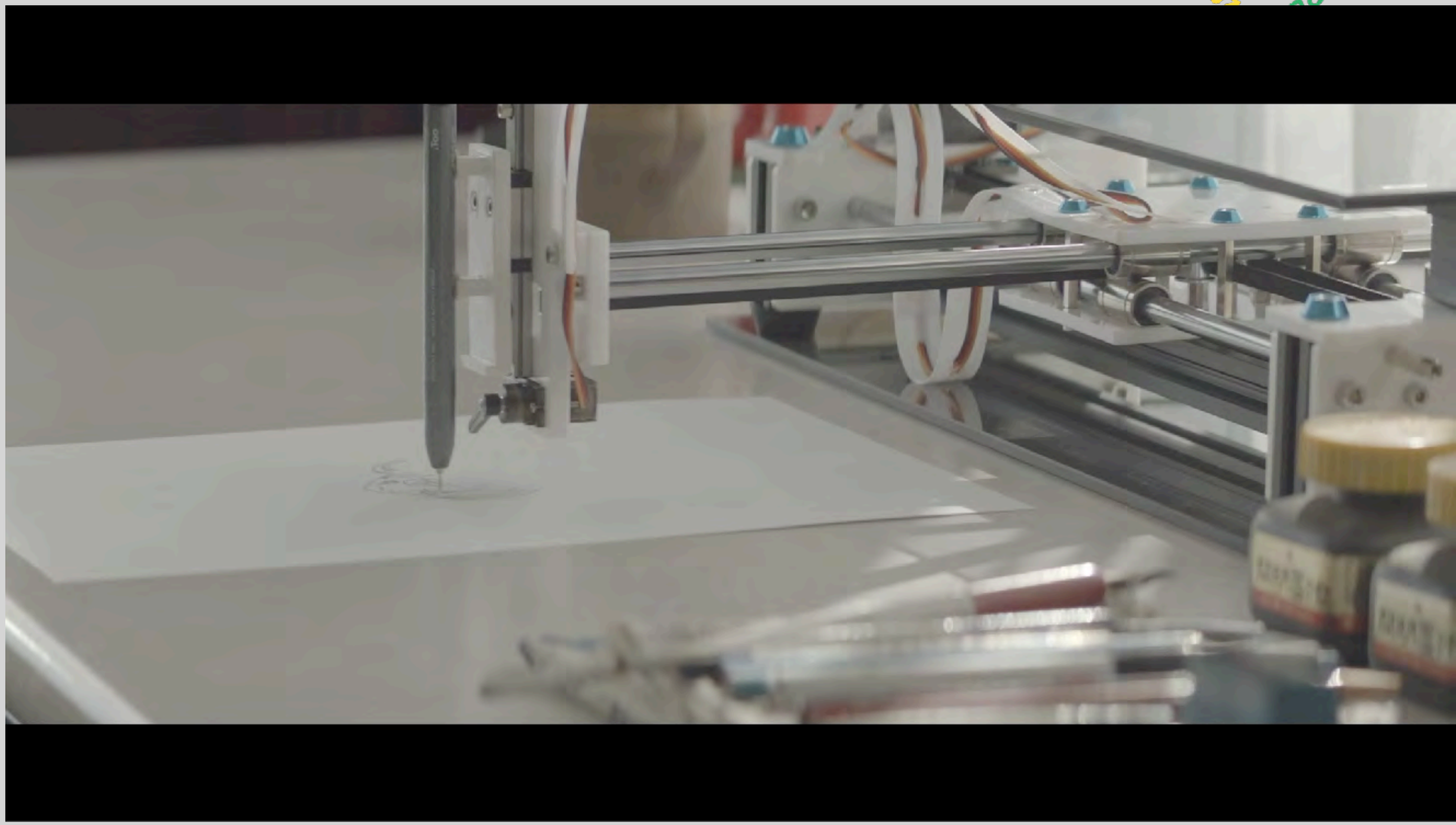


Drawing

Robo Arm
Training

Drawing

Robo Arm
Training Results

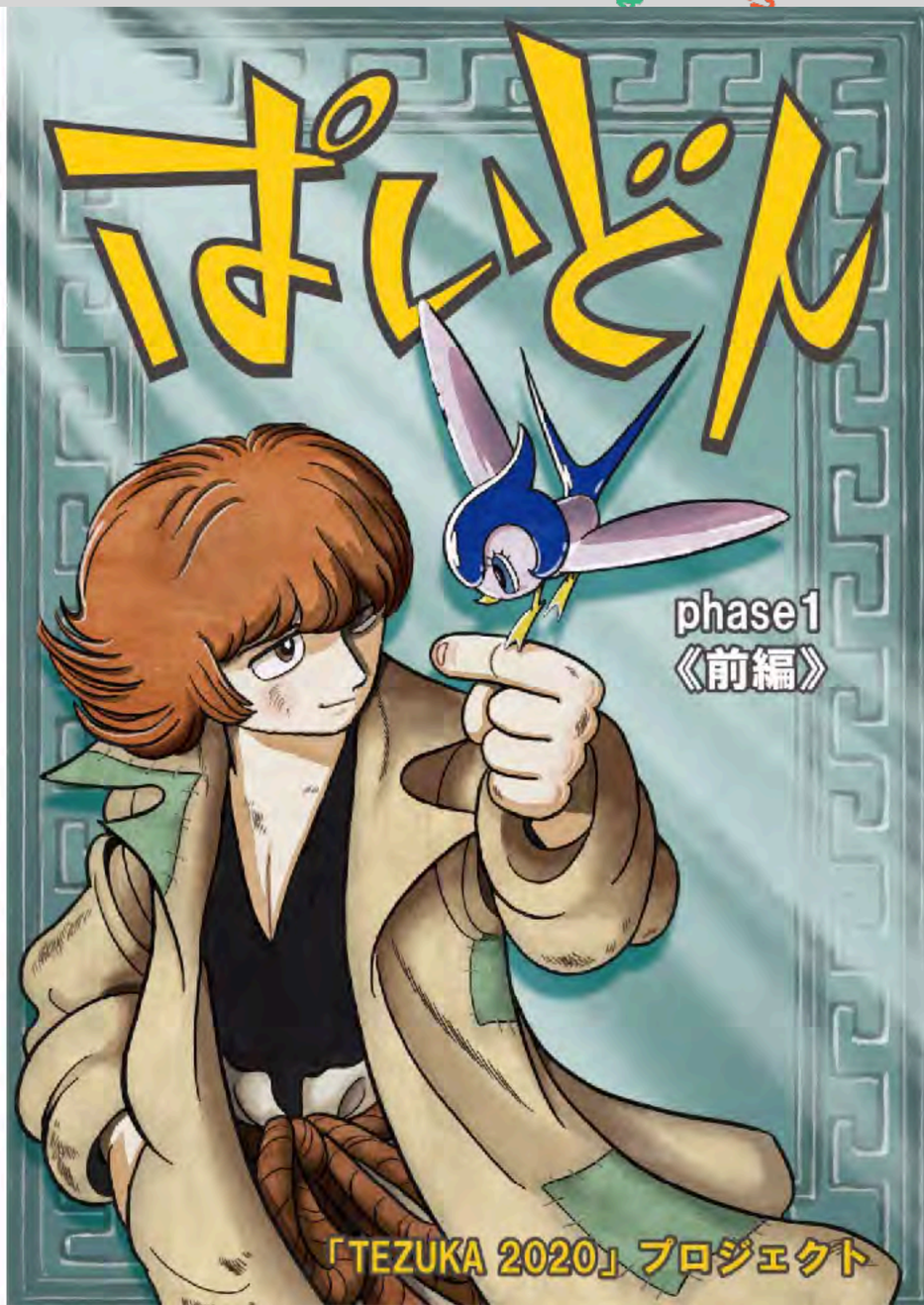


Drawing

Robo Arm
Training Results

Manga

Final.
Published Result



BreakGAN feat. BBOY STEEZ https://www.interactive.thedesignium.com/

【データグリッド】全身モデル自動生成AI | [DataGrid] Model generation AI https://datagrid.co.jp/en/case/

【データグリッド】アイドル自動生成AI　https://datagrid.co.jp/en/case/

Games And Animation

Character Generation

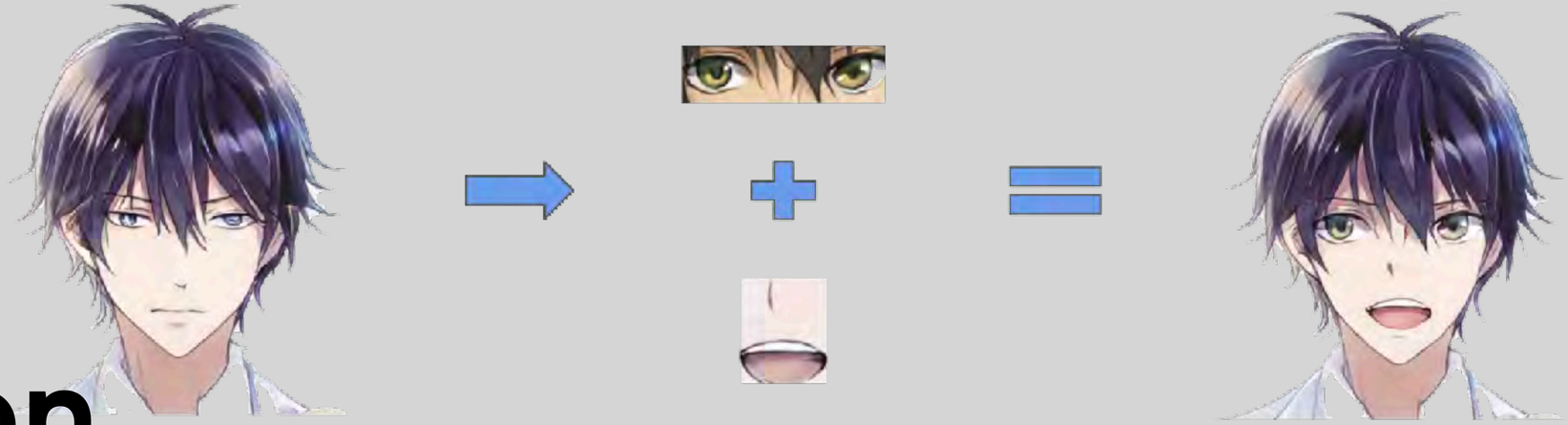
Crypko

2D GAN based character and animation generation.

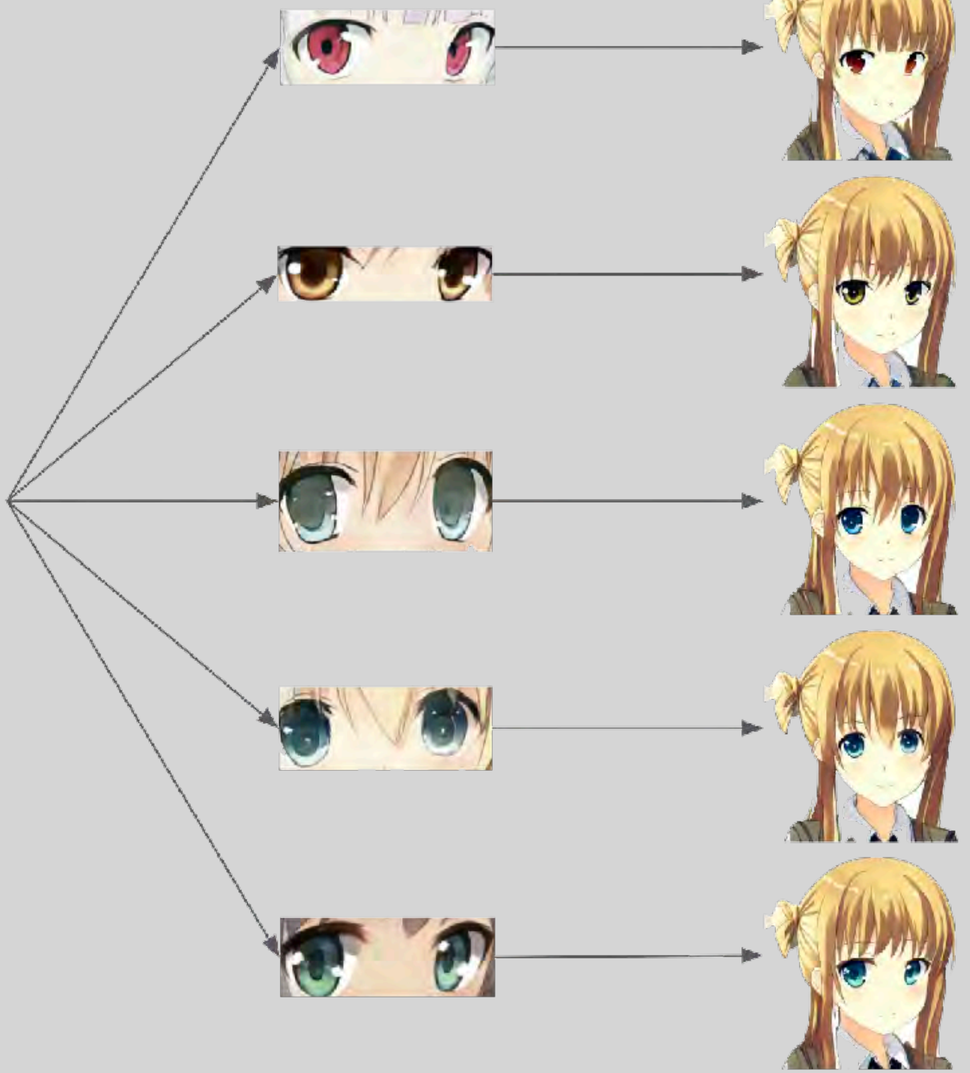
Preferred Networks
2018

https://crypko.ai/

Feature Extraction & Projection



Feature Extraction & Projection



Latent Walk Generation

cape
Campus Asia Plant Environment innovation

Yonse

Latent Walk Animation

CAPE WS 2021

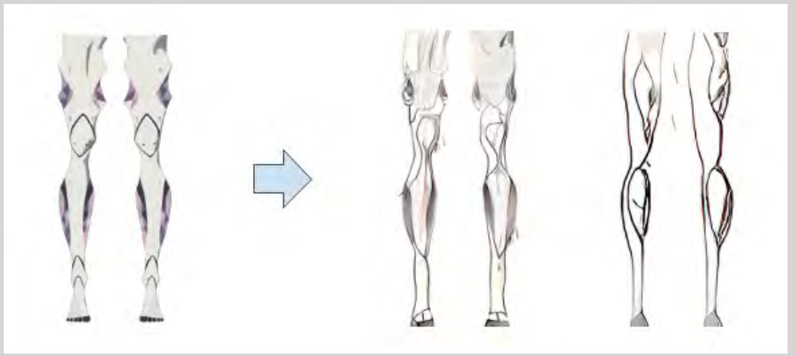

Crypko
生成
My Crypkos
眉毛
口
鼻
まつげ
黒目
白目
服
髪
顔
肌
選択
カーソルサイズ
適用
セーブ
ブラシ
消しゴム
選択
切り取り
カラーバケツ


WHOMOR INC.
Crypko
Crypkos
アニメーションパネル
インタラクティブ
ポジション初期化
メインタイムライン
普通
慌てる
笑顔
全自動
全自動2
動作リセット
微笑
嬉しい
喜ぶ
大好き
さらに表示
差分タイムライン
歩行、走行、登場、待機など
差分タイムラインを選択してください

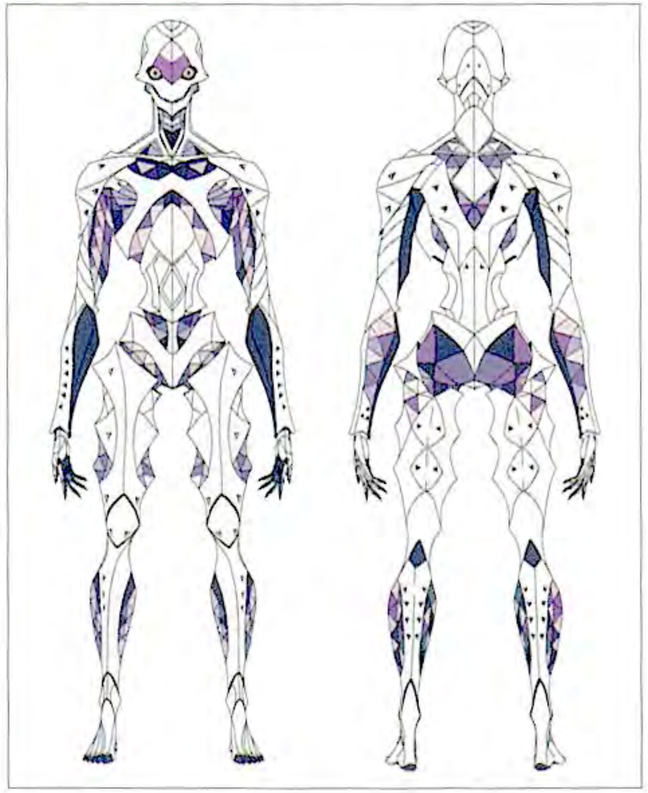
DesignChainer

GAN + Convolutional Neural Network Model based on Chainer for character generation.

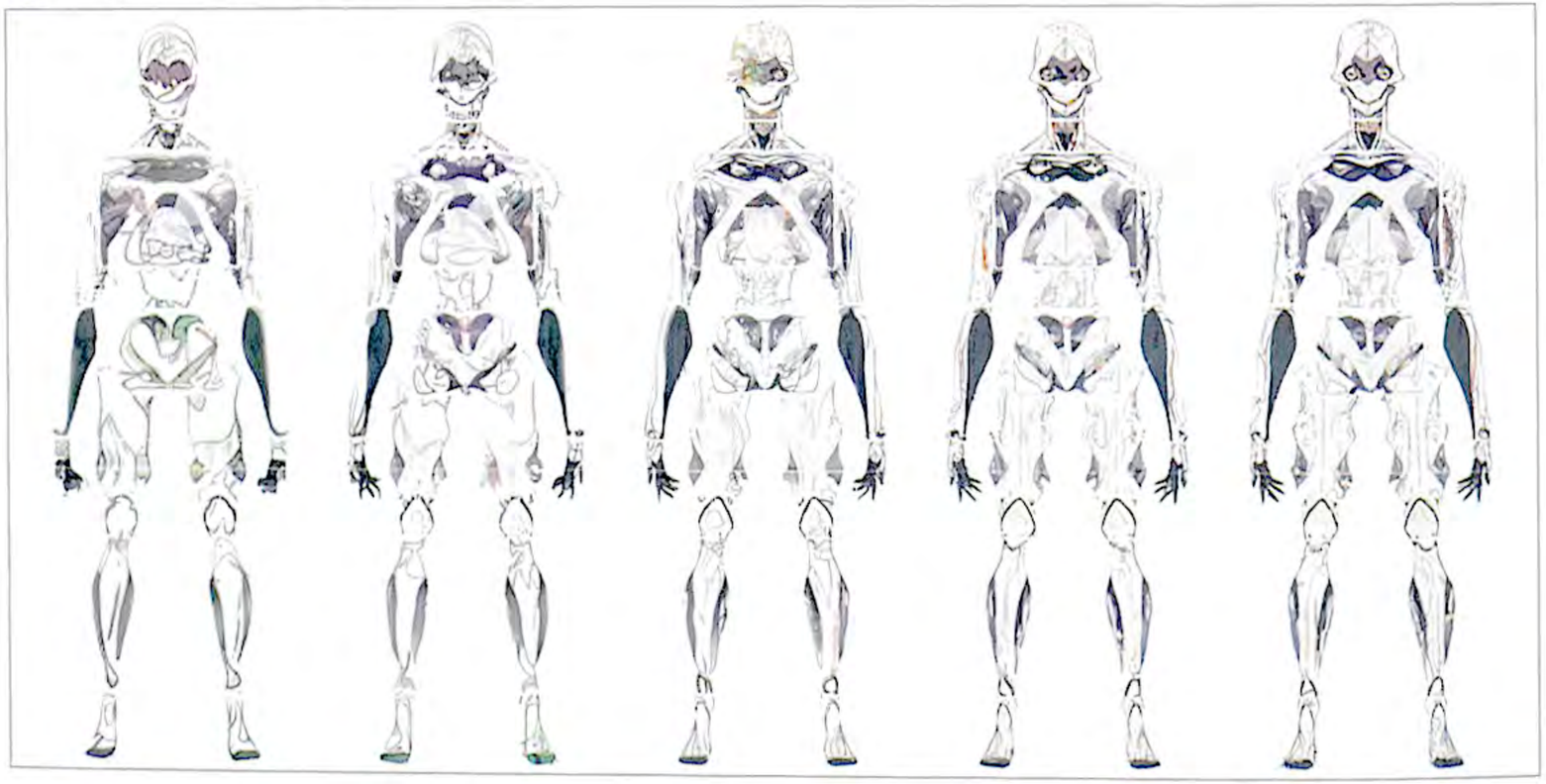
Preferred Networks.
Launched 2018.



