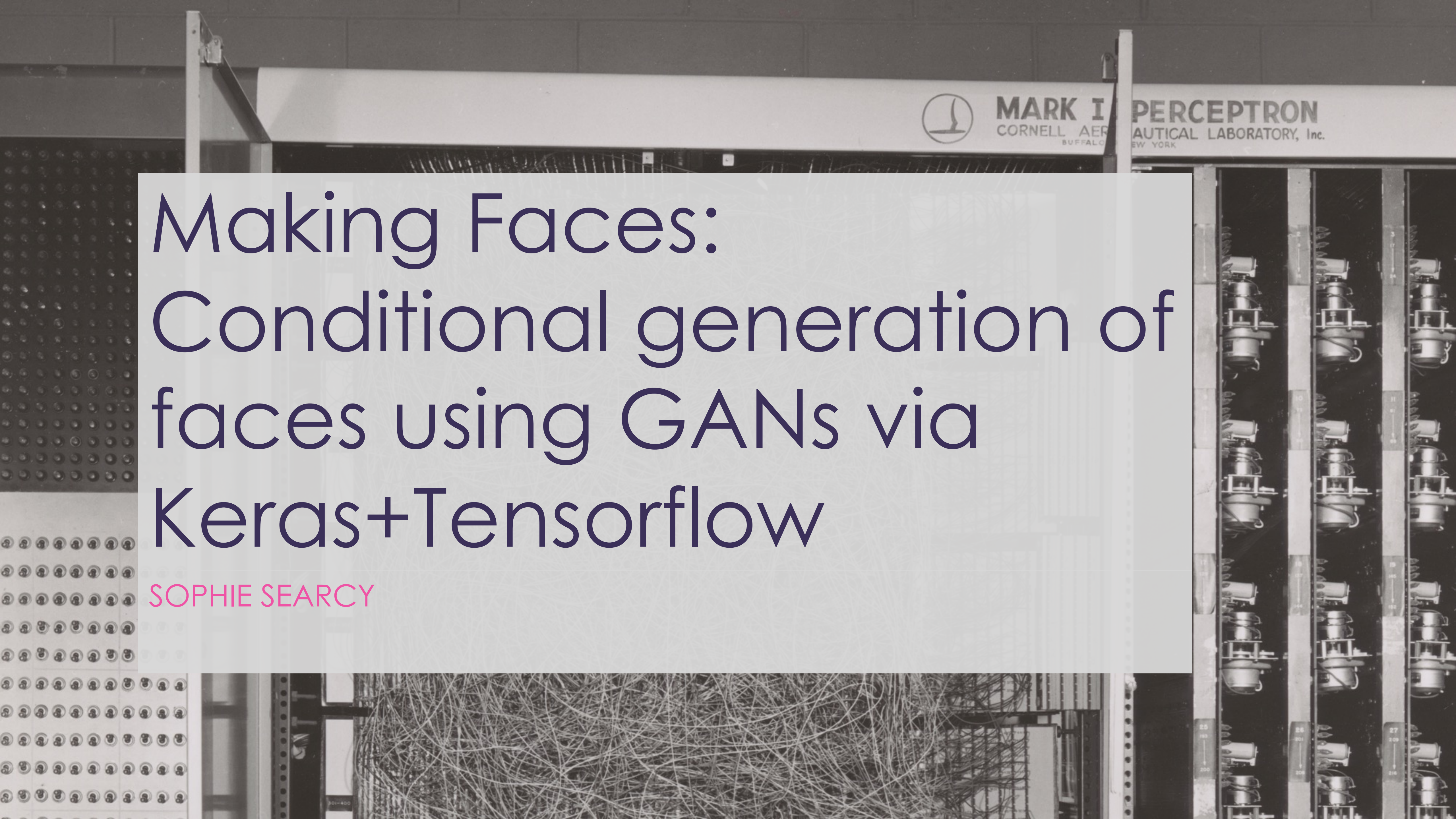


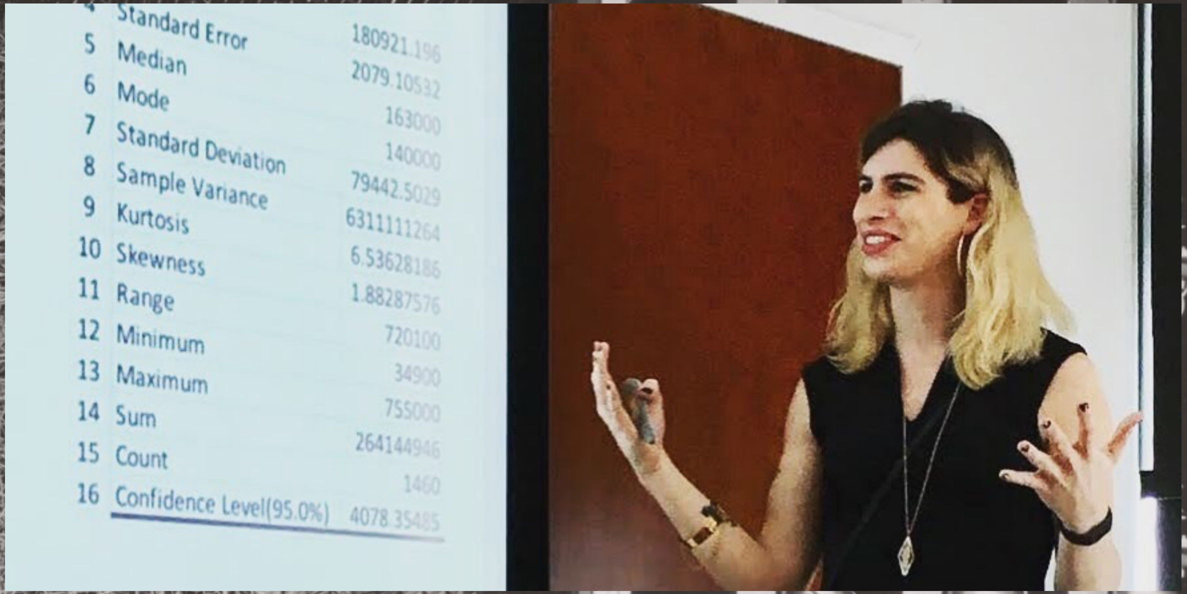
Making Faces: Conditional generation of faces using GANs via Keras+Tensorflow

SOPHIE SEARCY

Who am I?

Who am I?

- Sophie Searcy
- Curriculum development lead and Data Science Instructor at Metis
- Deep Learning and Data Science Ethics
- Write and lead free workshops with t4tech