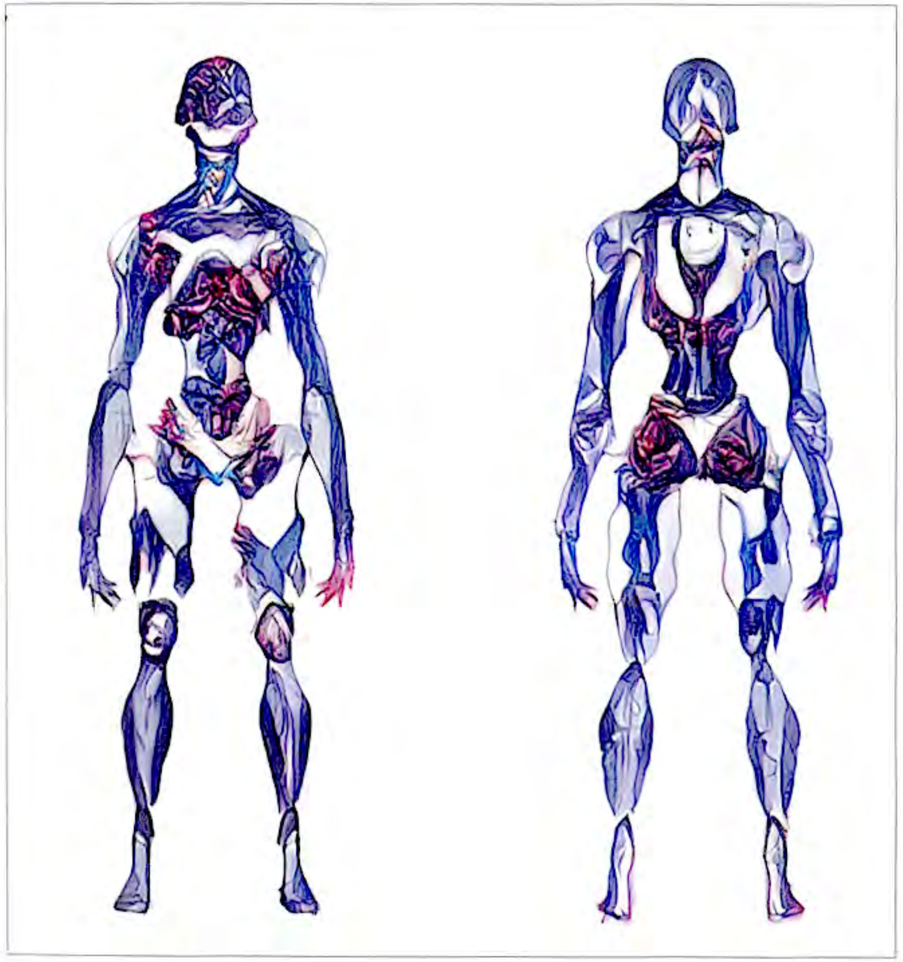
A アルマの3Dモデル（レンダリング）



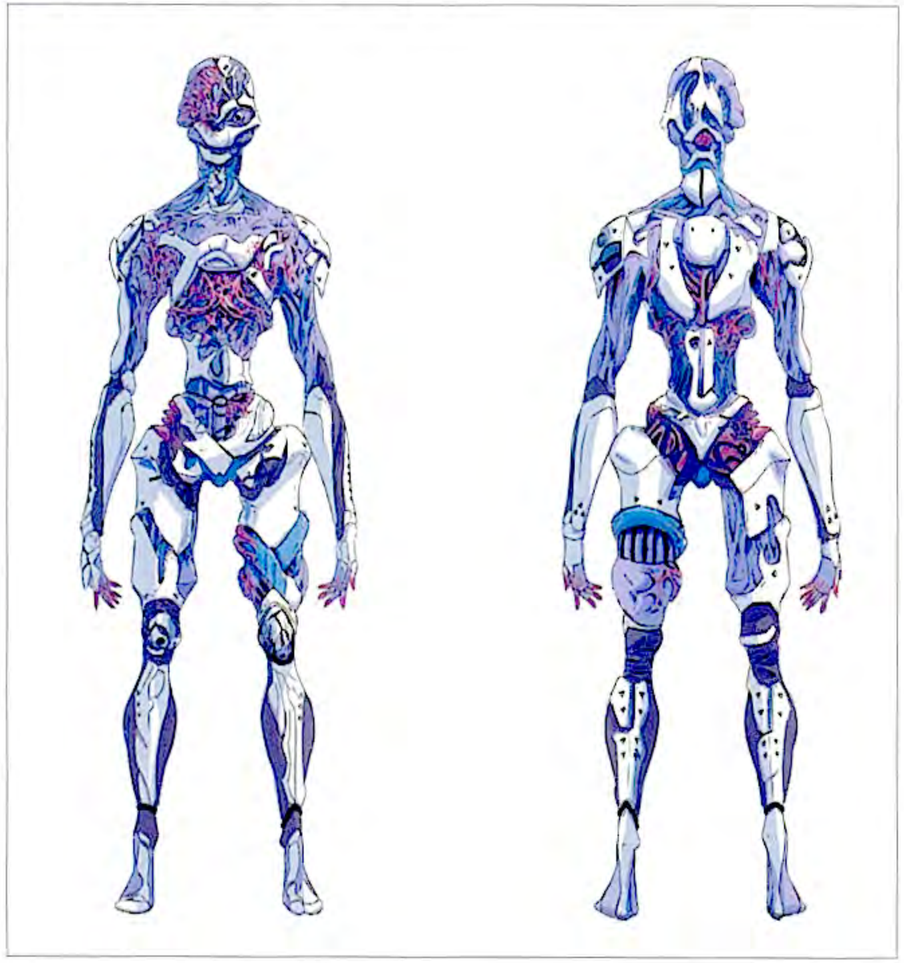
B DesignChainerが描いたデザイン（学習段階）

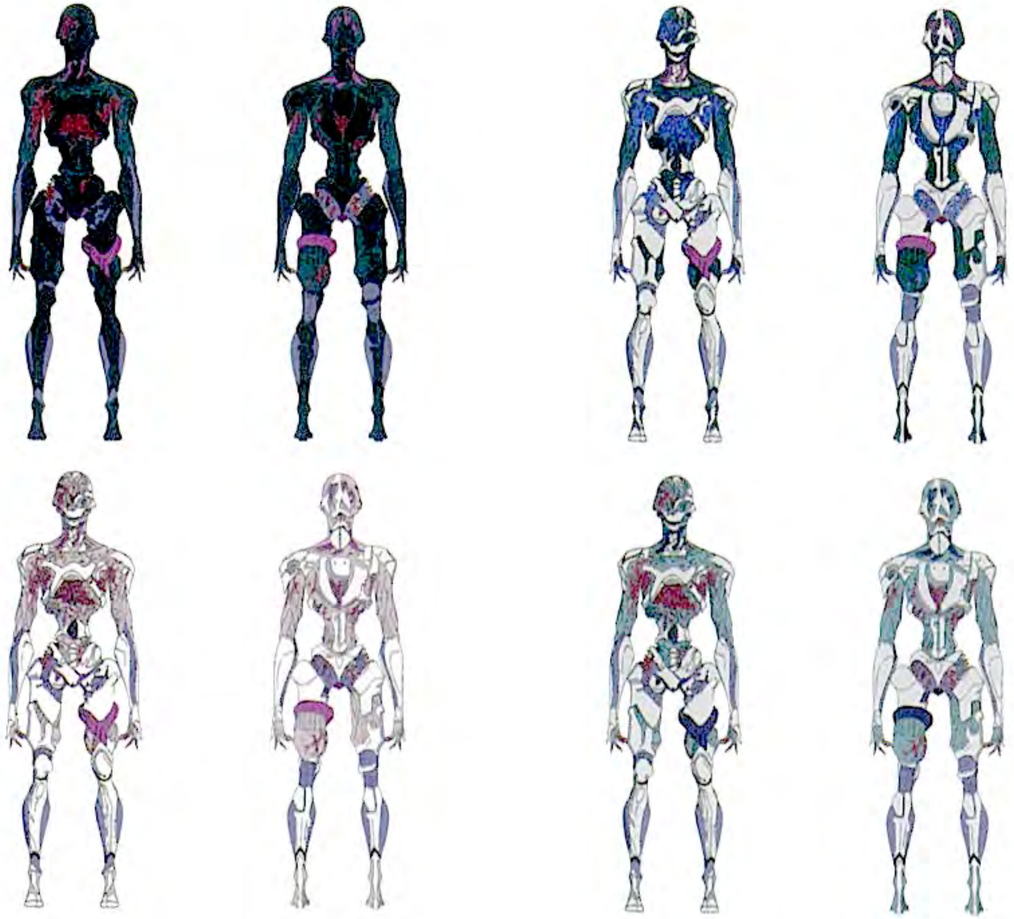


A アルマティックのデザイン原案

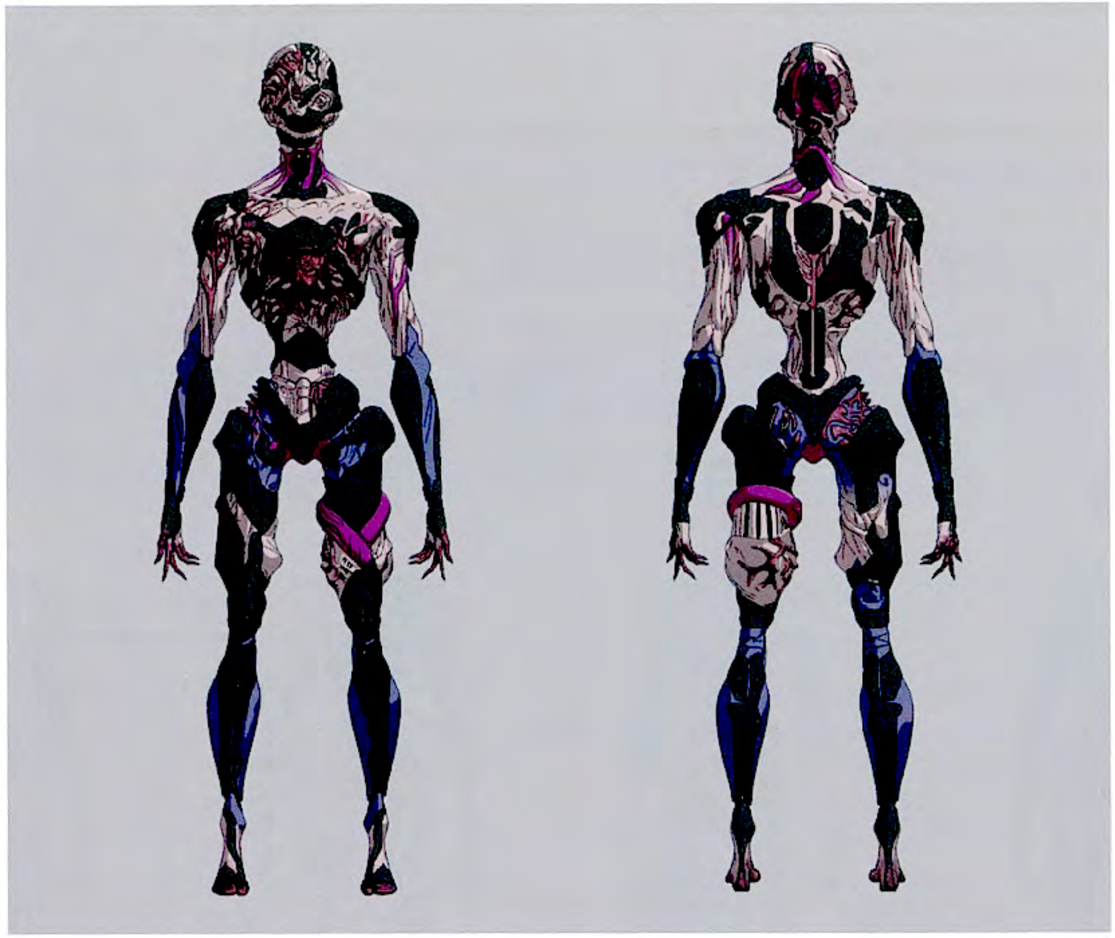


B アルマティックのキャラクターデザイン





Ⓓ アルマティックの完成モデル



cape
Campus Asia Plant Environment innovation

Yonsei University, Zhejiang University & Chiba University

CAPE WS 2021 CAPE WS 2021

MotionChainer

Bipedal Crowd Motion Enhanced by Neural Network.


Tentara baru, ALMATIC, tak pedulikan nyawa.

cape
Campus Asia Plant Environment innovation

Yonsei University, Zhejia

PSGAN Character

Progressive Structure-conditional GANs for character generation.

DENA.
Launched 2018.


pose sequence

YuruGAN: Yuru-Chara Mascot Generator Using Generative Adversarial Networks With Clustering Small Dataset

Yuki Hagiwara
Department of Electrical and Electronic Engineering
Tokyo University of Agriculture and Technology
2-24-16, Nakacho, Koganei-shi, Tokyo, 184-8588, Japan
Email: hagiwara19@sip.tuat.ac.jp

Toshihiko Tanaka
Department of Electrical and Electronic Engineering
Tokyo University of Agriculture and Technology
2-24-16, Nakacho, Koganei-shi, Tokyo, 184-8588, Japan
Email: tanaka@cc.tuat.ac.jp

Abstract—A *yuru-chara* is a mascot character created by local governments and companies for publicizing information on areas and products. Because it takes various costs to create a *yuru-chara*, the utilization of machine learning techniques such as generative adversarial networks (GANs) can be expected. In recent years, it has been reported that the use of class conditions in a dataset for GANs training stabilizes learning and improves the quality of the generated images. However, it is difficult to apply class conditional GANs when the amount of original data is small and when a clear class is not given, such as a *yuru-chara* image. In this paper, we propose a class conditional GAN based on clustering and data augmentation. Specifically, first, we performed clustering based on K-means++ on the *yuru-chara* image dataset and converted it into a class conditional dataset. Next, data augmentation was performed on the class conditional dataset so that the amount of data was increased five times. In addition, we built a model that incorporates ResBlock and self-attention into a network based on class conditional GAN and trained the class conditional *yuru-chara* dataset. As a result of evaluating the generated images, the effect on the generated images by the difference of the clustering method was confirmed.

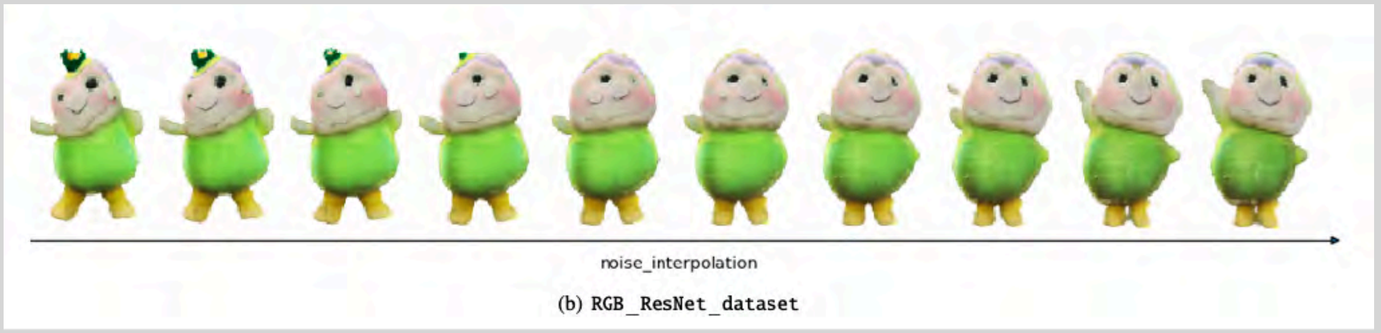
I. Introduction

A *yuru-chara* is a "unusual character" originating from Japan. It is typically created by local governments and companies for the purpose of promoting events, regional revitalization, products, and so forth. The number of *yuru-charas* is increasing year by year, and an annual "the Yuru-chara Grand Prix" ranks entries by the popularity of these *yuru-charas*. The economic effects produced by *yuru-chara* are significant. For example, the Kumamoto prefecture's *yuru-chara* "Kumamon," which won the Yuru-chara Grand Prix 2011", has generated an economic effect of about 124.4 billion yen thanks to its popularity throughout the country [1]. This is one of the incentives for local governments and companies to create characters that can be ranked high in "the Yuru-chara Grand Prix." Regarding the relationship between *yuru-chara* and popularity, Nakasato et al. analyzed the characters that participated in "the Yuru-chara Grand Prix" and the corresponding rankings using machine learning and found that the design and colors of *yuru-chara* greatly contributed to their popularity [2].

Because of copyrights and creator fees, it costs a lot to create a new *yuru-chara*. Spending money and time to design a *yuru-chara* does not guarantee its popularity. Therefore, by automating the design of *yuru-charas*, a cost reduction can be expected.

A promising way to realize automatic image generation by machine learning is the generative adversarial networks (GANs) method proposed by Goodfellow et al. [3]. GANs can generate images that are almost indistinguishable from training data statistically and have succeeded in generating realistic images compared with other image generative models. Furthermore, various improved versions have been proposed to improve the learning of GANs. The conditional GAN proposed by Mirza et al. reports that by giving class information to a model, a phenomenon called mode collapse [3], [4] can be suppressed, and the quality of generated images can be improved [5]. Depending on the tuning of the model, mode collapse is a phenomenon in which a model generates only the same image. Miyato et al. also proposed a model called cGAN with projection discrimination, reporting that by devising how to give class information to the model, learning was stable and mode collapse in class units could be suppressed [6]. In the self-attention GAN (SA-GAN) proposed by Zhang et al., a self-attention layer was introduced into the GAN model proposed by Miyato et al. [7], and this has successfully created high-quality images.

However, these GAN-related studies use large datasets such as ImageNet [8] and CIFAR-10 [9], as shown in Table I, which are designed for machine learning research. When it comes to the datasets used in these studies, all images are preliminarily labeled with class information [8]. In the current paper, *yuru-chara* images are used as training data, however, a total of 10,435 *yuru-charas* participated in "the Yuru-chara Grand Prix" from 2011 to 2015 (3,908 if entries from multiple years are excluded), so the amount of data is very small compared with the research datasets, and no class label is given to the images of *yuru-chara*. Therefore, there is no guarantee that



(b) RGB_ResNet_dataset

YuruGAN: Yuru-Chara Mascot Generator Using Generative Adversarial Networks With Clustering Small Dataset

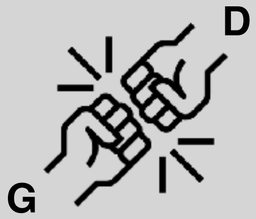
Yuki Hagiwara, Tokyo University

Yesterday: **GAN**

Generative

Generates
Fake Data

Adversarial

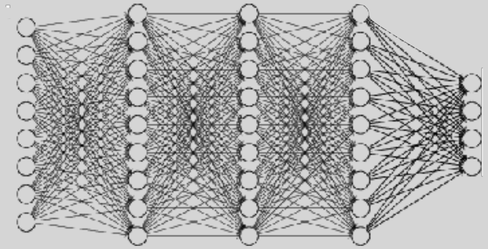


Generator & Discriminator
competing to win

Generator: Work Hard to fake

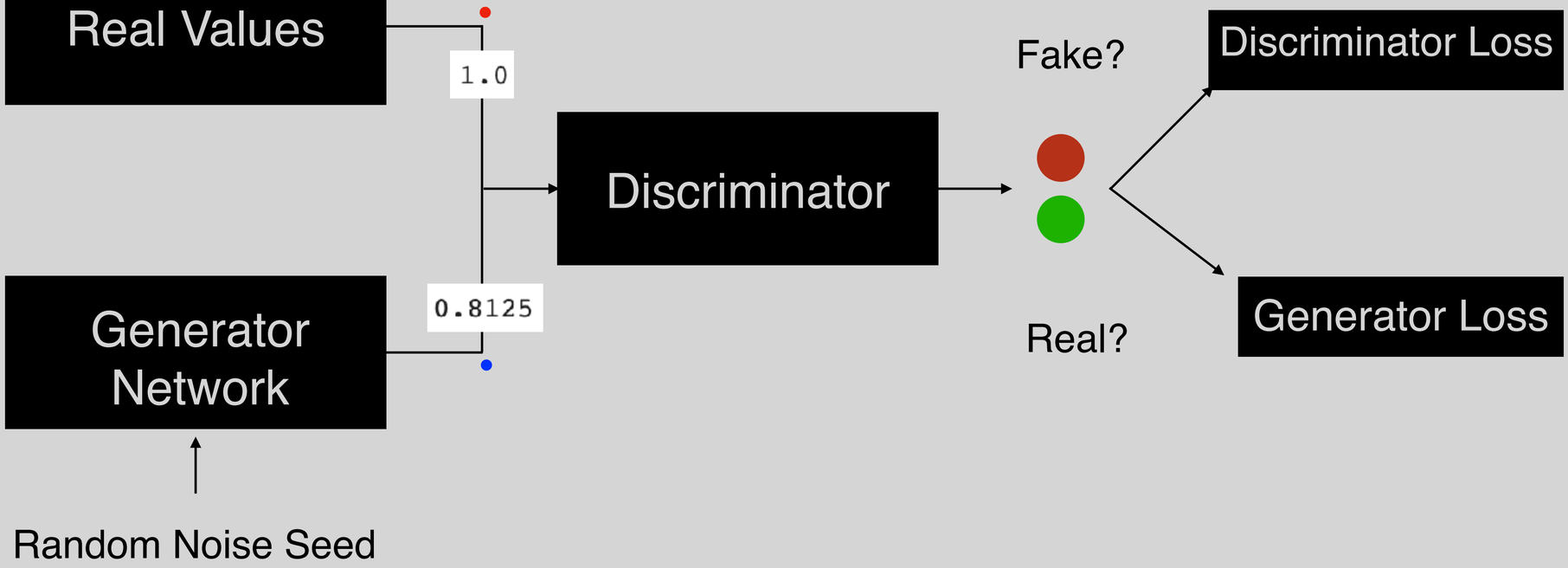
Discriminator: Work Hard to
not be fooled

Networks



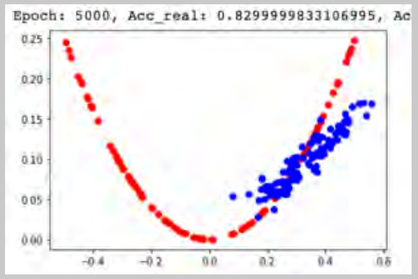
Fully Connected

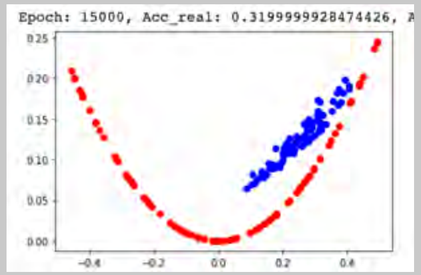
Exercise 1

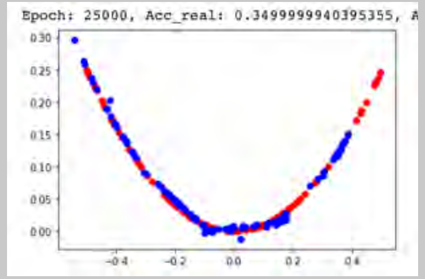


Training

Visualization







Basic

Concepts

Numpy Array: A multi dimensional array.
A grid of indexed values, all of the same type.

Tensor: Numpy arrays, also called tensors.

Epoch: One Epoch is when an ENTIRE dataset is passed forward and backward through the neural network only ONCE.

Batch Size: Total number of training examples present in a single batch.

Note: Batch size and number of batches are two different things.

Batch: The number of sets or parts in which the data set is divided. We can not pass the entire dataset into the neural net at once. That's why you need to divide dataset into Number of Batches or sets or parts.

Iterations: Is the number of batches needed to complete one epoch.

Libraries

Numpy: is a Python library used for working with arrays. It is an open source project and you can use it freely.

Tensorflow: Is an open source library for numerical computation and large-scale machine learning. TensorFlow bundles together a slew of machine learning and deep learning models and algorithms and makes them useful by way of a common metaphor.

Keras: Keras is a deep learning API written in Python, running on top of the machine learning platform TensorFlow. It was developed with a focus on enabling fast experimentation.

Convolutional Neural Networks

Convolutional Neural Networks

(**CNN**, or ConvNet)

Is a class of deep **neural networks**, most commonly applied to analyzing **visual imagery.**

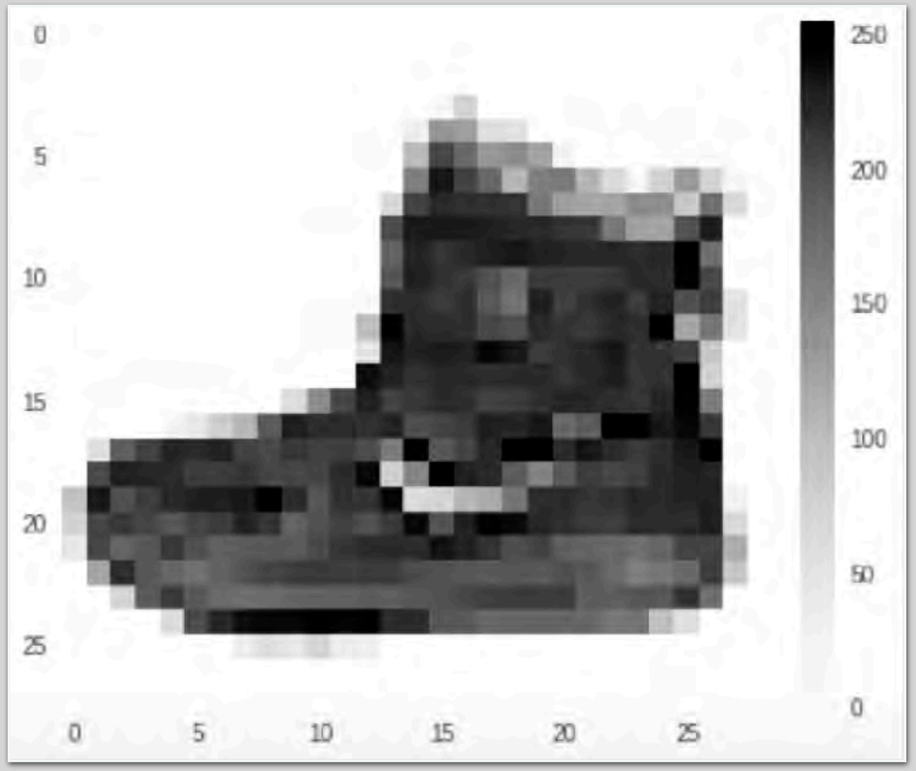
(also be used for other data analysis)

CNN

Convolutional Neural Networks

Artificial neural network that has some type of specialization for being able to pick out or detect patterns and make sense of them

Pattern detection makes CNN useful for **image analysis**


0
5
10
15
20
25
0
5
10
15
20
25
250
200
150
100
50
0

CNN

CNN & Filters

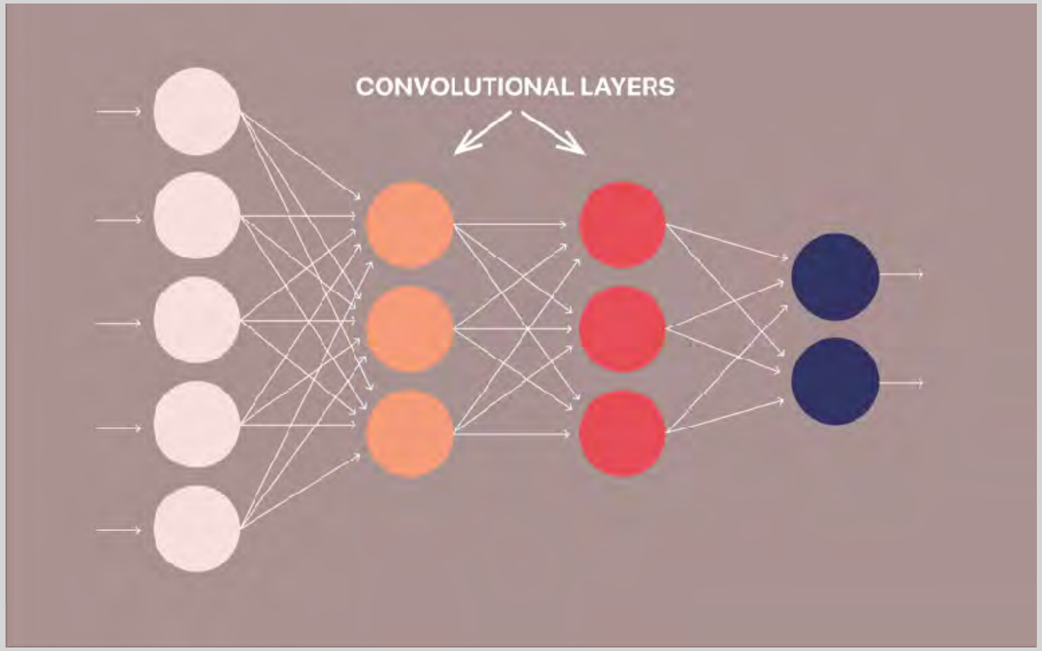
Convolutional neural network
will filter the images before
training the deep neural network.

After filtering the images, features within the images could then come to the forefront and then we can identify something.

Filters

CNN has convolutional layers

For each convolutional layer we need to specify the number of filters



What is a Filter?

A set of multipliers

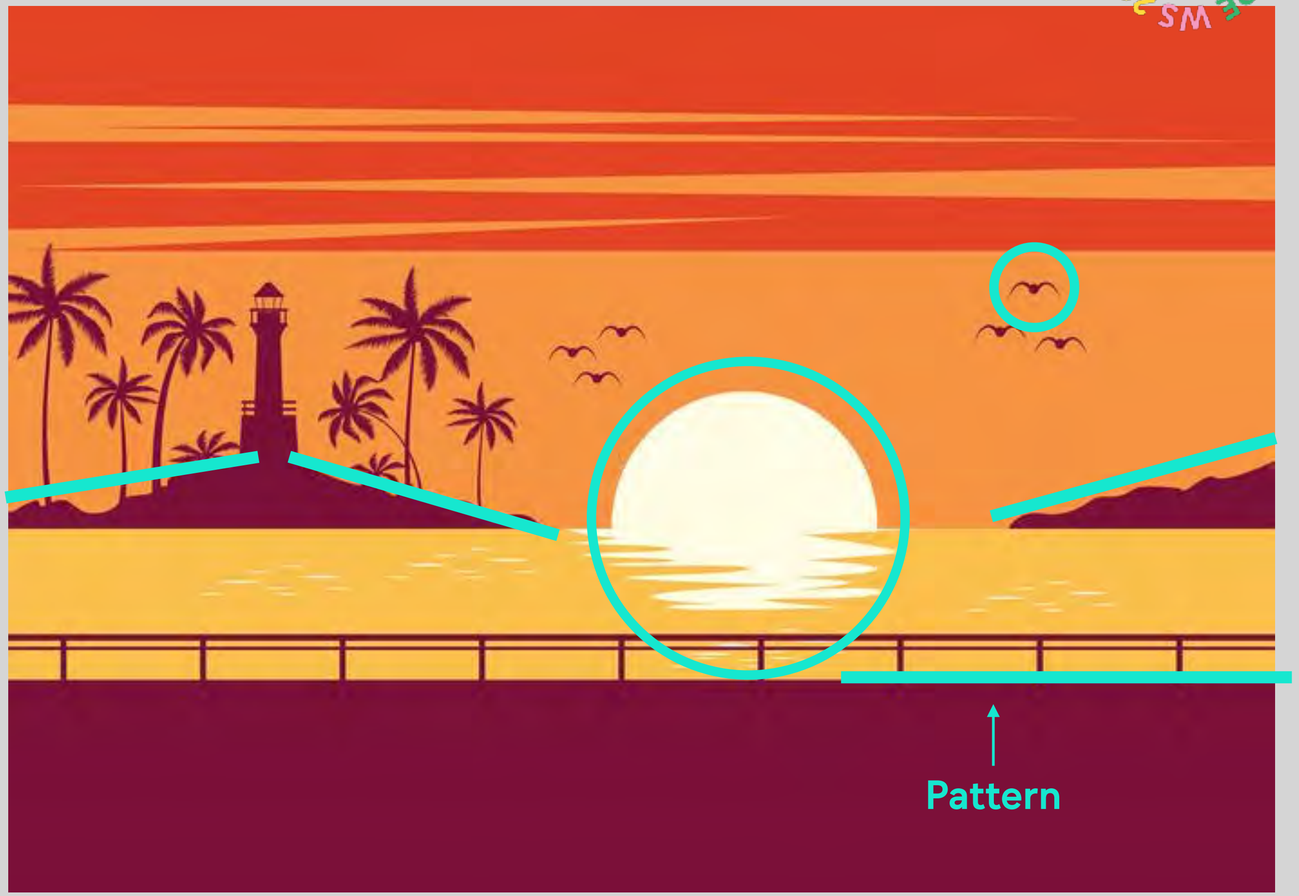
Detect spatial patterns such as edges in an image by detecting the changes in intensity values of the image

Detect multiple edges, shapes, textures, objects. This filter can be called: Edge Detector

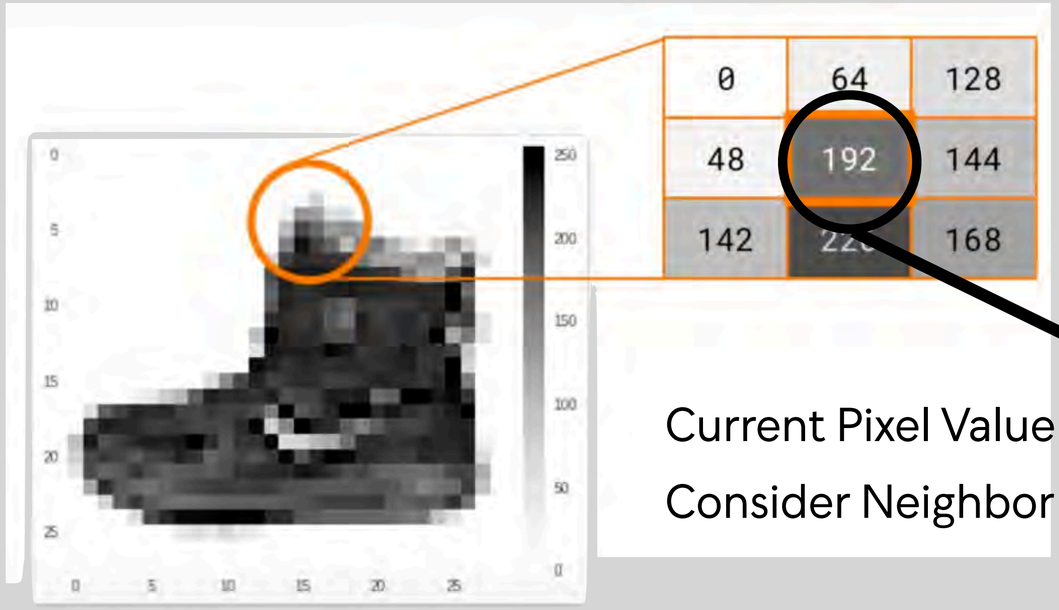
Some other filters may detect corners, circles, squares

The deeper our network goes the more sophisticated these filters become.

In later layers rather than edges or shapes our filters may be able to detect specific objects like eyes, ears, hair, fur or complete animals etc.



Filters



-1	0	-2
.5	4.5	-1.5
1.5		-3

Current Pixel Value 192

Consider Neighbor values

```
CURRENT_PIXEL_VALUE =  192

NEW_PIXEL_VALUE = (-1 * 0) + (0 * 64) + (-2 * 128) +
                  (0.5 * 48) + (4.5 * 192) + (-1.5 * 144) +
                  (1.5 * 142) + (2 * 226)  + (-3 * 168)
```

-1	0	1
-2	0	2
-1	0	1

-1	-2	-1
0	0	0
-1	2	1

Pooling

Groups up the pixels in the image and filters them down to a subset.

Max Pooling two by two

Will group the image into sets of 2x2 pixels and simply pick the largest.

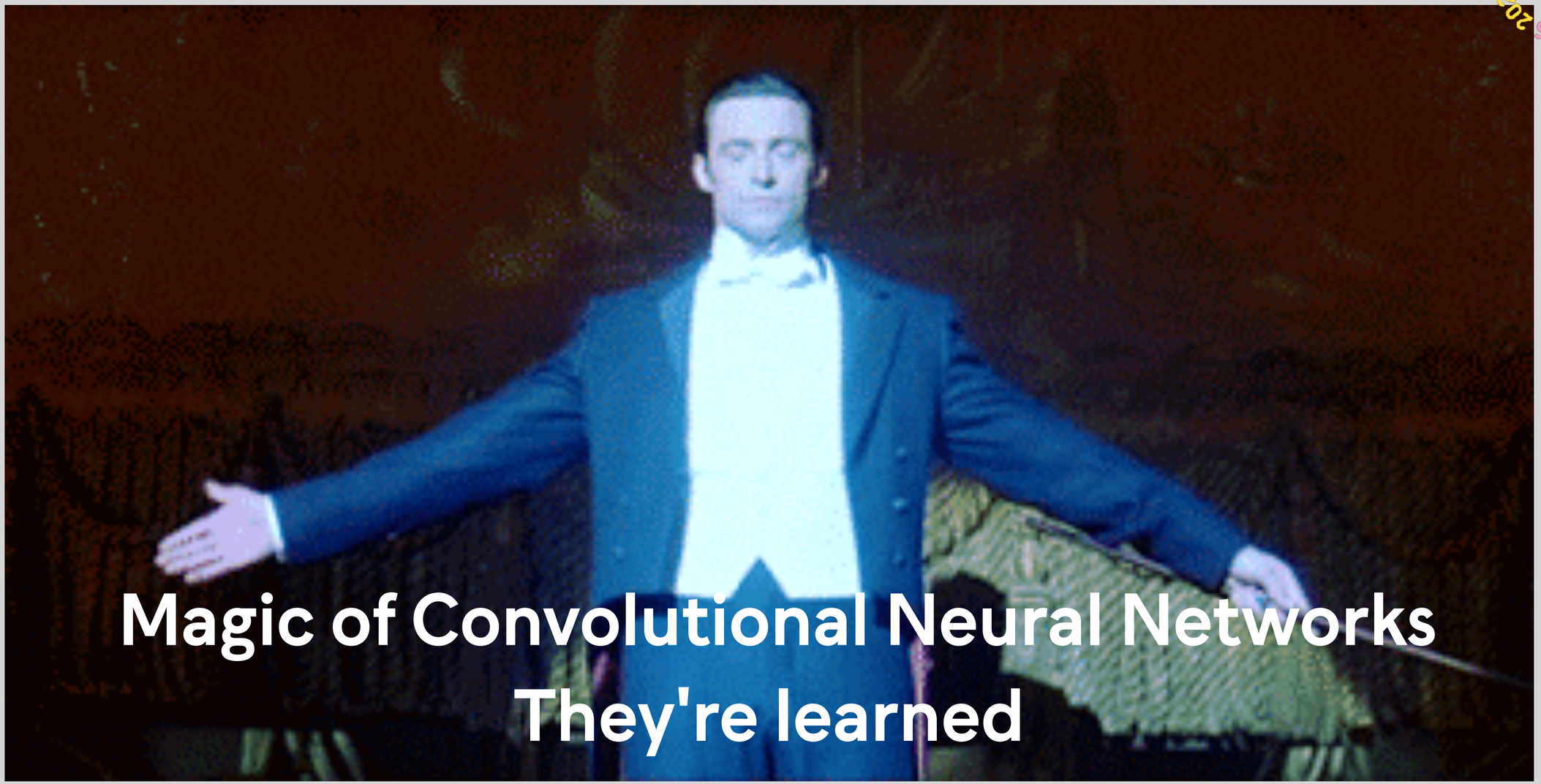
(**Pooling** layers are used to reduce the dimensions of the feature maps)

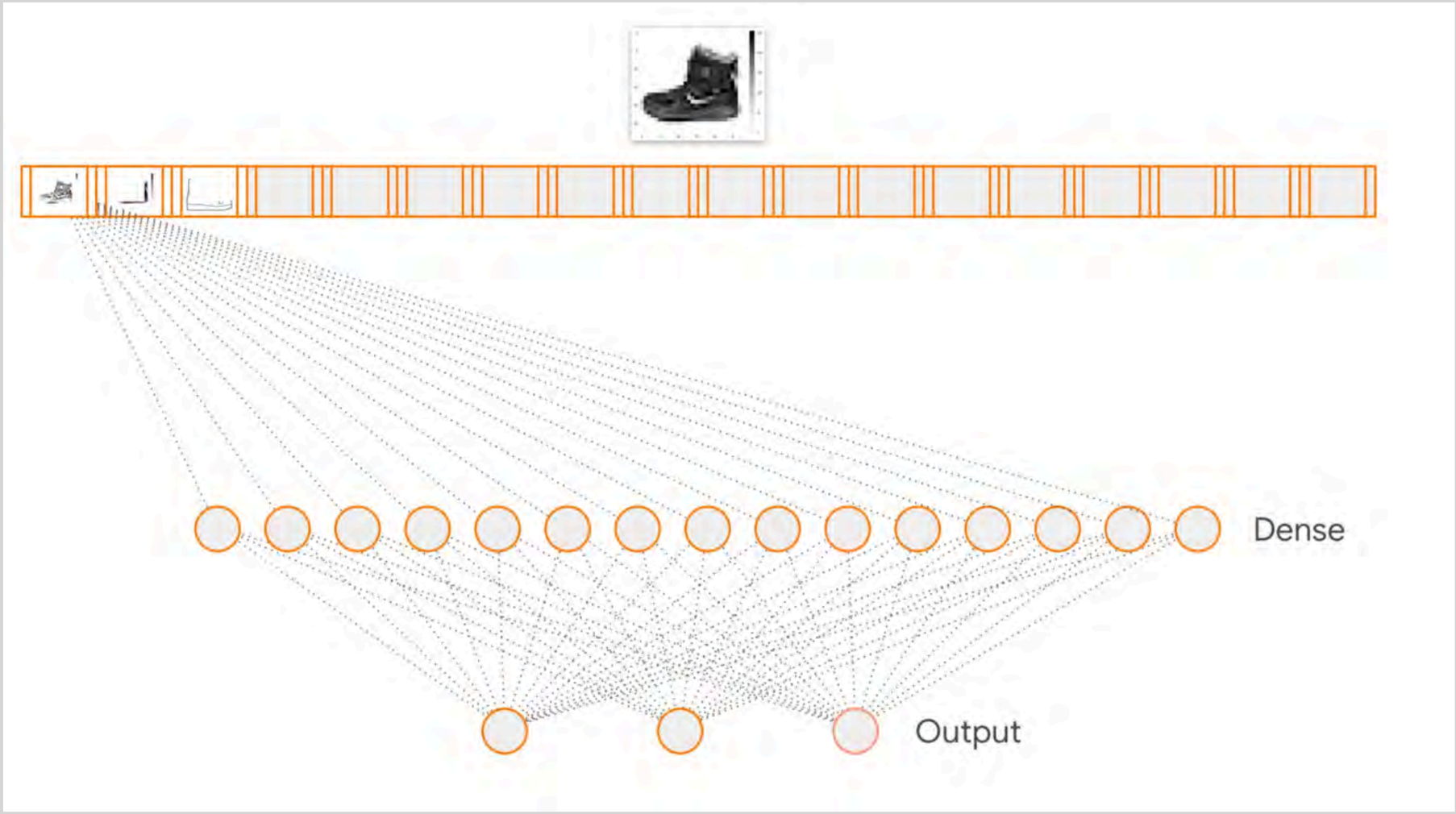
The image will be reduced to a quarter of its original size but the features can still be maintained.

Max Pooling two by two

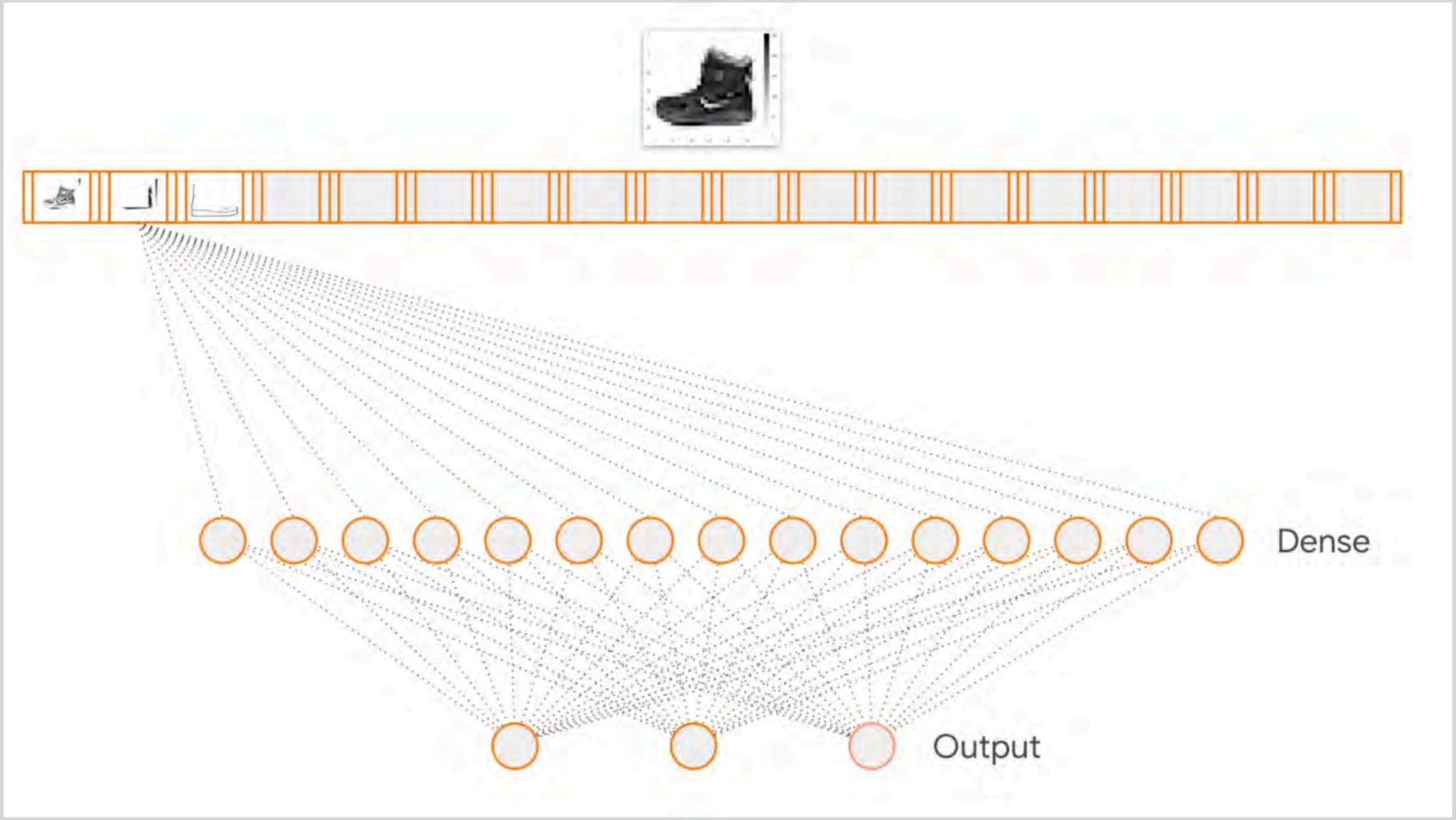
This image is one quarter the size of the one on the left, but the vertical line features were maintained and enhanced.

Where did these filters come from?