METIS

t4tech

Signposting

Who is this for

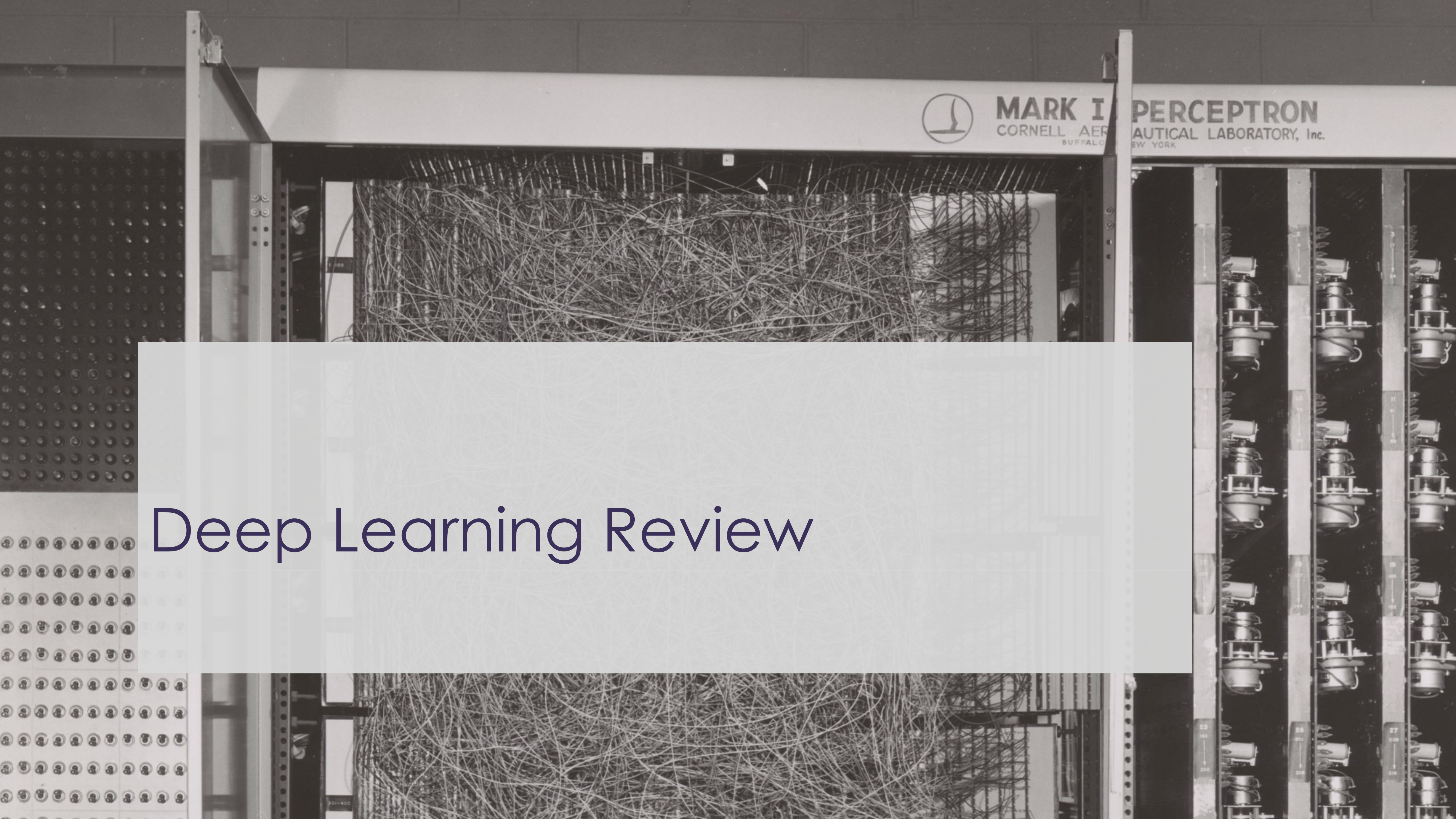
Someone who

- Understands and can explain the fundamentals of modern Deep Learning (there will be a review)
  - BackProp
  - Stochastic Gradient Descent
  - Common loss and activation functions
- Has built models using a recent Deep Learning package (PyTorch, Theano, Keras, etc.)

What we'll cover

Students should be able to:

- Understand and explain the important components of Generative Adversarial Networks
- Use provided boilerplate code and adapt it for new purposes
- State of The Art techniques in GANs:
  - Students will be exposed to a few important, recent developments.
  - Students will have the building blocks needed to independently explore new techniques.



Deep Learning Review

What makes deep Learning special?

Typical Machine Learning

Data

Transformations	Feature engineering, feature extraction	Tuned by hand, parameter search
Model	Linear model, SVM, RF, etc.	Optimized wrt objective function

Output

What makes deep Learning special?