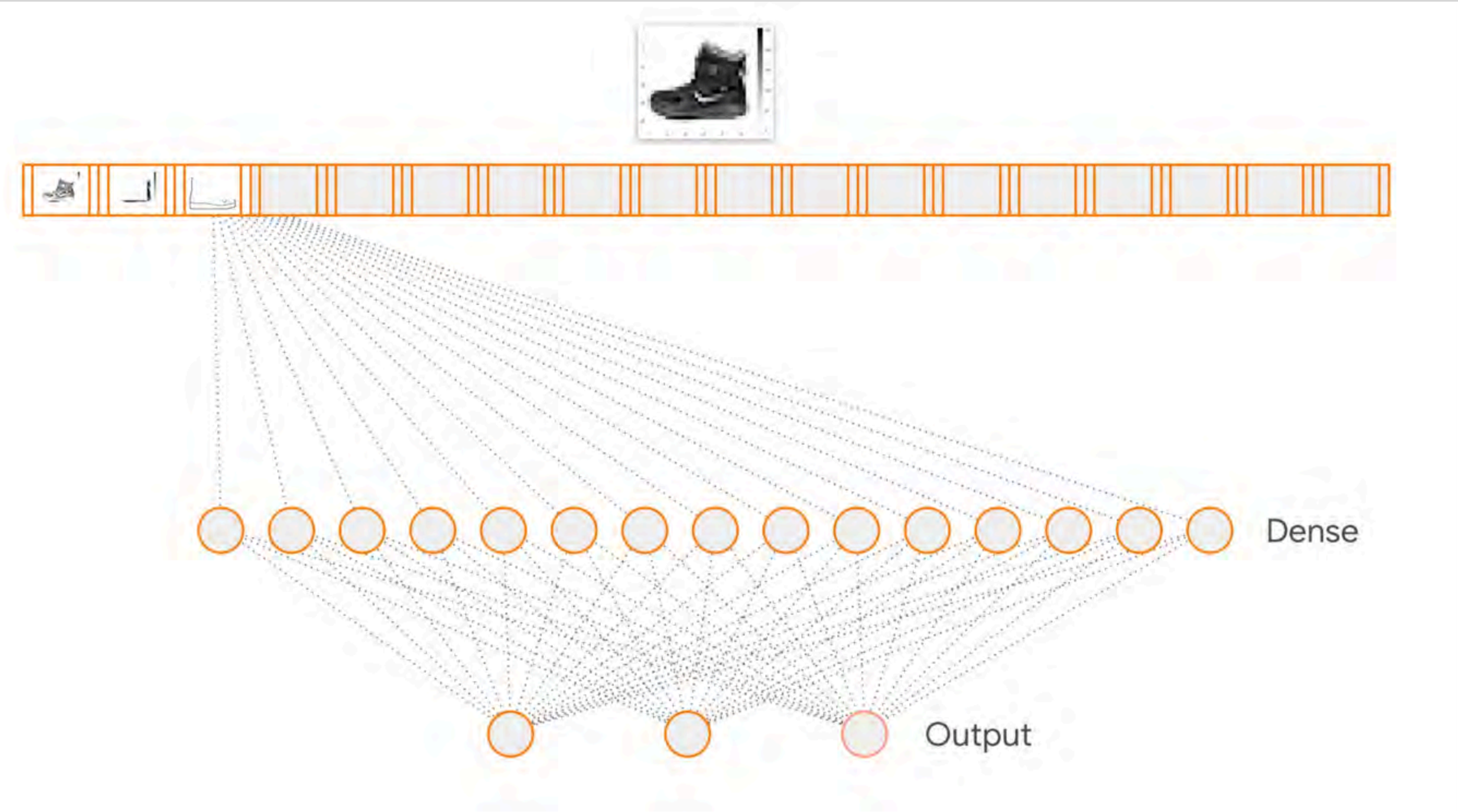




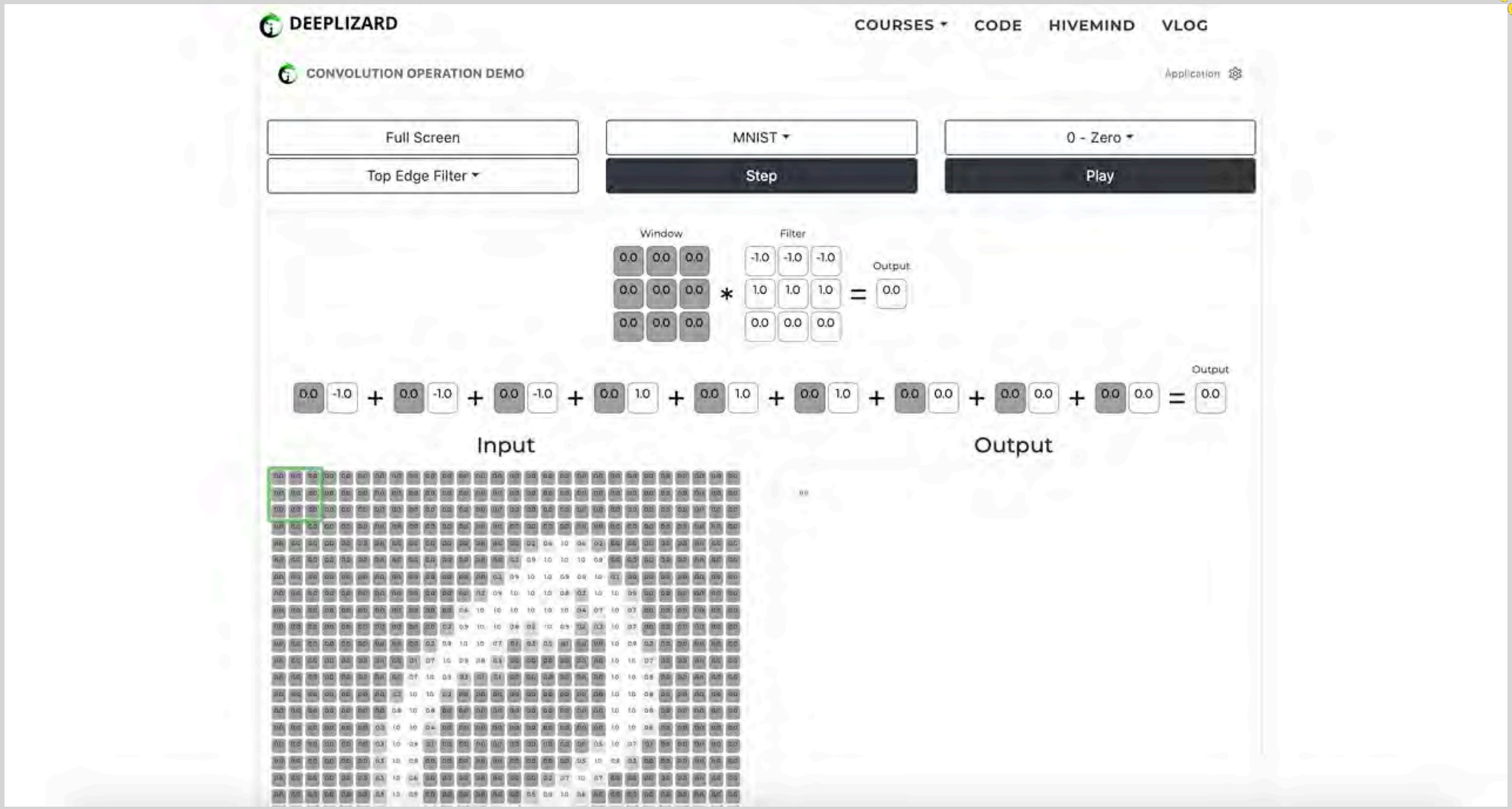
As our image is fed into the convolutional layer, a number of randomly initialized filters
will pass over the image. The results of these are fed into the next layer matching is performed by the neural network.



As our image is fed into the convolutional layer, a number of randomly initialized filters
will pass over the image. The results of these are fed into the next layer matching is performed by the neural network.



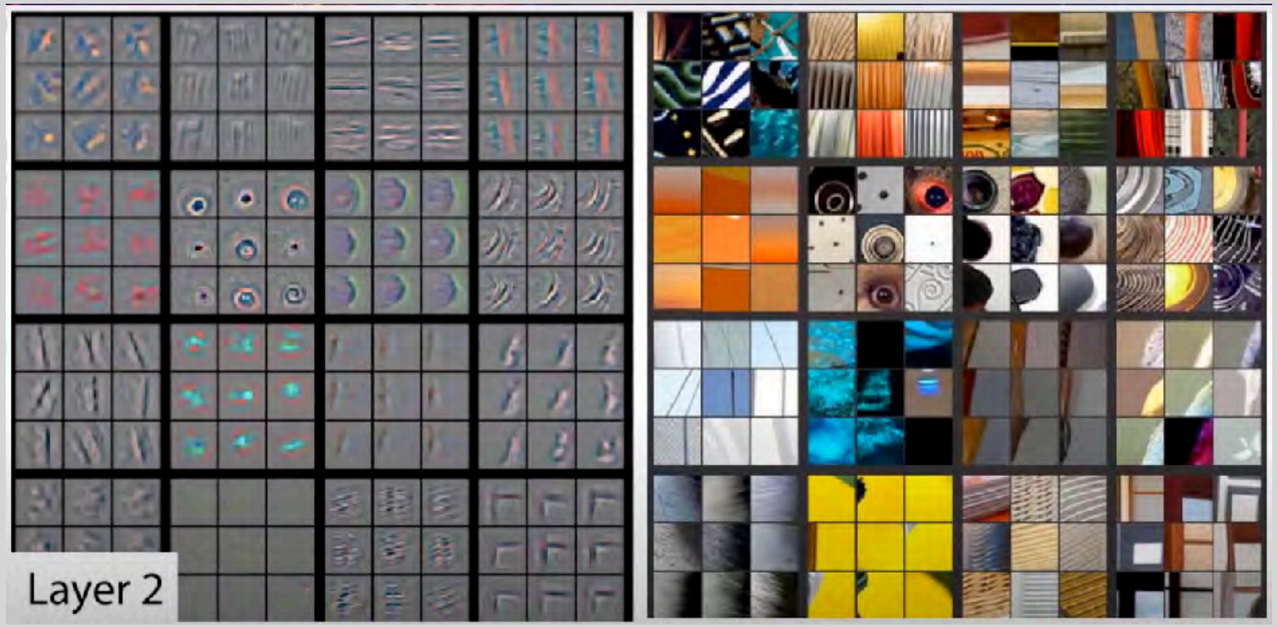
And over time, the filters that give us the best image output matches will be learned. The process is called feature extraction.

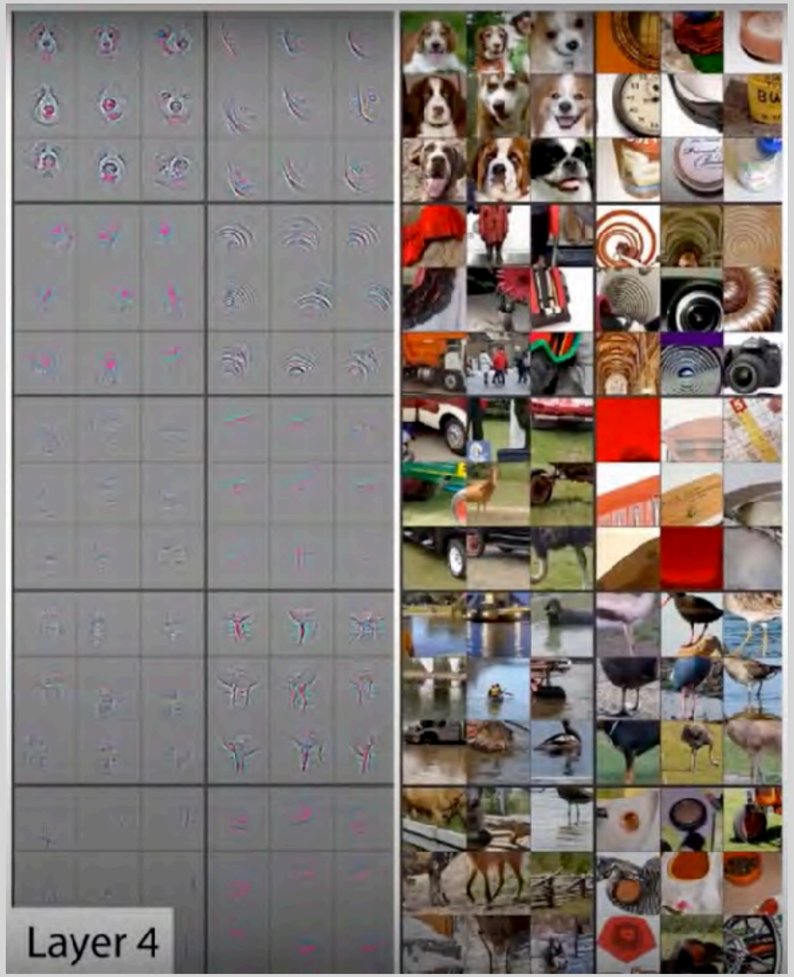


https://deeplizard.com/resource/pavq7noze2

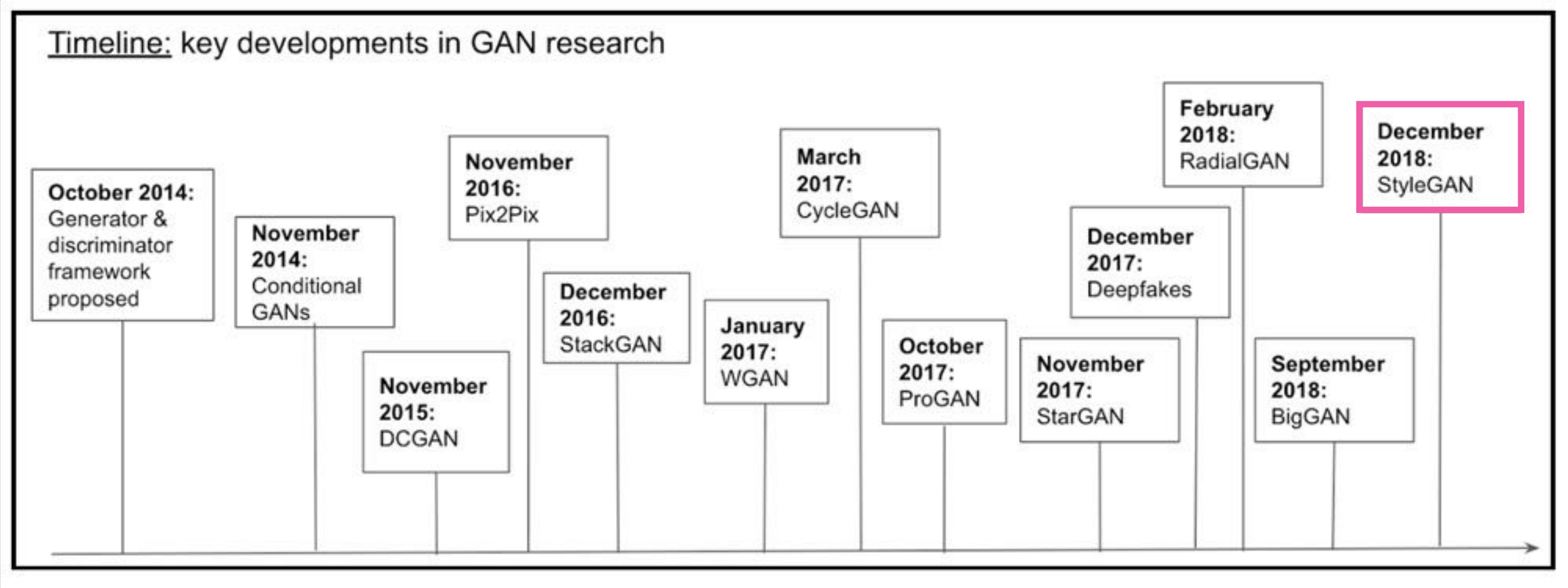
With this methodology, our network starts to learn based on the features of the image instead of just the raw patterns of pixels.

This principles will extend into more complex images


Layer 2


Layer 4

GAN variations and timelines



StyleGAN

StyleGAN is a GAN design released by researchers at **NVIDIA** in December 2018.

It is essentially an upgraded version of ProGAN. It combined ProGAN with neural style transfer.

At the core of the StyleGAN architecture is a **style-transfer technique.**

The model set a new record for face generation tasks and can also be used to **generate realistic images** of cars, bedrooms, houses, and so on.

A Style-Based Generator Architecture for Generative Adversarial Networks
Tero Karras, Samuli Laine, Timo Aila
NVIDIA

https://arxiv.org/pdf/1812.04948.pdf

StyleGAN

The StyleGAN for short, is an extension to the GAN architecture that proposes large changes to the generator model, including the **use of a mapping network** to map points in latent space to an intermediate latent space, the use of the intermediate latent space to **control style** at each point in the generator model, and the introduction to **noise as a source of variation at each point** in the generator model.

A Style-Based Generator Architecture for Generative Adversarial Networks
Tero Karras, Samuli Laine, Timo Aila
NVIDIA

https://arxiv.org/pdf/1812.04948.pdf

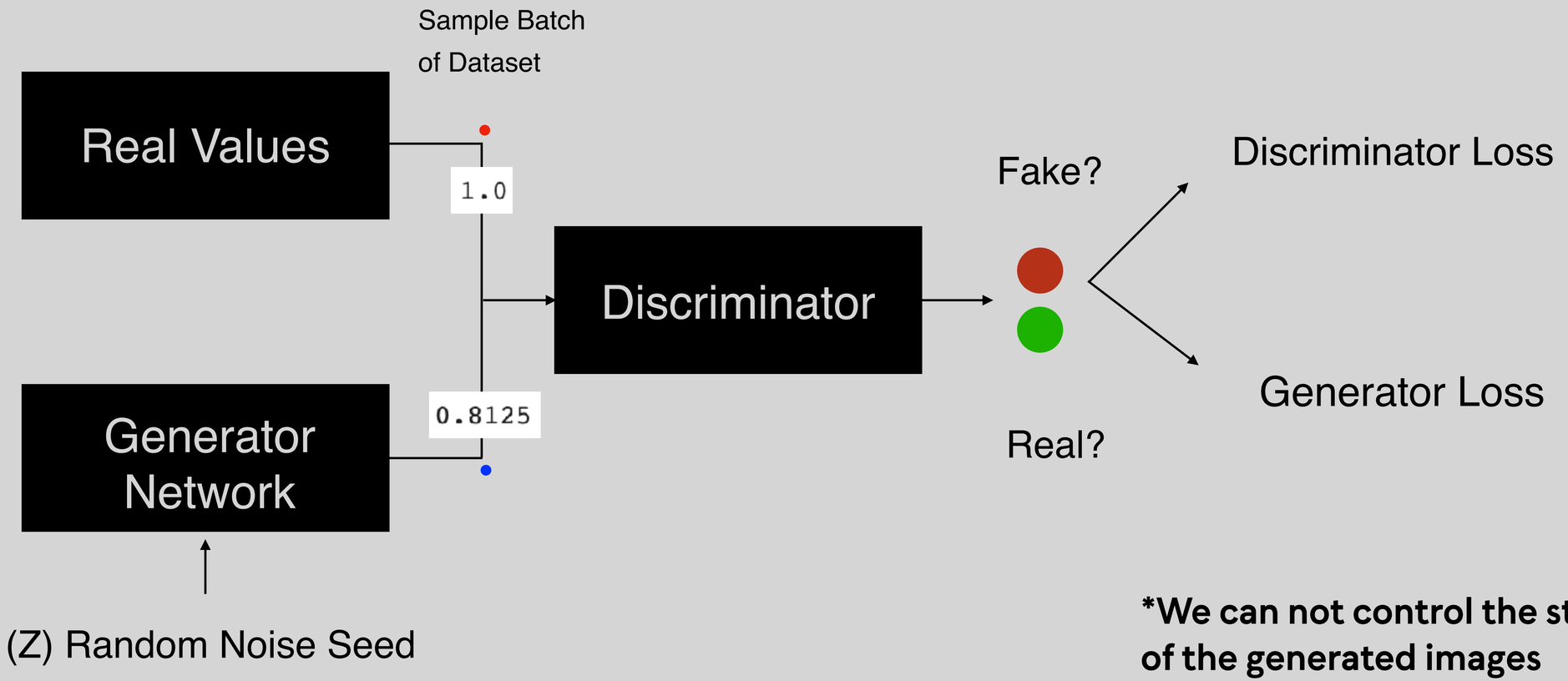
Contribution

Propose a Generator Architecture where can control the synthesis (Control the style)

Separation of Pose, Identify face, freckles, hair

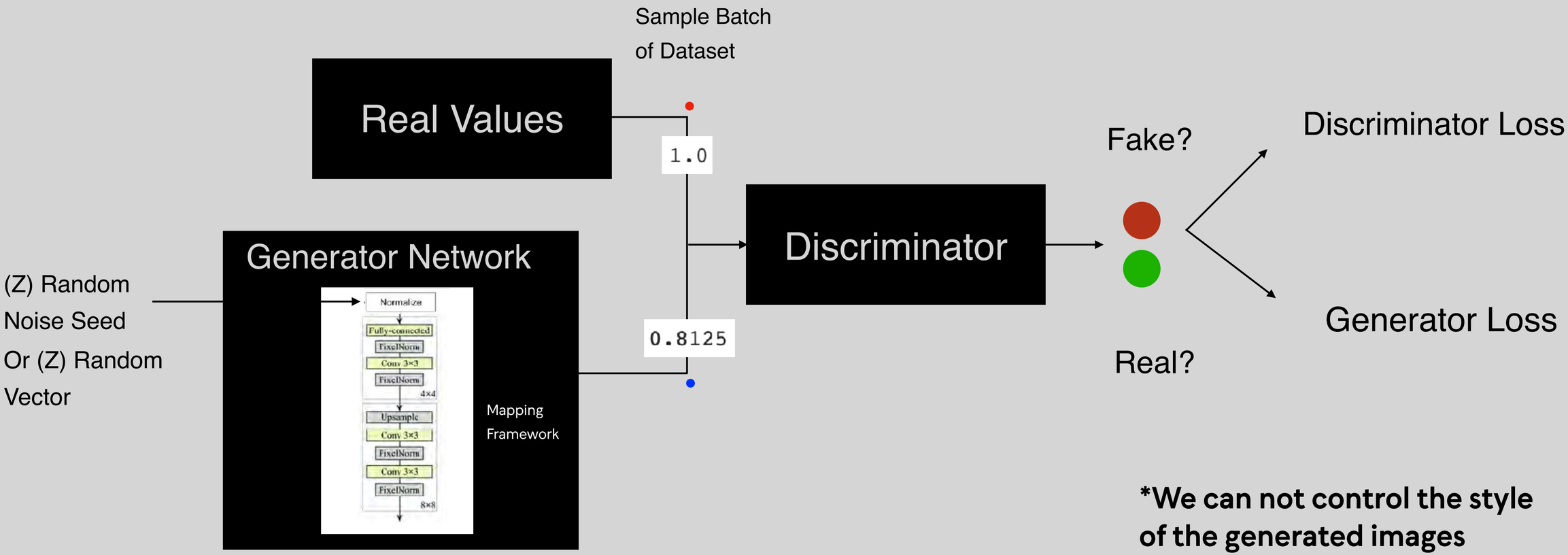
High Quality faces dataset: Flickr Faces-HQ (FHHQ) of 70K with very high resolution (1024x1024) of diverse faces

GAN structure



***We can not control the style of the generated images**

GAN structure



***We can not control the style of the generated images**

GAN structure

Sample Batch
of Dataset

Real Values

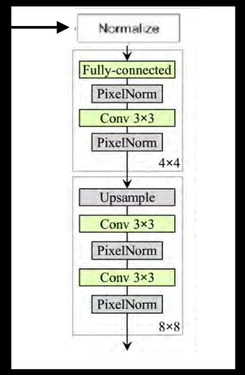
1.0

Fake?

Discriminator

Generator Network

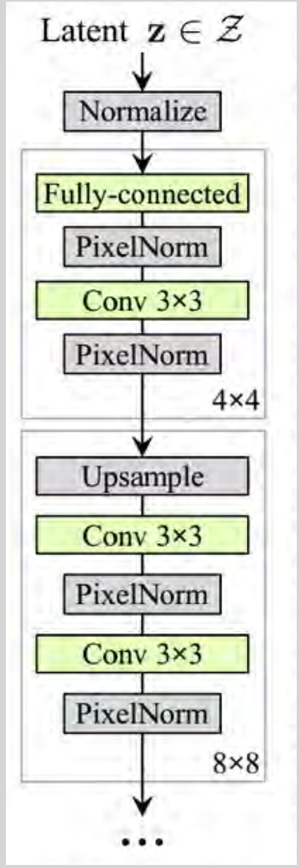
(Z) Random
Noise Seed
Or (Z) Random
Vector



0.8125

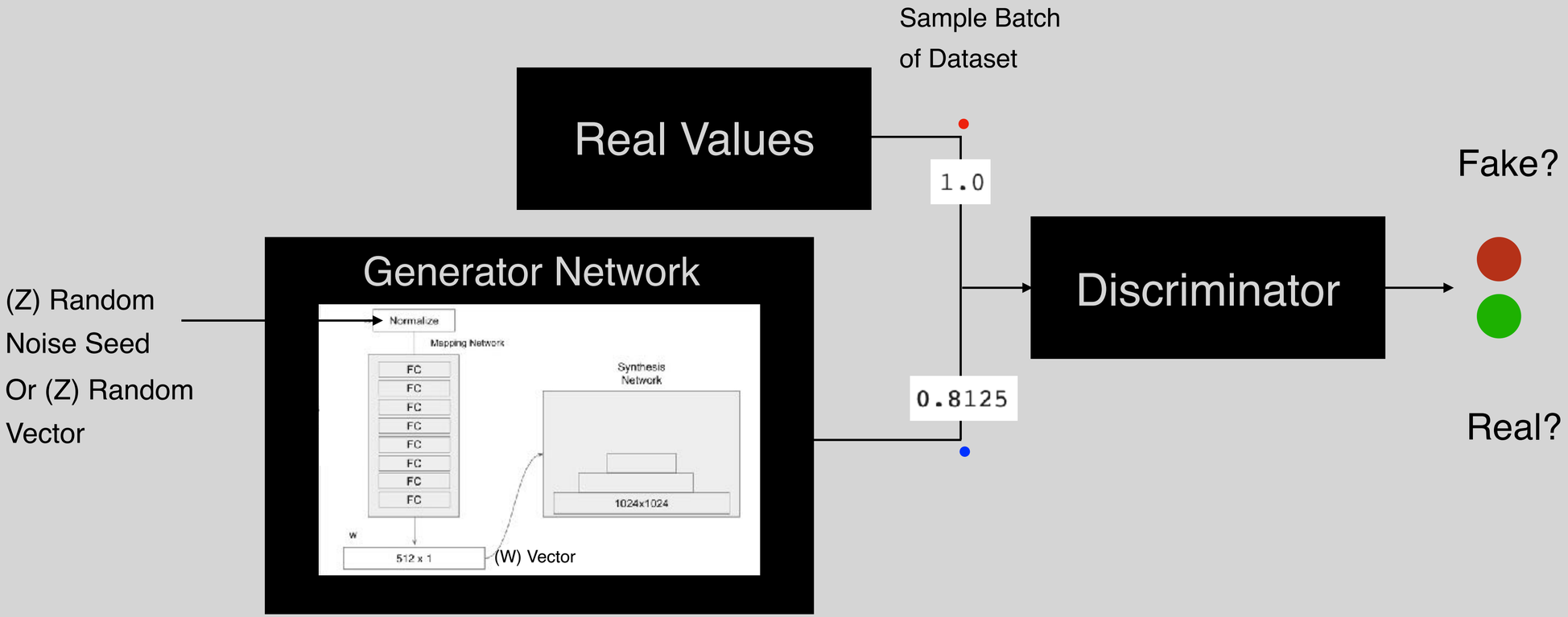
Real?

Traditional GAN
Generator Network



StyleGAN structure

-Introduce Mapping Framework
-Introduce Synthesis Network



StyleGAN structure

-Affine transformation
-Adaptive normalization

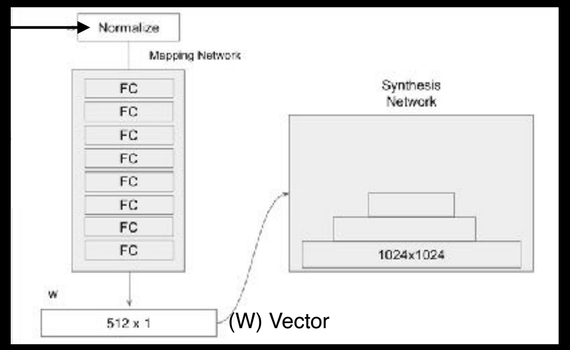
Sample Batch
of Dataset

Real Values

1.0

(Z) Random
Vector

Generator Network



(W) Vector

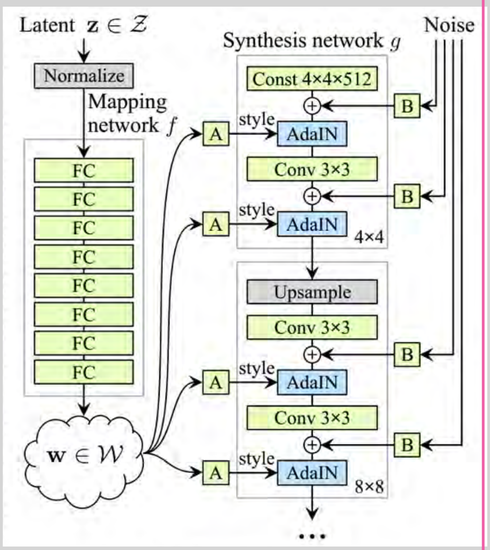
0.8125

Discriminator

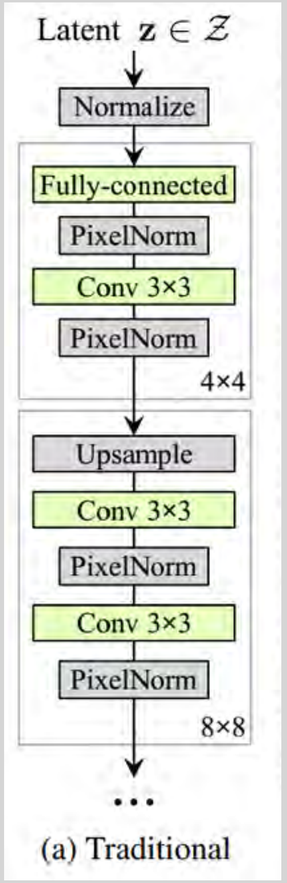
Fake?

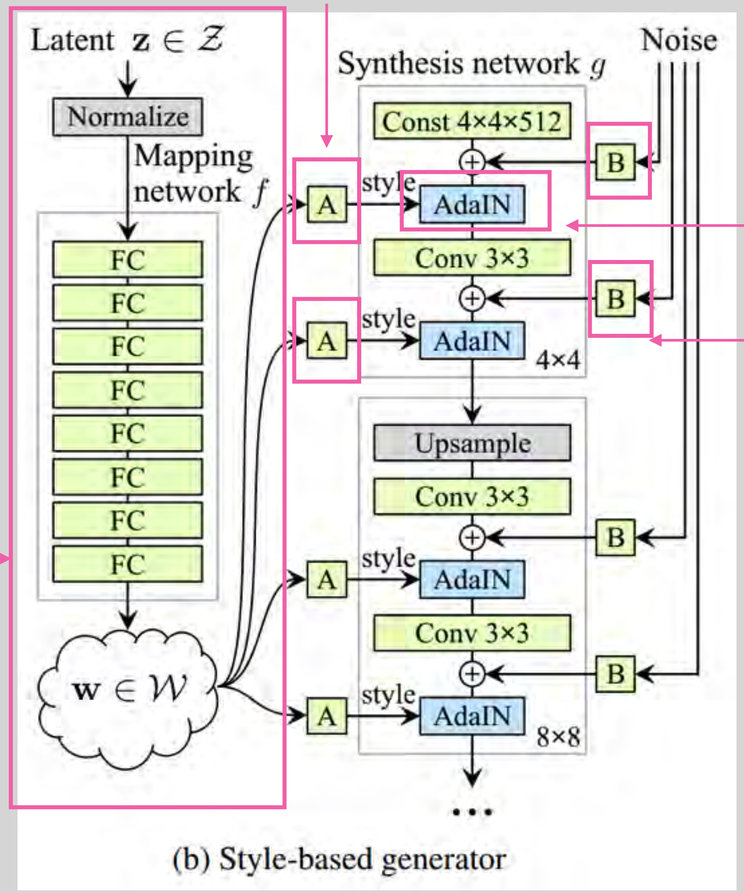
Real?

Styled-Based
Generator Network



GAN vs. StyleGAN





Affine Transformation (A)

The recovered vector W is then fed to a series of learned affine transformations which **predicts a style vector** that control an Adaptive Instance Normalization (AdaIN)

The **mapping network** takes as input the latent vector (Z) and then maps it to another vector called (W)

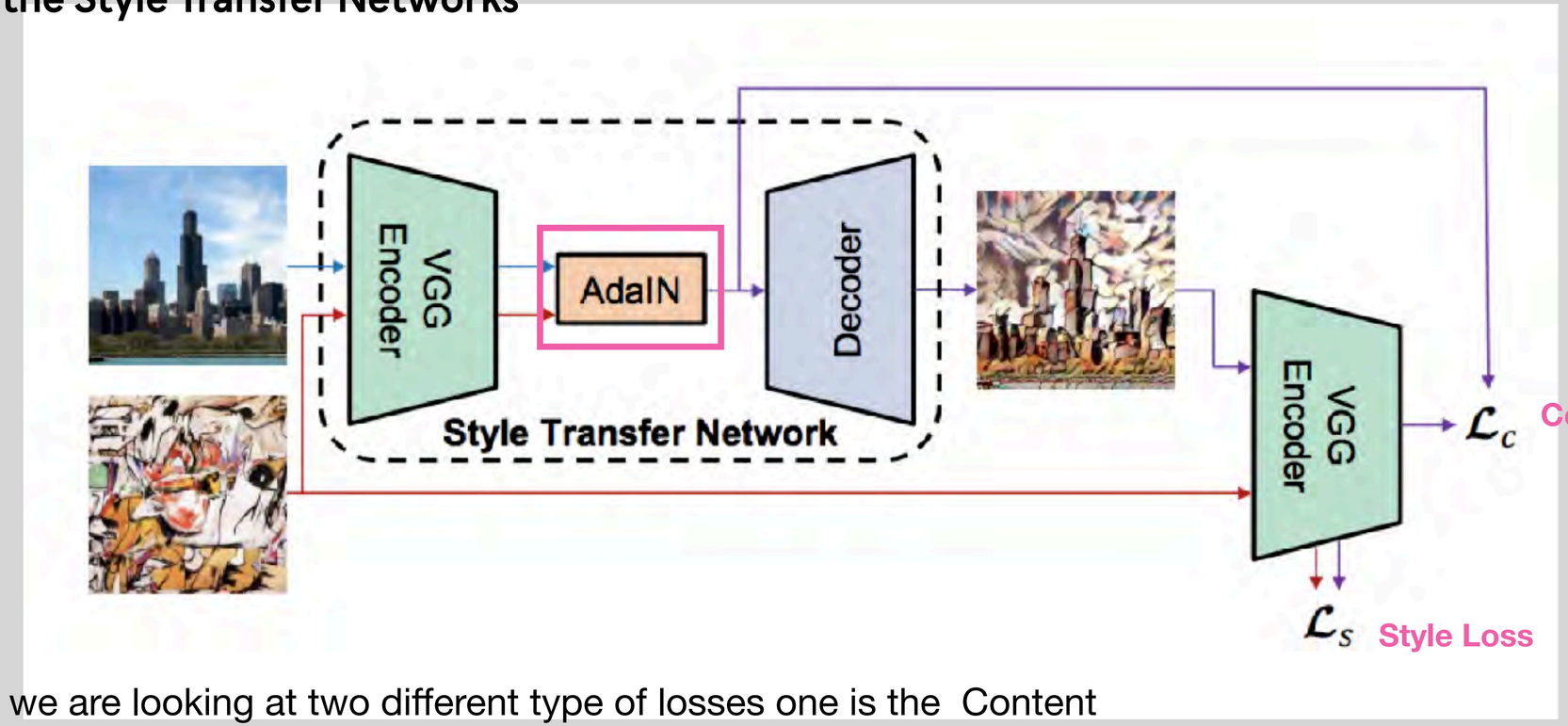
Adaptive Normalization (AdaIN)

Noise Vector (B)

The system feeds noise to the input of each AdaIN operation

Style Transfer

StyleGAN proposes a model for the Generator that is inspired by the Style Transfer Networks



During this process we are looking at two different type of losses one is the Content Loss that you see in the network and the Style loss.
The idea is to minimize the style loss to effectively transfer the style to the image.

A Literature Review of Neural Style Transfer - Haochen Li. Princeton University, Princeton NJ 08544

Batch Normalization / IN /AdaIN

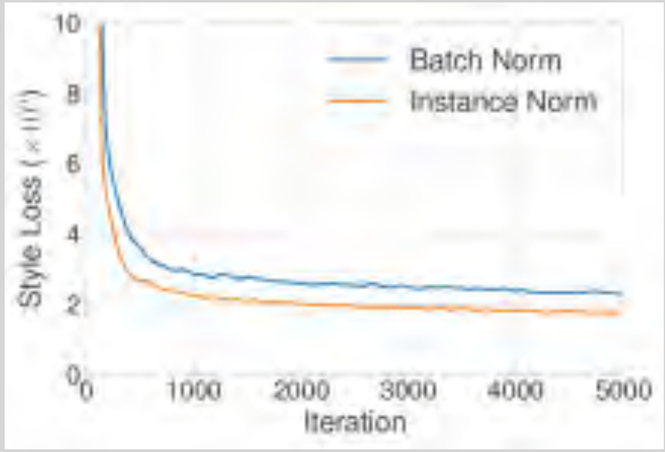
GANs uses batch normalization to improve the discriminate training.
This wasn't used initially in the generator. Doing batch normalization works very nicely but then the next step was to do it for every instance

I. Batch Normalization (BN) layer

$$\mathrm{BN}(x) = \gamma\left(\frac{x - \mu(x)}{\sigma(x)}\right) + \beta \quad (1)$$

2. Instance Normalization

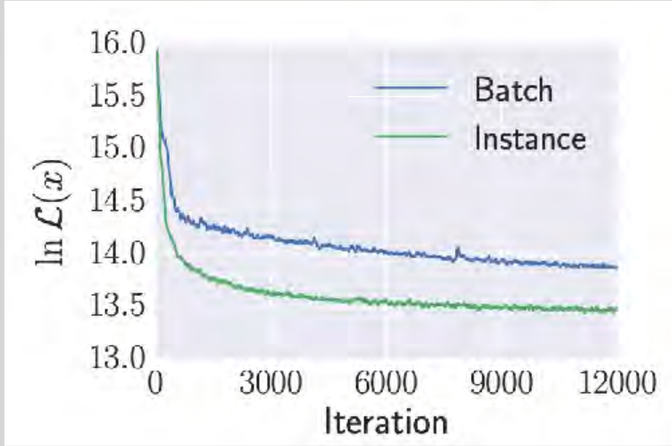
$$\mathrm{IN}(x) = \gamma\left(\frac{x - \mu(x)}{\sigma(x)}\right) + \beta \quad (4)$$



Style Transfer Network contains a BN layer after each convolutional layer. That significant improvement could be achieved simply by replacing BN layers with IN layers. Instead of calculating the mean and standard deviation or your batch and then learn these parameters, we're going to do the same but for every instance and when you do that for every instance, the style the network improves the learning.

BN vs IN

Batch Normalization (BN) vs Instance Normalization (IN)



Content

Style

StyleNet BN

StyleNet IN (ours)

Improved Texture Networks: Maximizing Quality and Diversity in Feed-forward Stylization and Texture Synthesis. Dmitry Ulyanov, Andrea Vedaldi, Victor Lempitsky 2017

Adaptive Normalization (AdaIN)

1. Batch Normalization

$$\mathrm{BN}(x) = \gamma\left(\frac{x-\mu(x)}{\sigma(x)}\right) + \beta$$

Scale Factor

Translation Factor

You have input (x), Batch Normalization computes the mean and standard deviation of (x). Also has parameters gamma and beta which is a scale factor and translation factor respectively.

2. Instance Normalization

$$\mathrm{IN}(x) = \gamma\left(\frac{x-\mu(x)}{\sigma(x)}\right) + \beta$$

Scale Factor

Translation Factor

Instance Normalization is a breakdown of Batch Normalization for specific instance. So you can treat each sample in a batch separately. Instead of calculating parameters for the whole batch, it will do it for each sample.

3. Adaptive Instance Normalization

Input Style

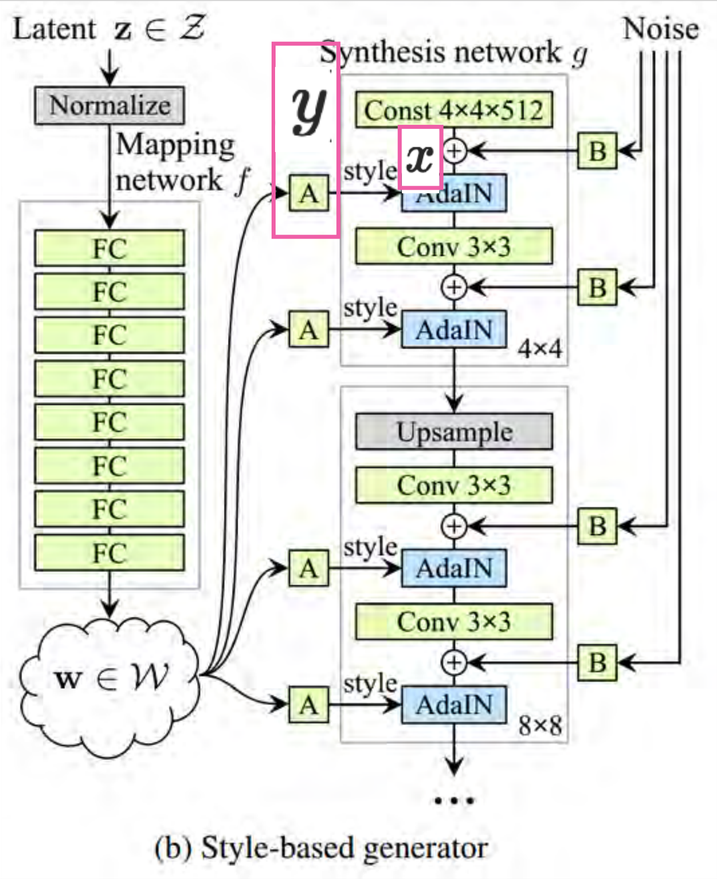
Scale Factor

Translation Factor

$$\mathrm{AdaIN}(x, y) = \sigma(y)\left(\frac{x-\mu(x)}{\sigma(x)}\right) + \mu(y)$$

Specialization of the Instance Normalization, where you have the input (x) and input (y). Also we have the (y) as a scale factor and translation Factor

Adaptive Normalization (AdaIN)



(b) Style-based generator

3. Adaptive Normalization

Input Style

Scale Factor

Translation Factor

$$\text{AdaIN}(x, y) = \sigma(y)\left(\frac{x - \mu(x)}{\sigma(x)}\right) + \mu(y)$$

The style y comes from W vector and the x come from the previous convolution layers. They both are fed into adaptive normalization (AdaIN) and because y has control over the scale and translation of the features, that's why we can say that you **have control over the output images that are generated**

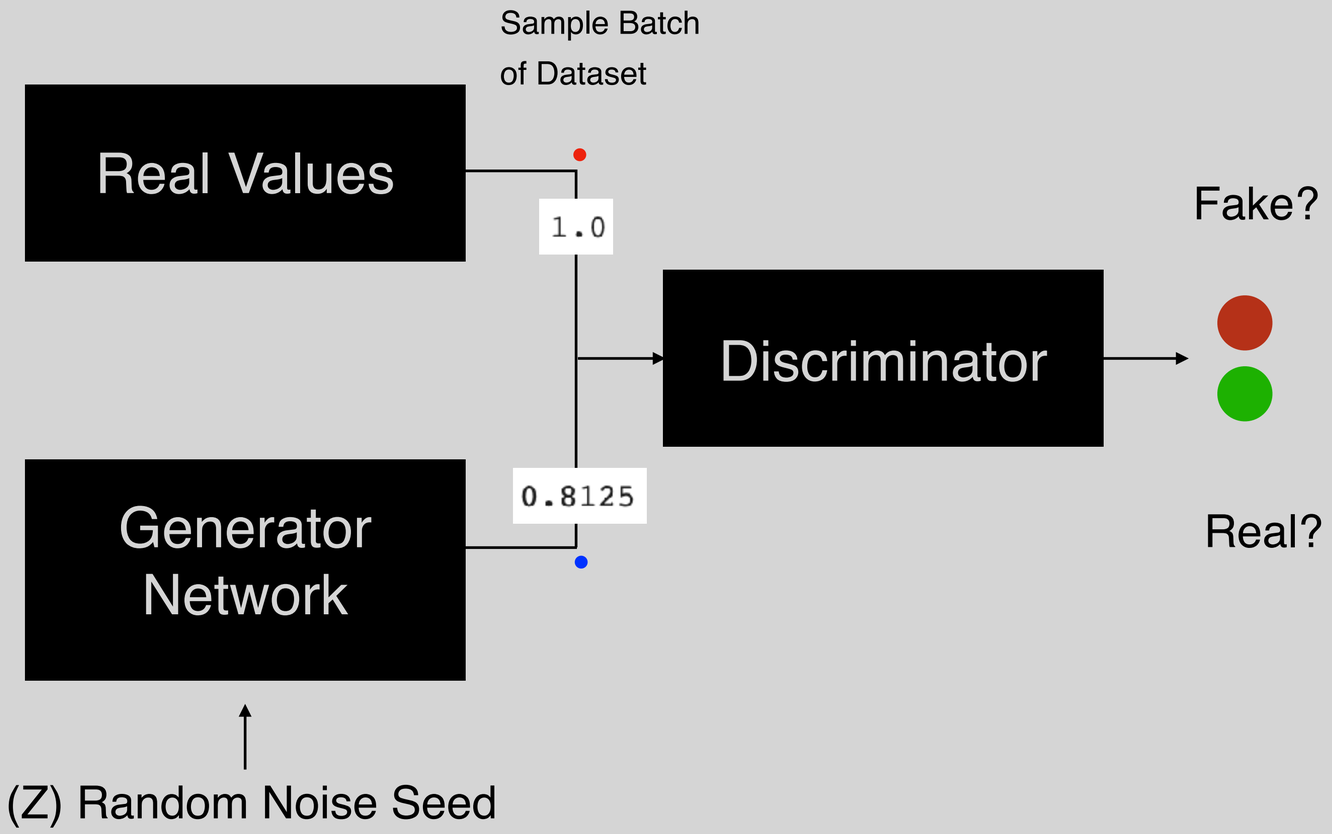
Styles

Our Generator thinks of an image as a collection of Styles, where each style controls the effects at a particular scale

Coarse styles ———- Pose, hair, face shape

Middle Styles ———-Facial features, eyes

Fine Styles ———— Color Scheme



Coarse styles:
Pose, hair, face shape

Middle Styles:
Facial features, eyes

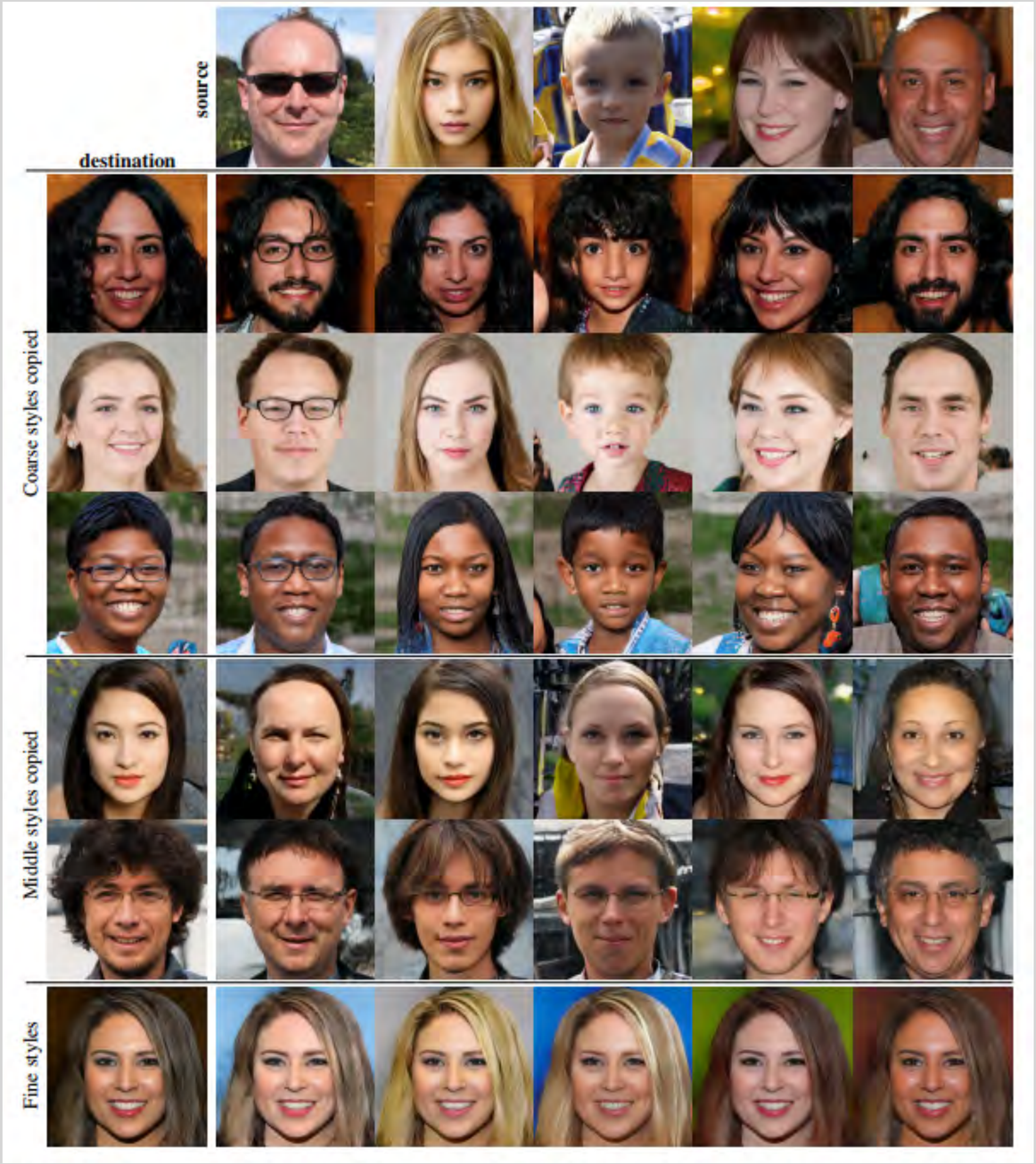
Fine Styles:
Color Scheme

cape
Campus Asia Plant Environment innovation

Coarse styles:
Pose, hair, face shape

Middle Styles:
Facial features, eyes

Fine Styles:
Color Scheme



CAPE WS 2021 CAPE WS 2021

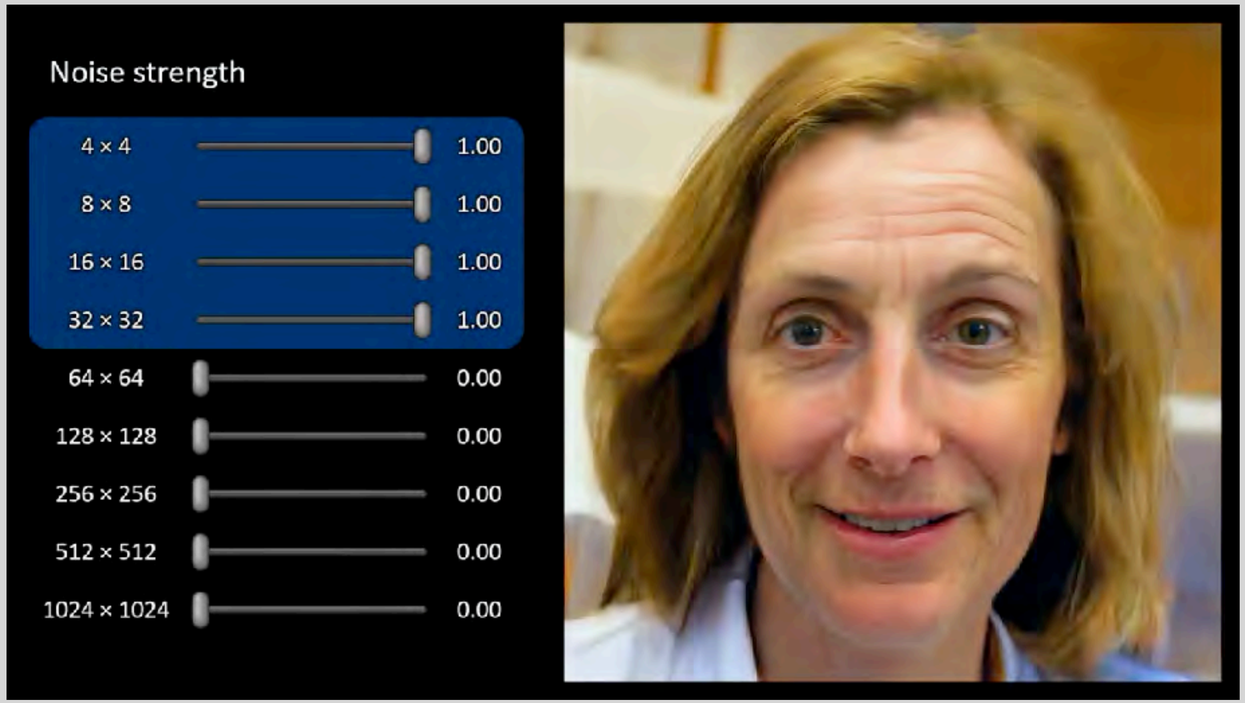
A Style-Based Generator Architecture for Generative Adversarial Networks