Deep Learning

- Data
- Transformations
- Model
- Output

Deep Learning model

Optimized wrt objective function

Essential parts: Differentiable functions

- $h_0 = f(W_0^T x + b_0)$
- $h_1 = f(W_1^T h_0 + b_1)$
- …
- $y = f(W_n^T h_n + b_n)$
- DL models use these functions to process data in steps from input → output
  - Traditional application:
    - Tabular data → Regression/Classification
  - New (ish) applications
    - Image → Text
    - Image → Image

Essential parts: Stochastic Gradient Descent + BackProp

Gradient Descent

- Finds adjustment to function parameters that minimizes the loss function

Back Propagation

- Chain rule of calculus in algorithm form.
- Applies gradient descent over many layers of a network.

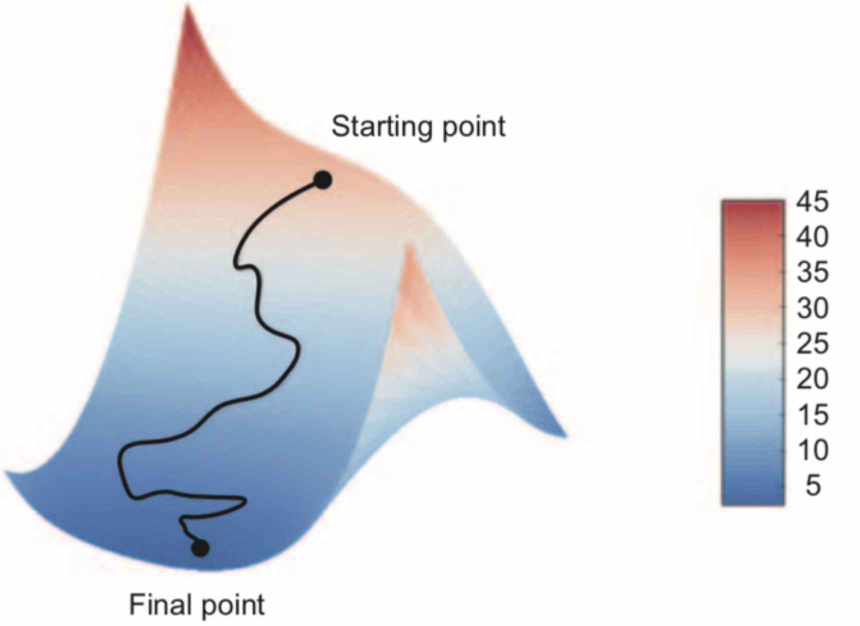


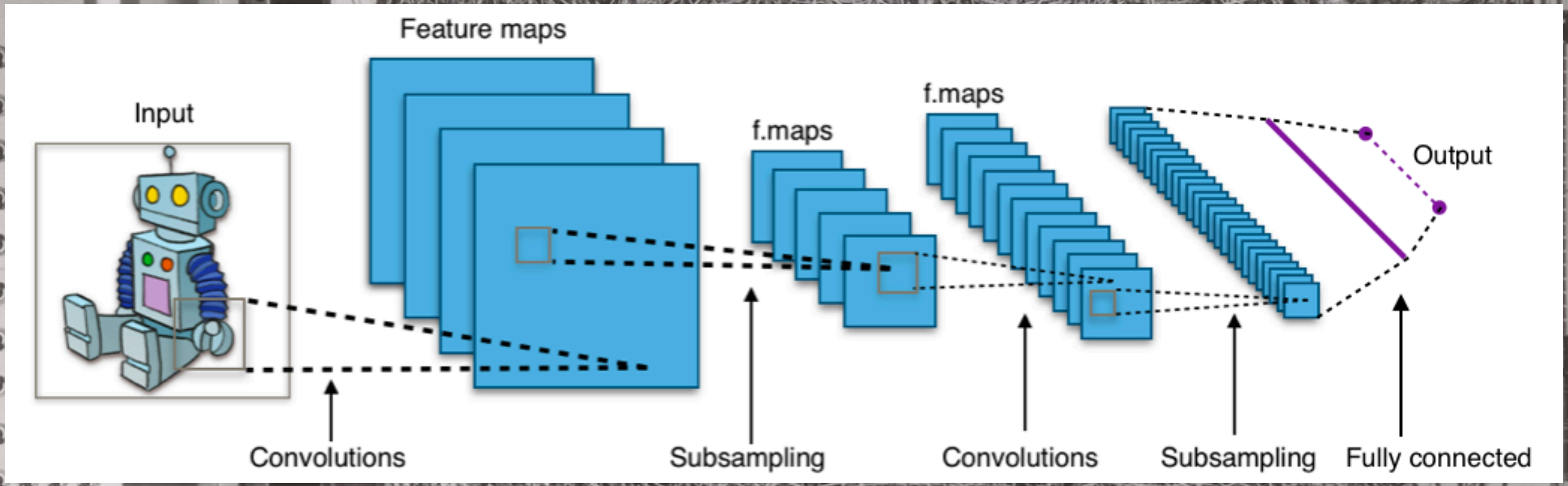
Figure 2.12 **Gradient descent down a 2D loss surface (two learnable parameters)**