Tero Karras, Samuli Laine, Timo Aila
NVIDIA

https://arxiv.org/pdf/1812.04948.pdf

Noise

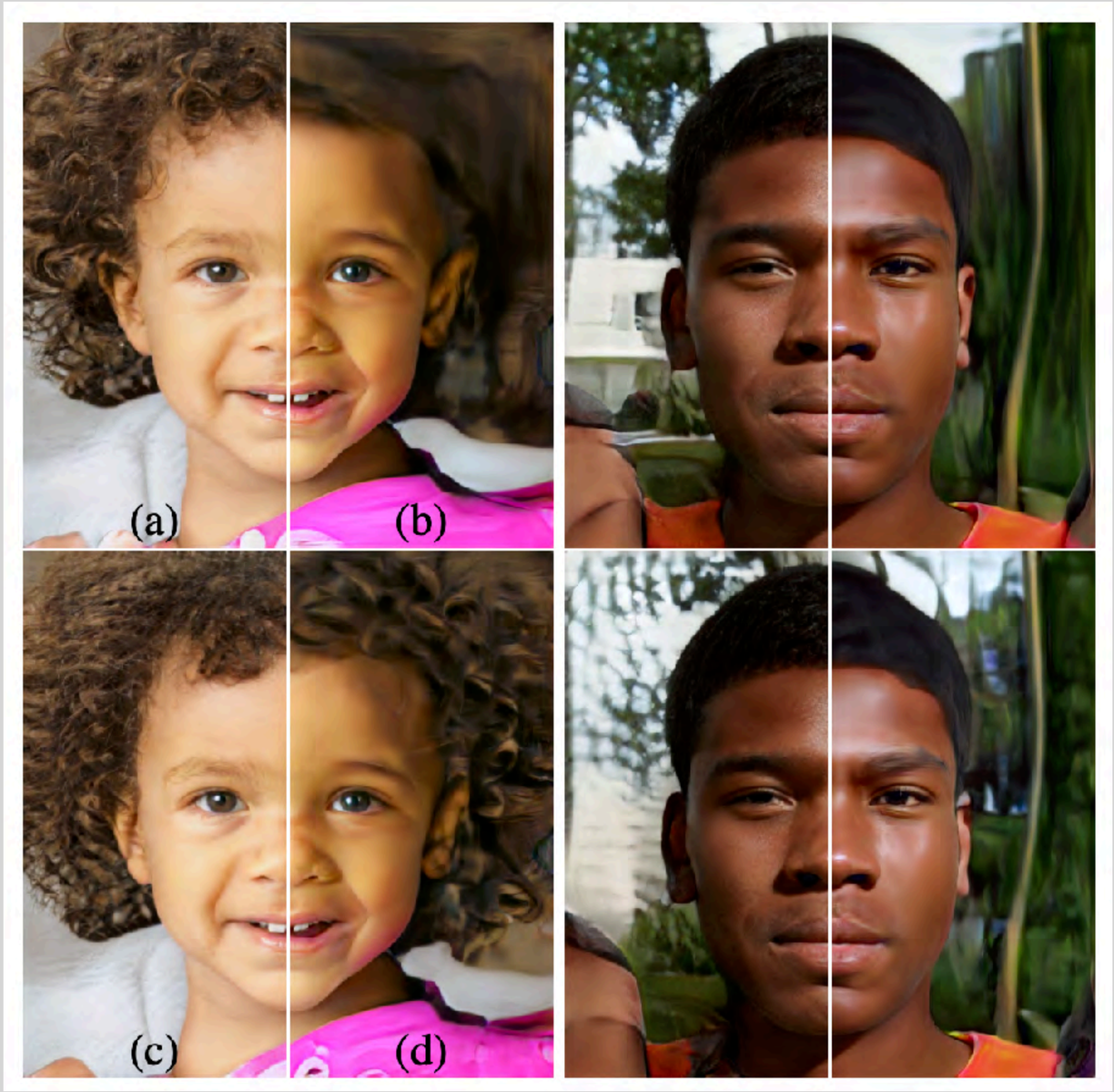
Additionally, our generator automatically separates inconsequential variation from high-level attributes (pose, identity etc)



Coarse Noise: Large-Scale curling of hair

Fine Noise: Finer details, texture

No Noise : Featureless “painterly” look



(a) Noise is applied to all layer
(b) No noise
(c) Noise in Fine Layers Only
(d) Noise in coarse layers only

A Style-Based Generator Architecture for Generative Adversarial Networks

Tero Karras, Samuli Laine, Timo Aila
NVIDIA

https://arxiv.org/pdf/1812.04948.pdf

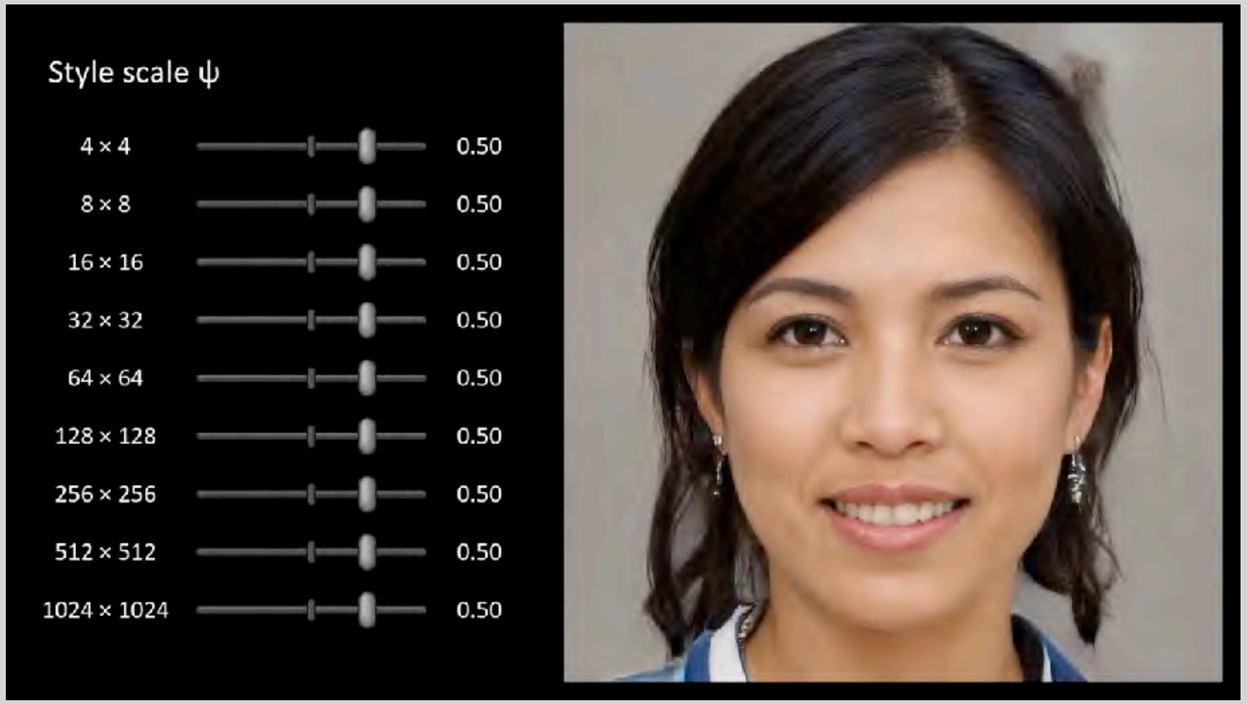
Strength of the style

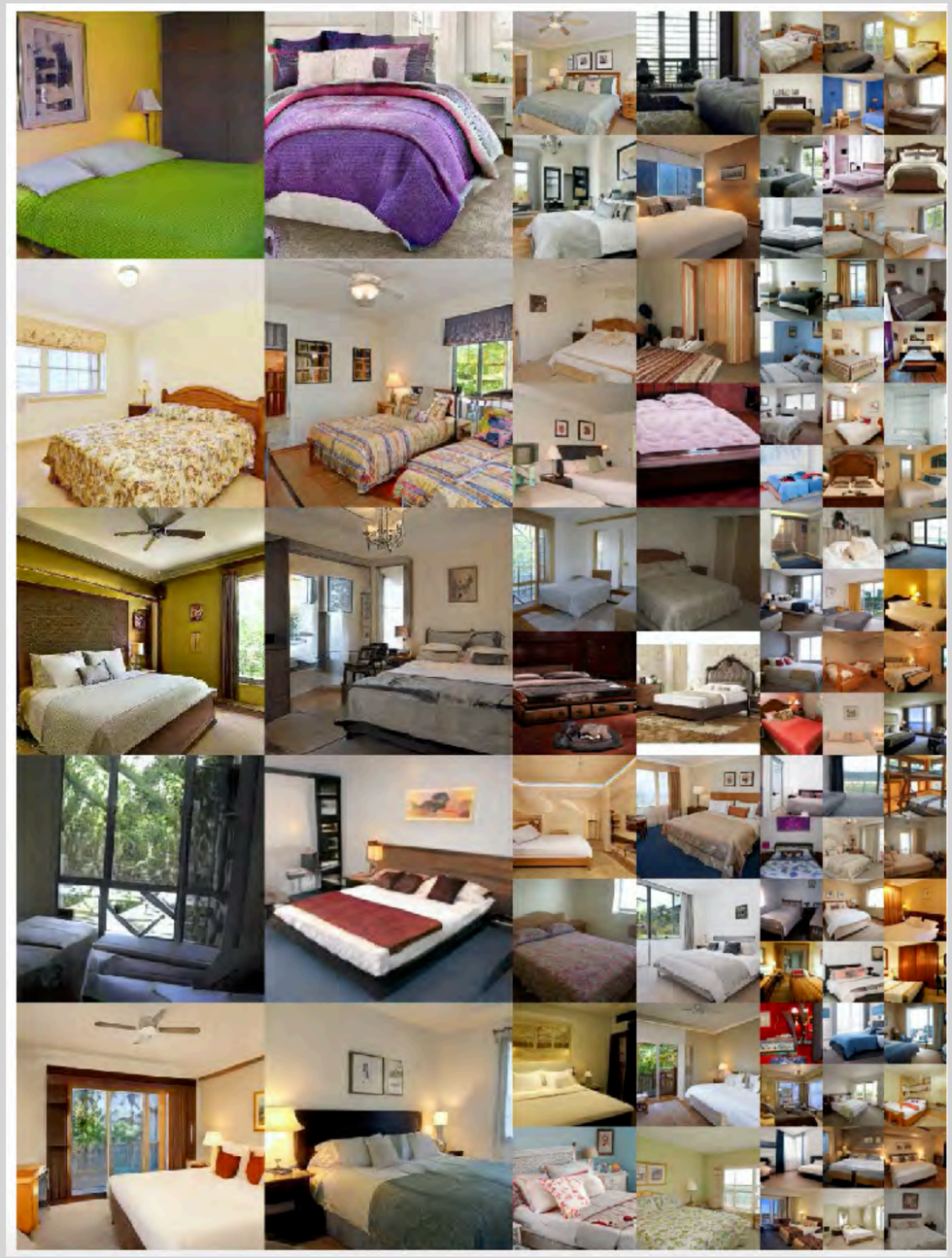
We can choose the strength at which each style is applied with respect to an average face.

High Strength: Maximal variation (Broken images)

Low Strength: Reduced variation (No broken)

By selecting the appropriate strength we can get good images (With a slightly reduced variations)





Uncurated set of images produced by our style-based generator (config F) with the LSUN CAR dataset at 512 × 384. FID computed for 50K images was 3.27.

Frechet **Inception Distance** score, or **FID** for short, is a metric that calculates the **distance** between feature vectors calculated for real and generated images.

A Style-Based Generator Architecture for Generative Adversarial Networks

Tero Karras, Samuli Laine, Timo Aila
NVIDIA

https://arxiv.org/pdf/1812.04948.pdf

Uncurated set of images produced by style-based generator (config F) with the LSUN BEDROOM dataset at 256x256. FID computed for 50K images was 2.65.

Frechet **Inception Distance** score, or **FID** for short, is a metric that calculates the **distance** between feature vectors calculated for real and generated images.

A Style-Based Generator Architecture for Generative Adversarial Networks

Tero Karras, Samuli Laine, Timo Aila
NVIDIA

https://arxiv.org/pdf/1812.04948.pdf

StyleGAN Samples

https://github.com/t04glovern/stylegan-pokemon

t04glovern / stylegan-pokemon

Watch 0 | Star 31 | Fork 7

Code | Issues 1 | Pull requests | Actions | Projects | Security | Insights

master | stylegan-pokemon / README.md | Go to file | ...

t04glovern added sample website — Latest commit ceee7df on May 22, 2019 — History

1 contributor

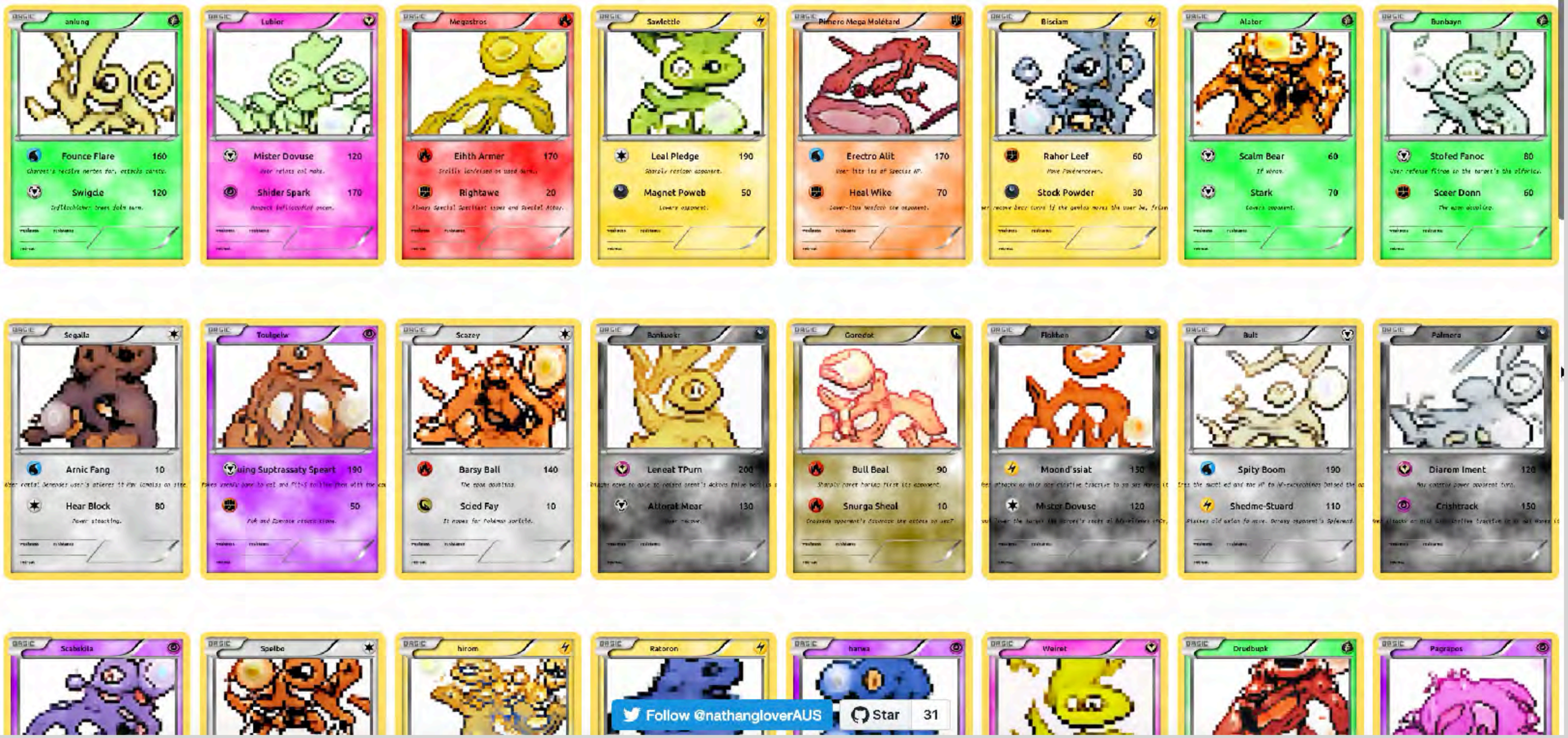
167 lines (125 sloc) 3.69 KB — Raw | Blame

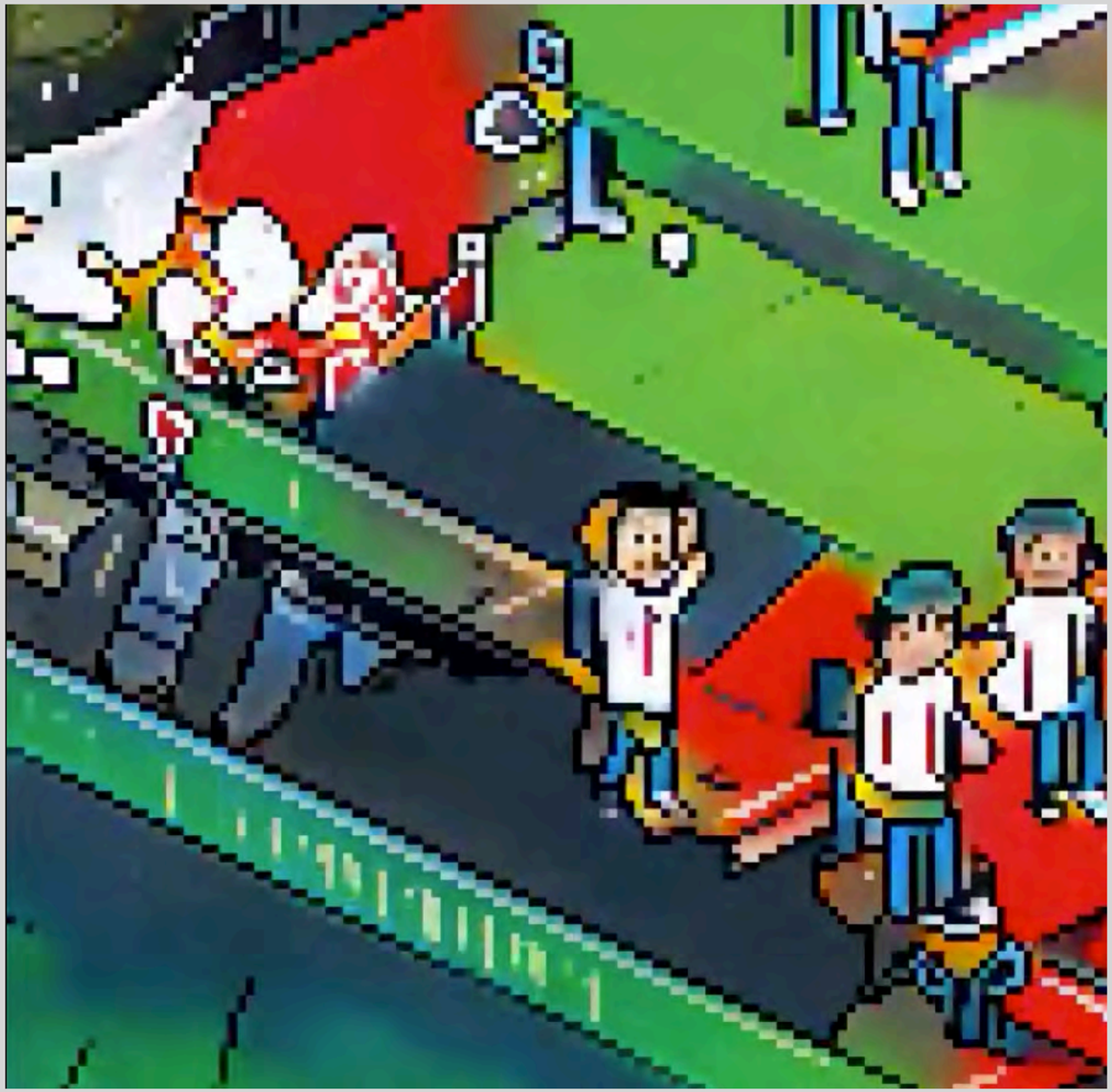
Pokemon Card Generator

Generating Pokemon cards using a mixture of StyleGAN and RNN to create beautiful & vibrant cards ready for battle!

StyleGAN Pokemon Card Generator

Learn More: StyleGAN & RNN Pokemon Card Generator





An interesting AI art project that combines a GAN on an infinite loop, trained on the eBoy dataset.
By Max Braun

For those unfamiliar, eBoy creates reusable pixel objects and uses those to create art, make toys, and more.

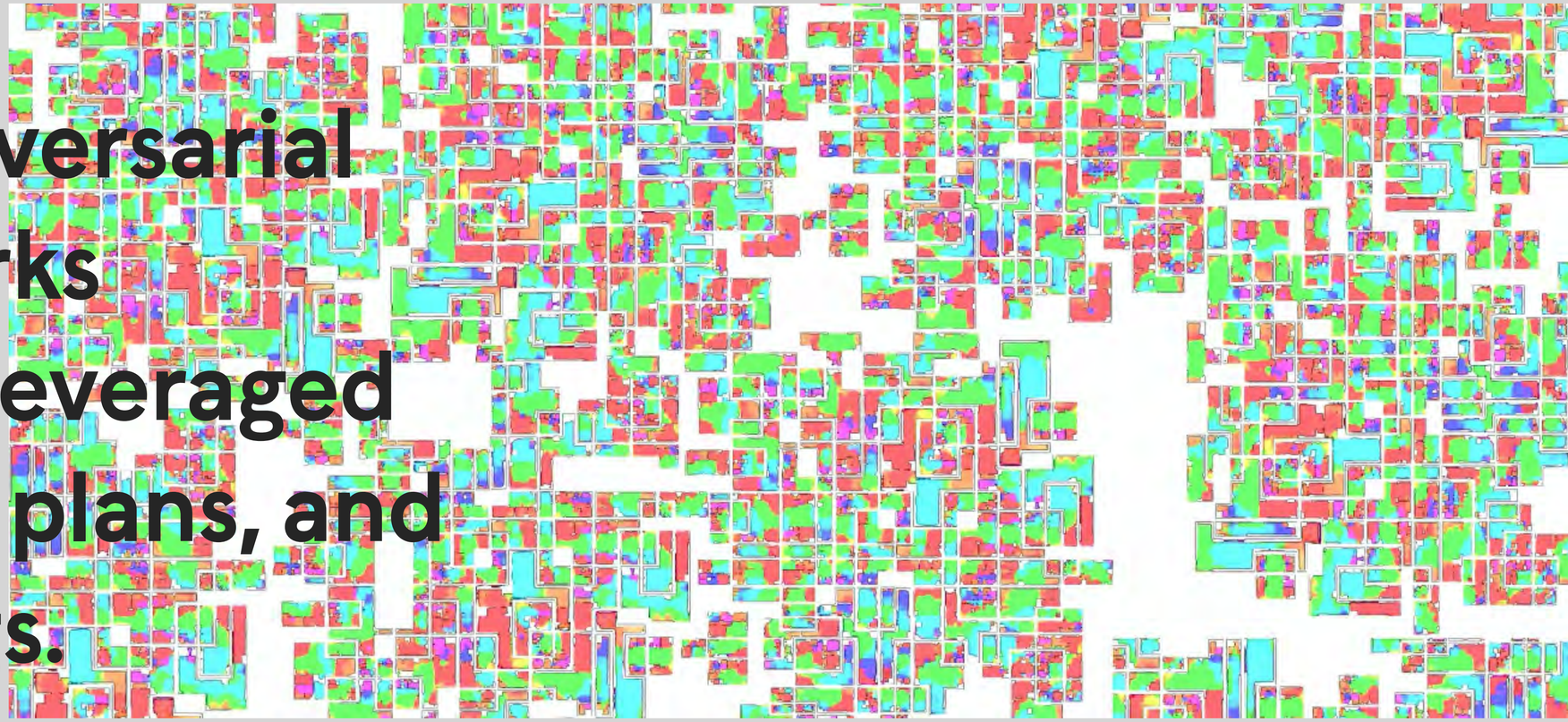
https://colab.research.google.com/drive/1IXI9cBgqS1_4A9Quhhve3B7uPMshauKX#forceEdit=true&offline=true&sandboxMode=true

cape
Campus Asia Plant Environment innovation

Yonsei University, Zhejiang University & Chiba University

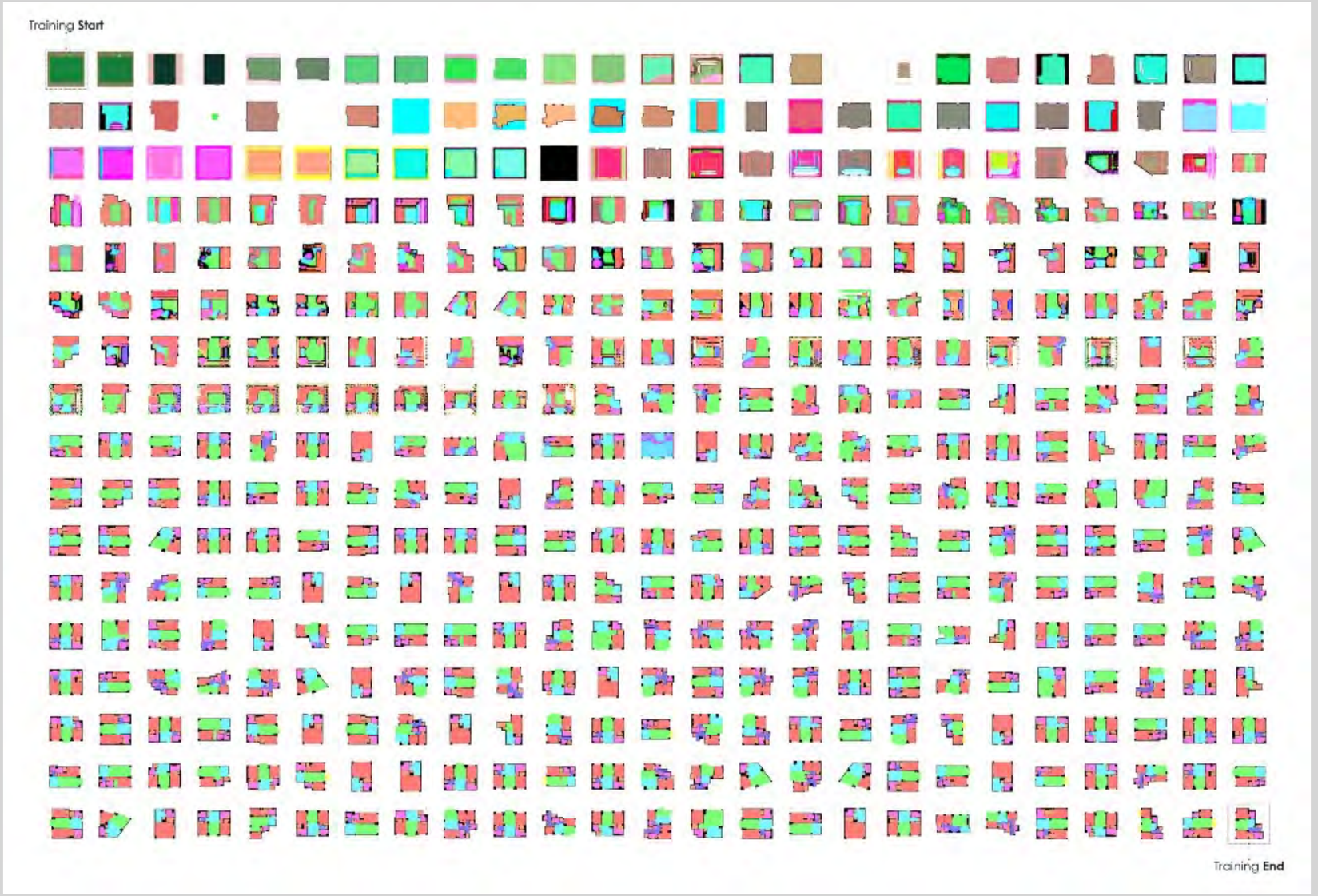
CAPE WS 2021

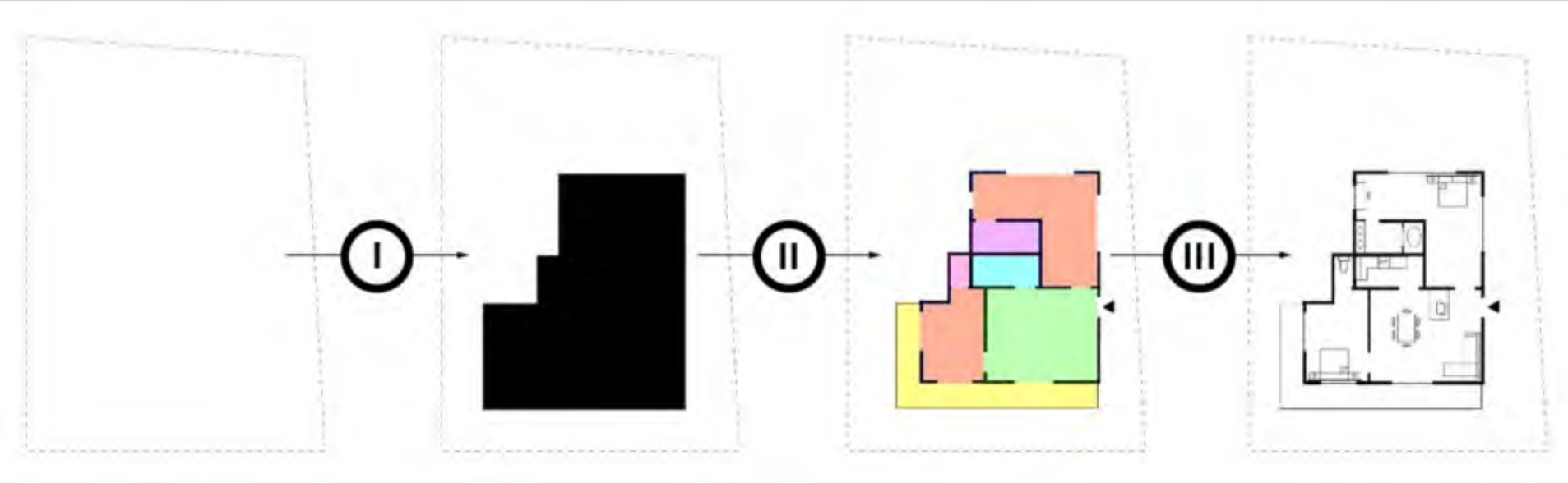
Generative Adversarial Neural Networks (or GANs) get leveraged to design floor plans, and entire buildings.



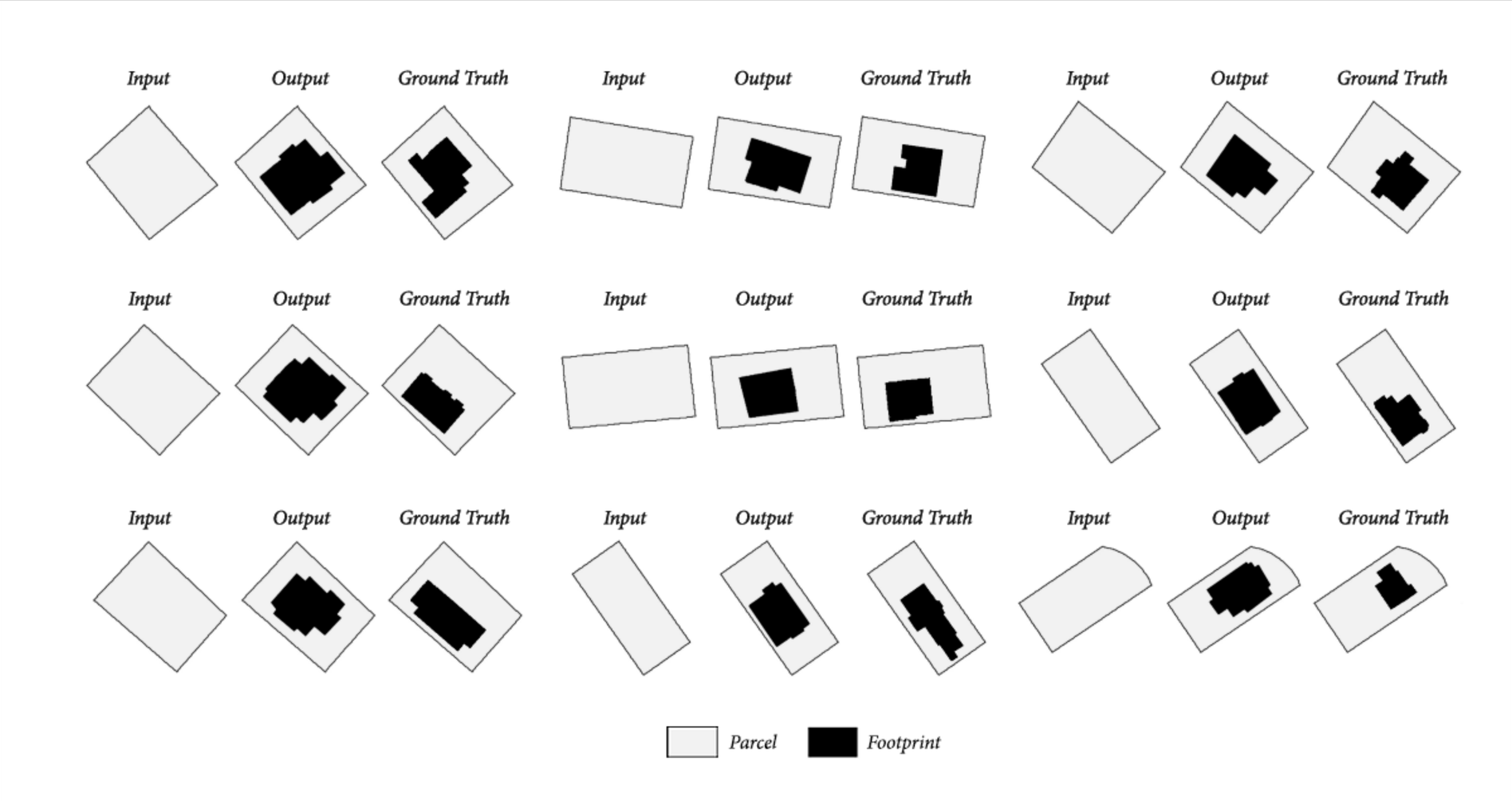
Stanislas Chaillou
Harvard Graduate School of Design

https://www.academia.edu/39599650/AI_Architecture_Towards_a_New_Approach

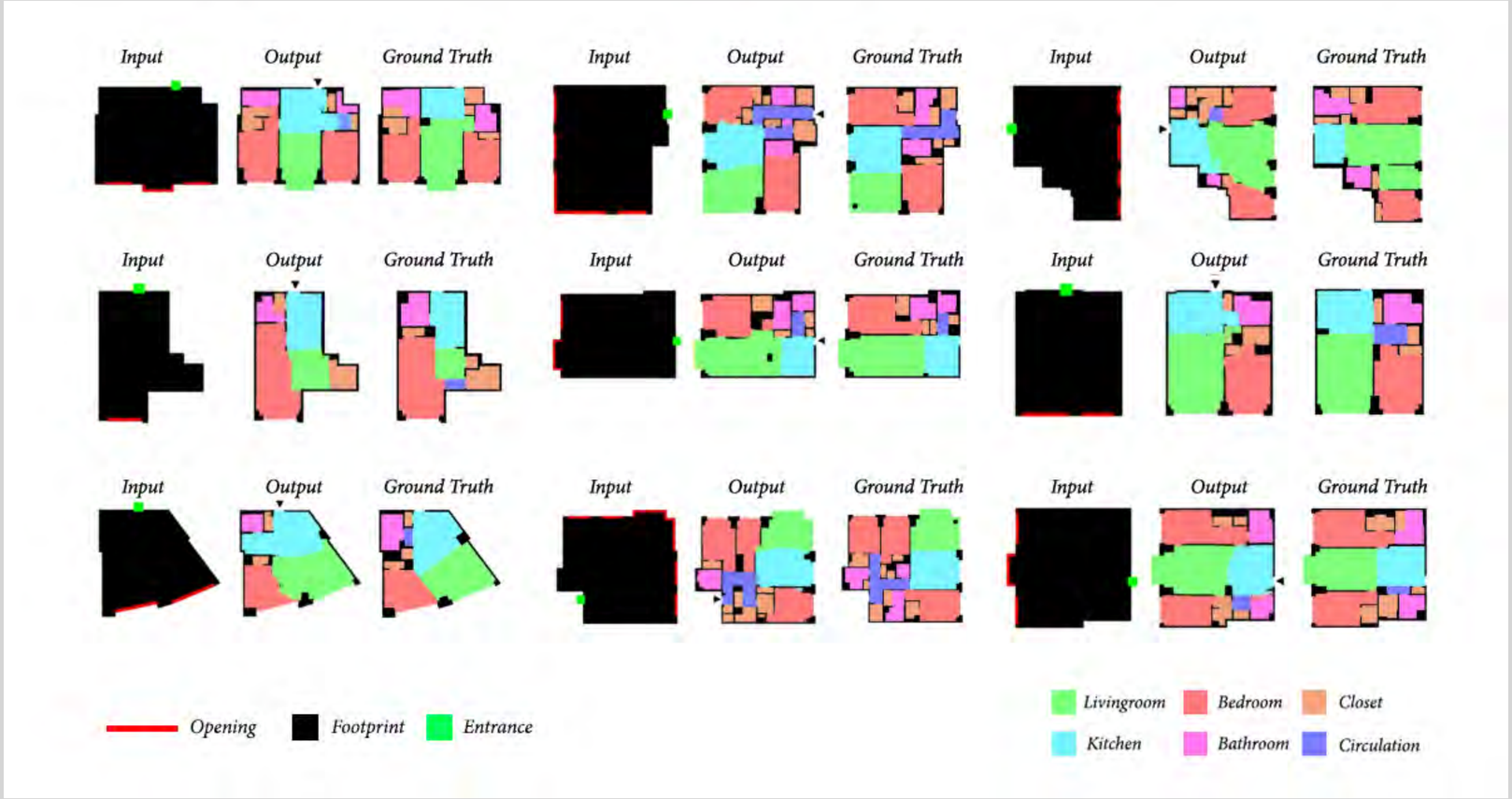

Training Start
Training End



(I) building footprint massing **(II)** program repartition **(III)** furniture layout



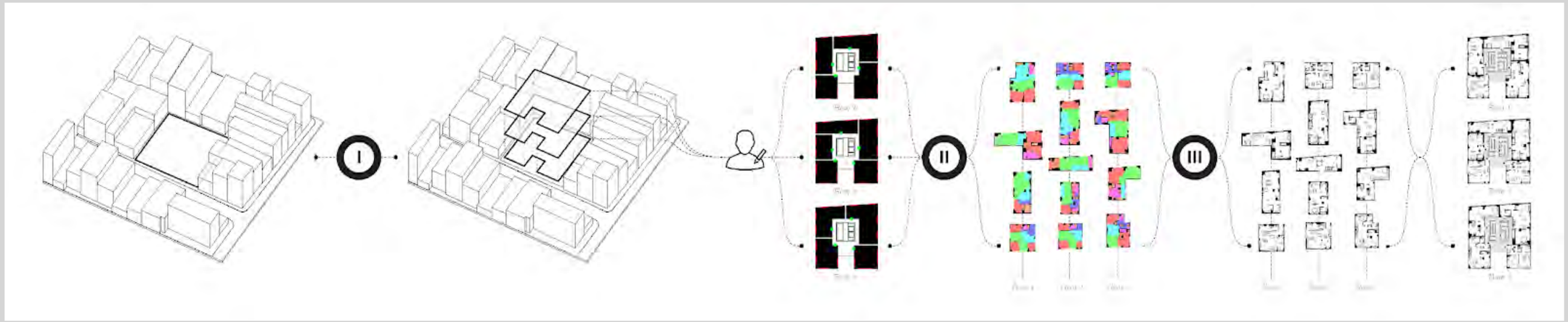
Results of model I



Results of model II

Results of model III

Model II Interface



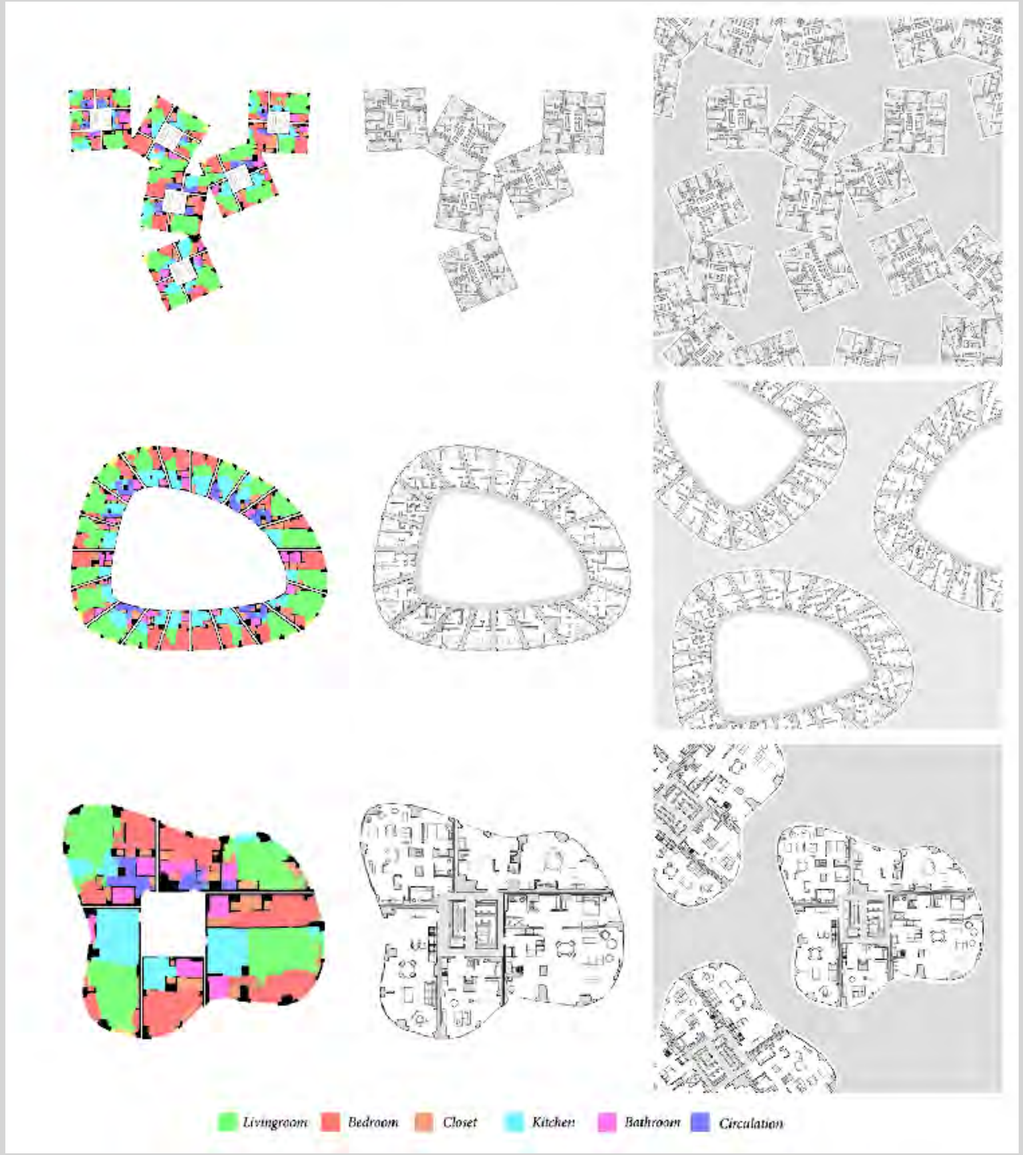
Apartment Building Generation Pipeline

- Floor 1 -

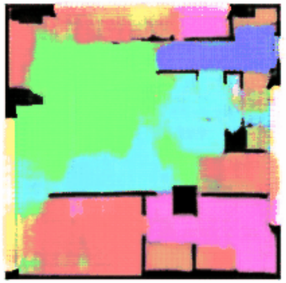
- Floor 2 -

- Floor 3 -

Apartment Building Generation



GAN-enabled building layouts






StyleGAN

Exercise

Today

Build Dataset

Weekend

Training the GAN

Monday

Check Results

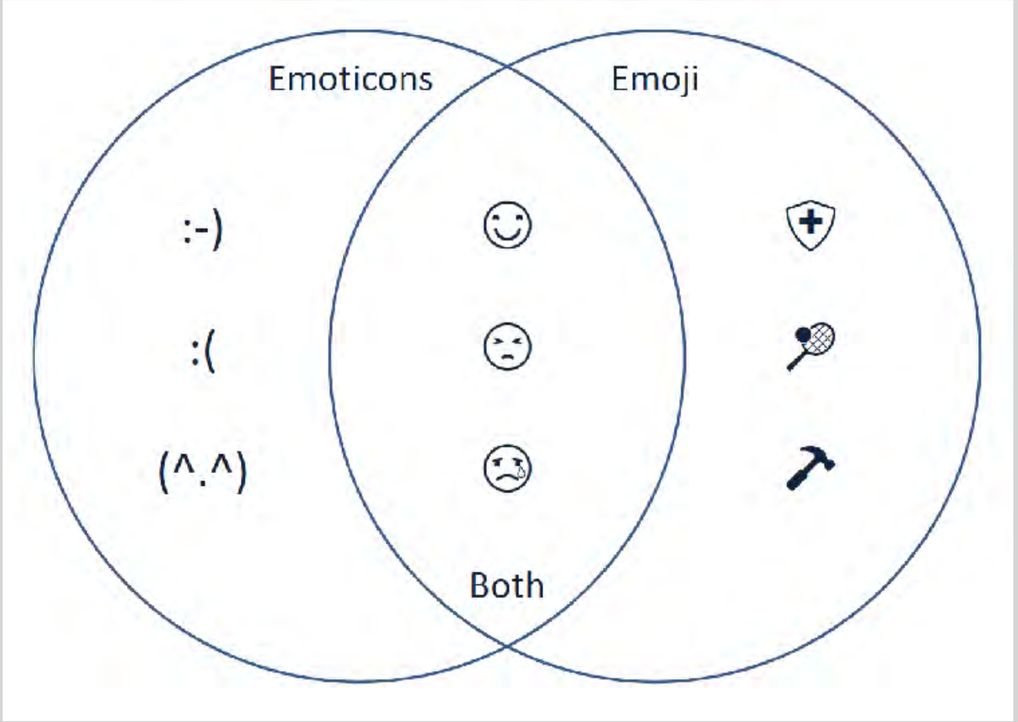
Plant Log (Tracking)

Day:—Picture —values—

Build Dataset

Emoji can be described as graphic symbols or ideogram that not only shows facial expression but also wider concepts and ideas such as celebrations, weathers, vehicles and buildings, food and drinks, animals and plants, emotions, feelings, and activities (Novak et al., 2015).