Chollet 2018

GAN Overview

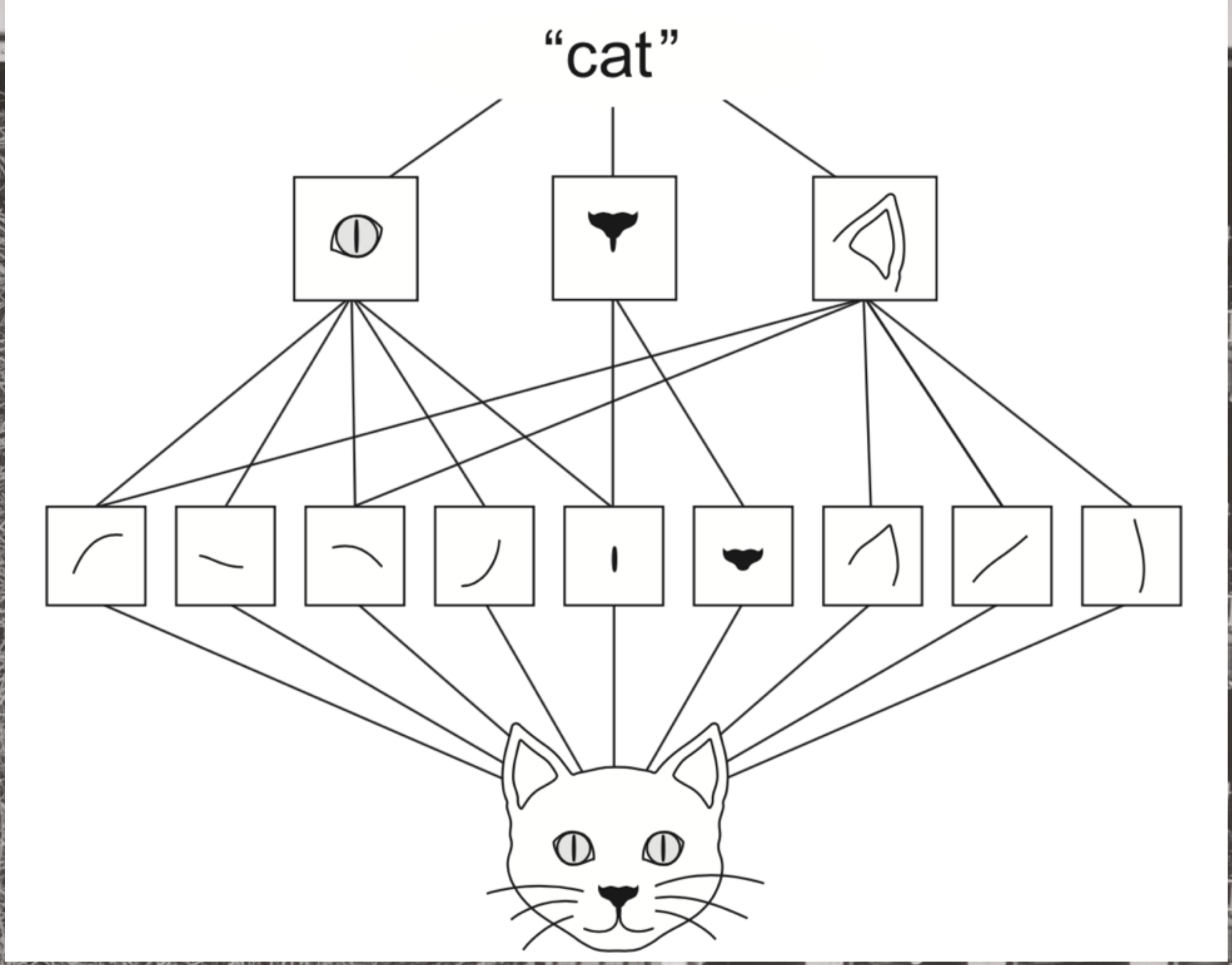
Convolutional Classifiers

- Convolutions learn feature maps
- Use sampling/pooling to summarize over height and width of image
- Output is some classification vector, e.g. probabilities