Emoticon can also be described as emotion used in text communication such as smile (Dresner and Herring, 2010). Emoticon is short character that use punctuation symbols in text message (Novak et al., 2015) and depicting emotional form in non-verbal language (Derks et al., 2018).



EMOJIS can influence communication

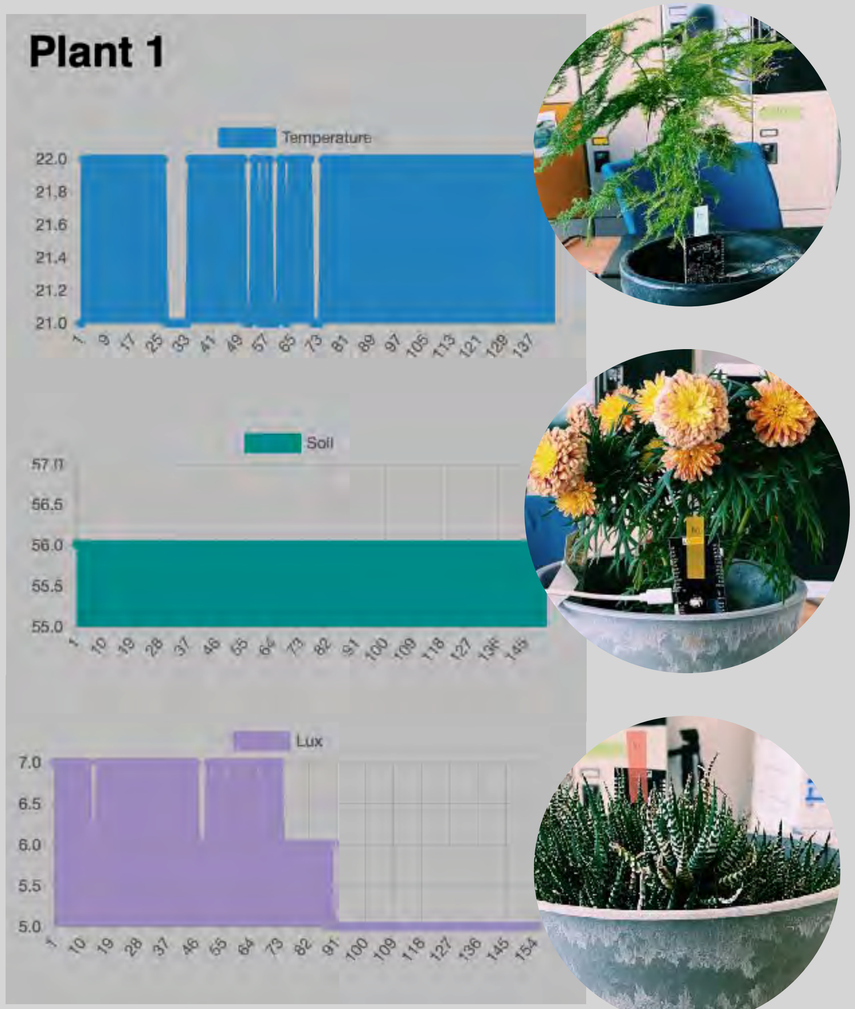
- Senders that use positive emojis are perceived as being warmer
- Emojis convey information about the sender's affect
- Congruent emojis enhance comprehension of text messages
- Users should favor positive emojis in their digital communications
- Emojis have the potential to enhance emotion communication, and in particular emotion inte

EMOJIS can influence communication

Individuals who use lots of positive emoticons on their Facebook page are perceived as more agreeable, conscientious, and open. (Wall et al. 2016)

However, the positive impact of emoticons on inferences about the personality of the sender may be limited to certain contexts. (Glikson et al. 2018).

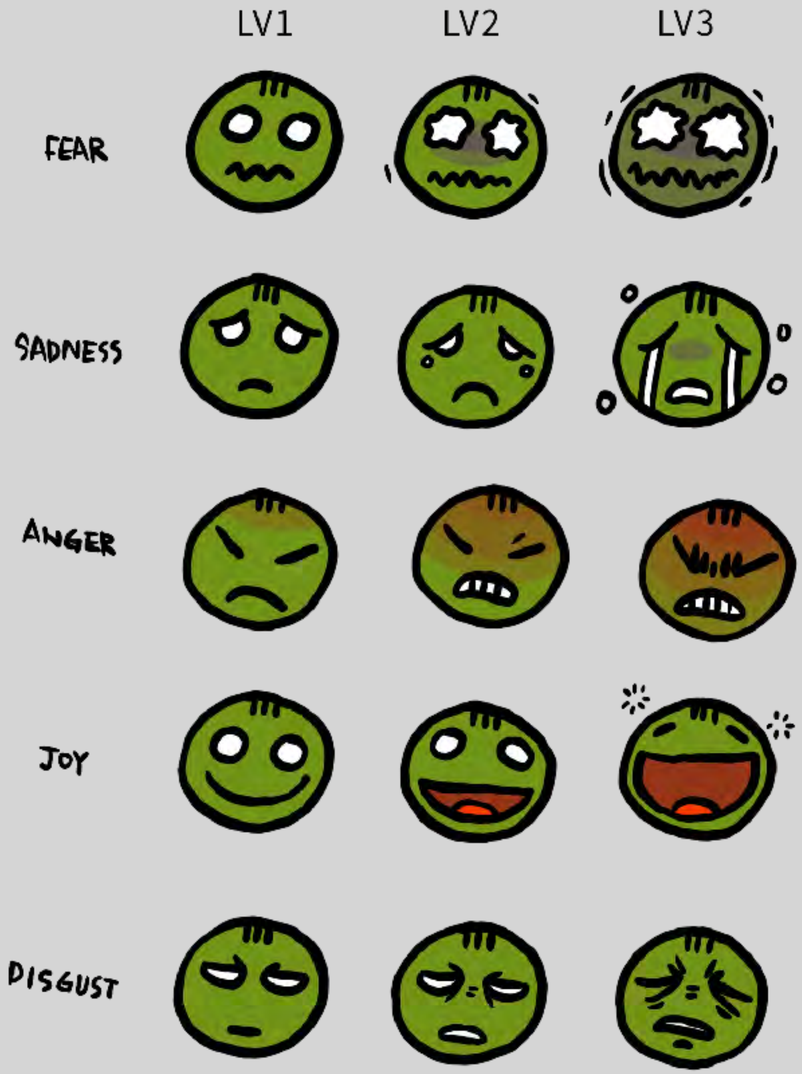
In professional contexts, smiling emoticons do not increase perceptions of warmth but instead decrease perceptions of competence.



Emotions That Human Can understand

Anger Disgust Fear Joy

Neutral Sadness Surprise


LV1
LV2
LV3
FEAR
SADNESS
ANGER
JOY
DISGUST

Build our own dataset

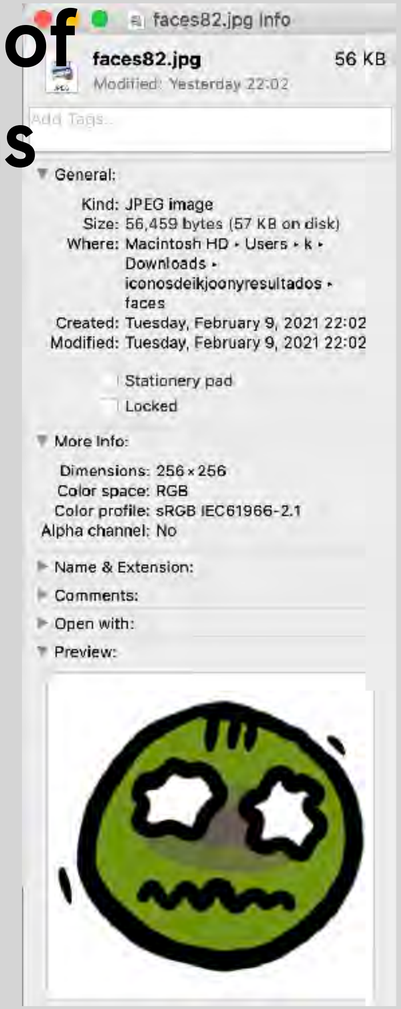
Select or Generate A set of Images

128px x 128px
.jpg (rgb)

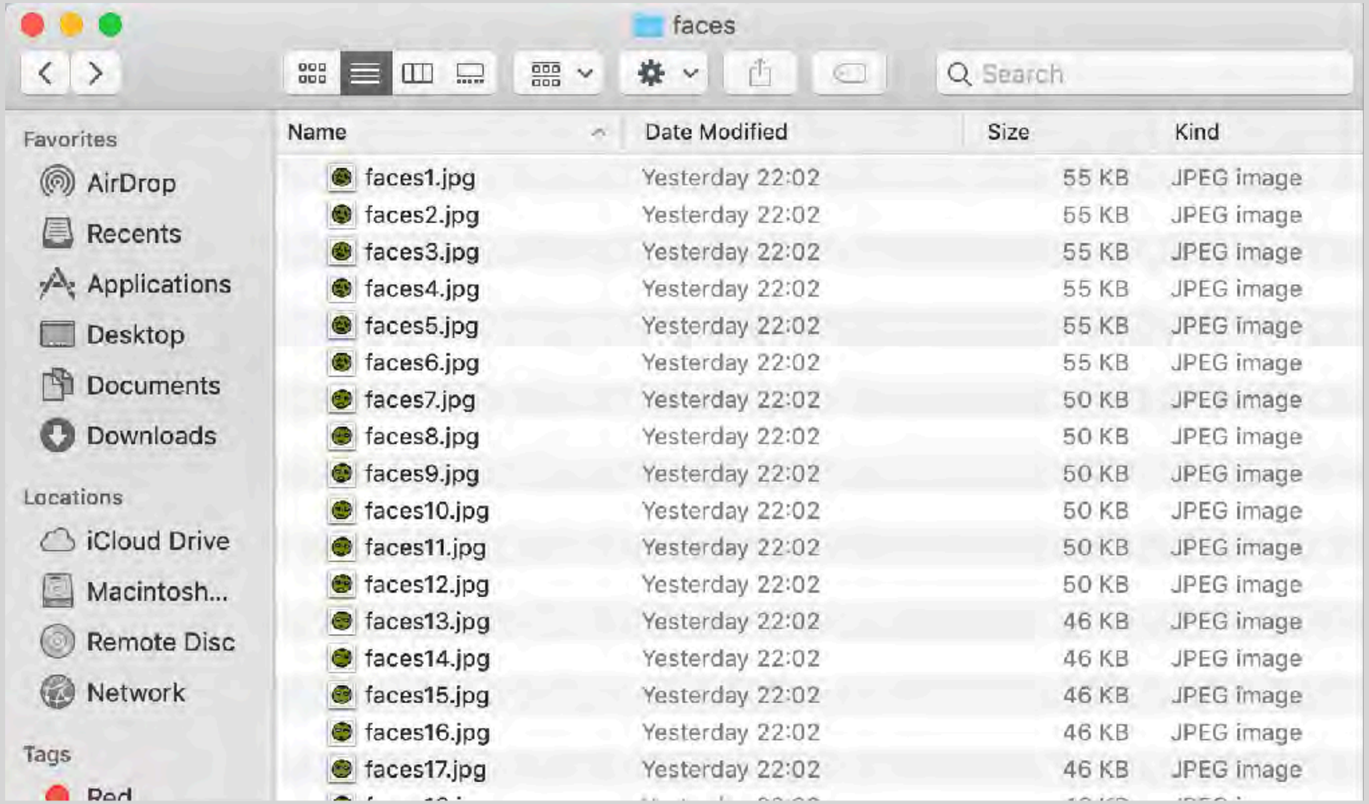
256px x 256px
.jpg (rgb)

The images must be square-shaped and they must all have the same power-of-two dimensions.

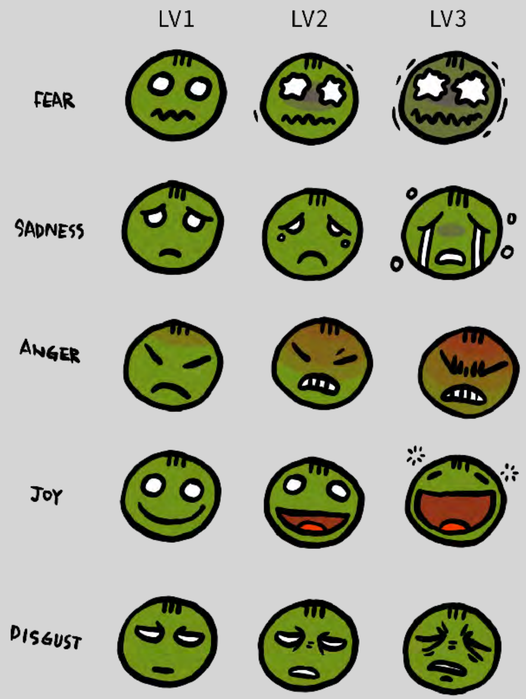
Make sure of Dimensions



All Images in a single folder



Every member Will create a set of Characters Representing 7 emotions In 3 different levels.



Total: 21 jpg images (256px x 256px) In a folder

Training the GAN

Weekend Exercise

Requirements and Preparation

Create Personal Dataset

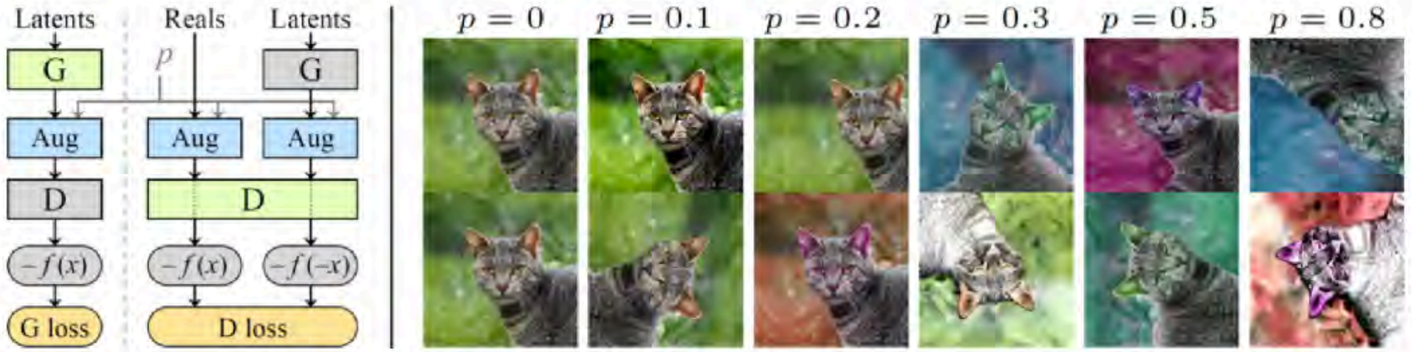
Requirements:

CUDA Toolkit

https://developer.nvidia.com/cuda-toolkit

Alternative:
Google Colab

StyleGAN2 with adaptive discriminator augmentation (ADA) — Official TensorFlow implementation



Training Generative Adversarial Networks with Limited Data

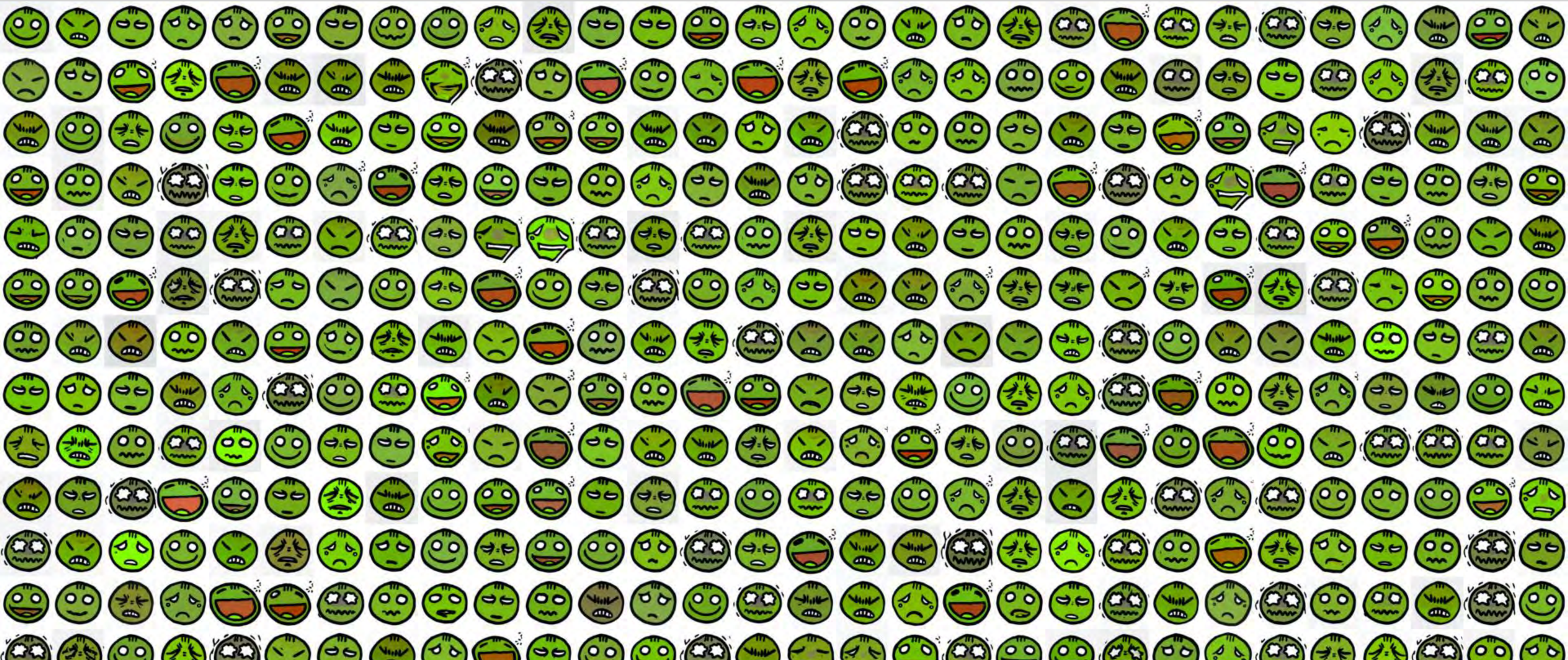
Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, Timo Aila

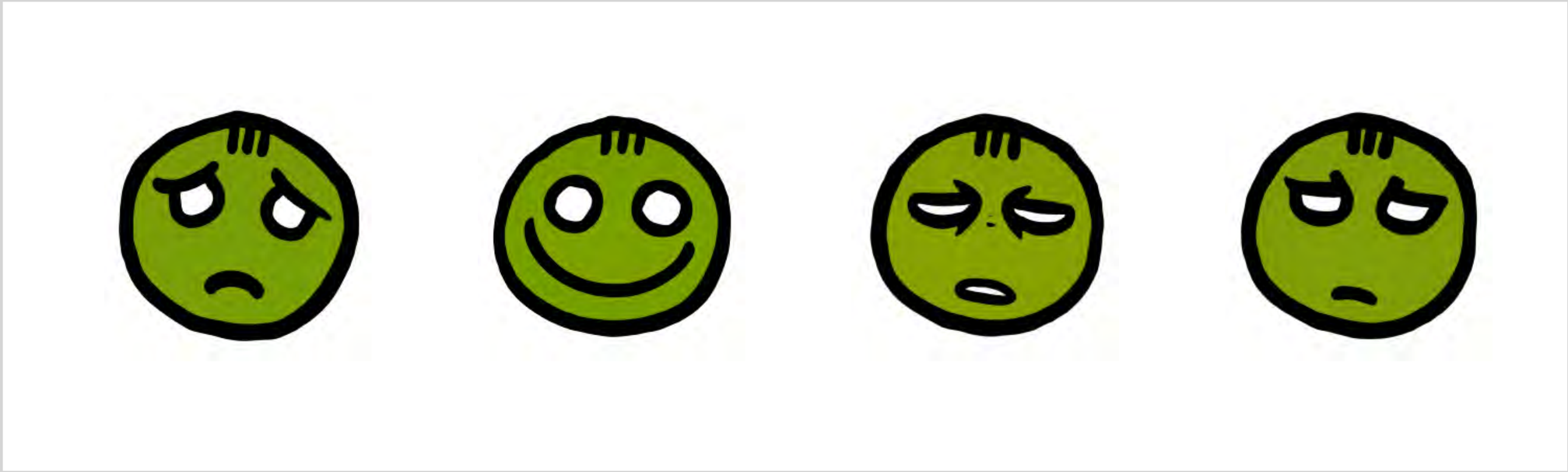
https://arxiv.org/abs/2006.06676

Abstract: *Training generative adversarial networks (GAN) using too little data typically leads to discriminator overfitting, causing training to diverge. We propose an adaptive discriminator augmentation mechanism that significantly stabilizes training in limited data regimes. The approach does not require changes to loss functions or network architectures, and is applicable both when training from scratch and when fine-tuning an existing GAN on another dataset. We demonstrate, on several datasets, that good results are now possible using only a few thousand training images, often matching StyleGAN2 results with an order of magnitude fewer images. We expect this to open up new application domains for GANs. We also find that the widely used CIFAR-10 is, in fact, a limited data benchmark, and improve the record FID from 5.59 to 2.42.*

https://github.com/NVlabs/stylegan2-ada

Output





Q&A

Questions & Answers

Cape Workshop 2021

	Thursday	Friday	Saturday	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday
Time JP	February 18	February 19	February 20	February 21	February 22	February 23	February 24	February 25	February 26
10:00-12:00			Trainning Day & Plant Tracking	Trainning Day & Plant Tracking		Students Team work	Students Team work	Students Team work	Students Team work
13:00-14:00	Introduction to the Workshop and Instructions. Students Presentation 5 to 10 Mins Presentation per School (School Introduction and team members)	Generative Design using GAN Lecture Part. 2			Showing Results of Trainning sessions and Tracking	Presentation of first Ideas	Generative Design Lecture Part.3	Preparation for Final Presentation	Preparation for Final Presentation
14:00-15:00	Presentation by Nagase Sensei & "Team formation"	Exercise using Generative Adversarial Networks			Conceptualization & preparation of first pitch presentation	Feedback Session	Design & Prototyping Team Work		
15:00-16:00	Introduction to Generative Design Lecture Part 1.				Team Work	Advise Session			Final Presentation
16:00-17:00		Closing Exercise 1							Feedback & Closure

* During the sessions marked in Yellow, Professors are Invited to Attend

Next Session:

Check Results
Check Tracking Plant Work in Proposals

See you tomorrow!

Tomorrow Q&A Session at 13:00 JPN

Contact us!

Hisa Nimi
hisanimi@chiba-u.jp
Design Researcher
Chiba University

Juan Carlos Chacón
juancarlos@chiba-u.jp
Design Researcher
Chiba University