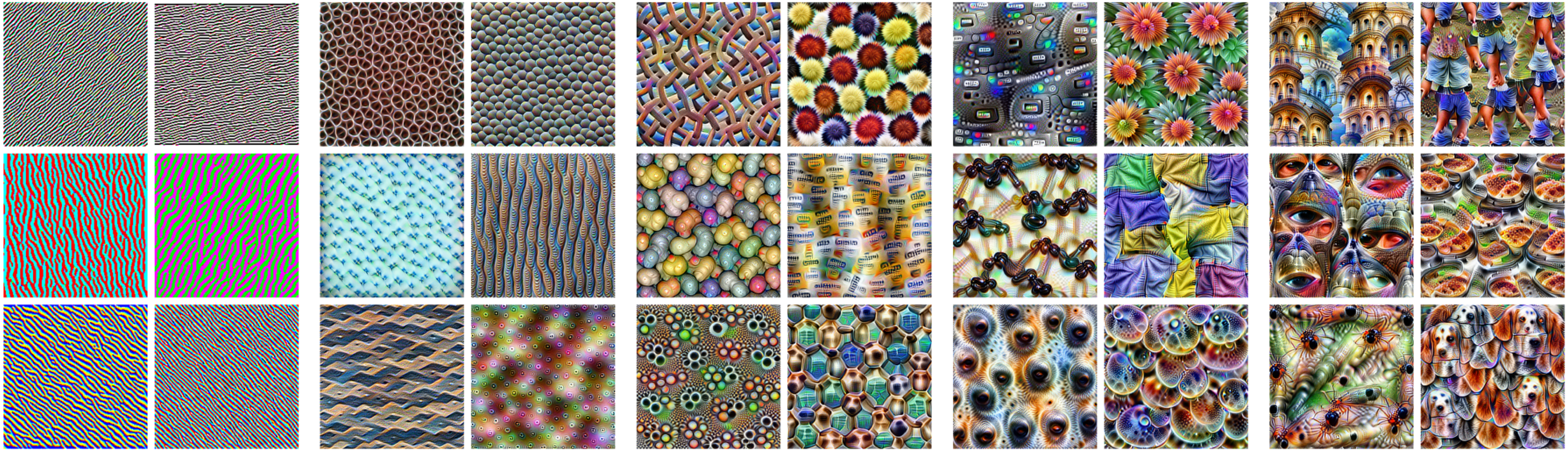
source

- Convolutional filiters
- Pixels → subparts → parts → whole



Chollet 2018

- Convolutional filters
- Pixels → subparts → parts → whole
- We can visualize this progression by finding input that maximizes activity at layer

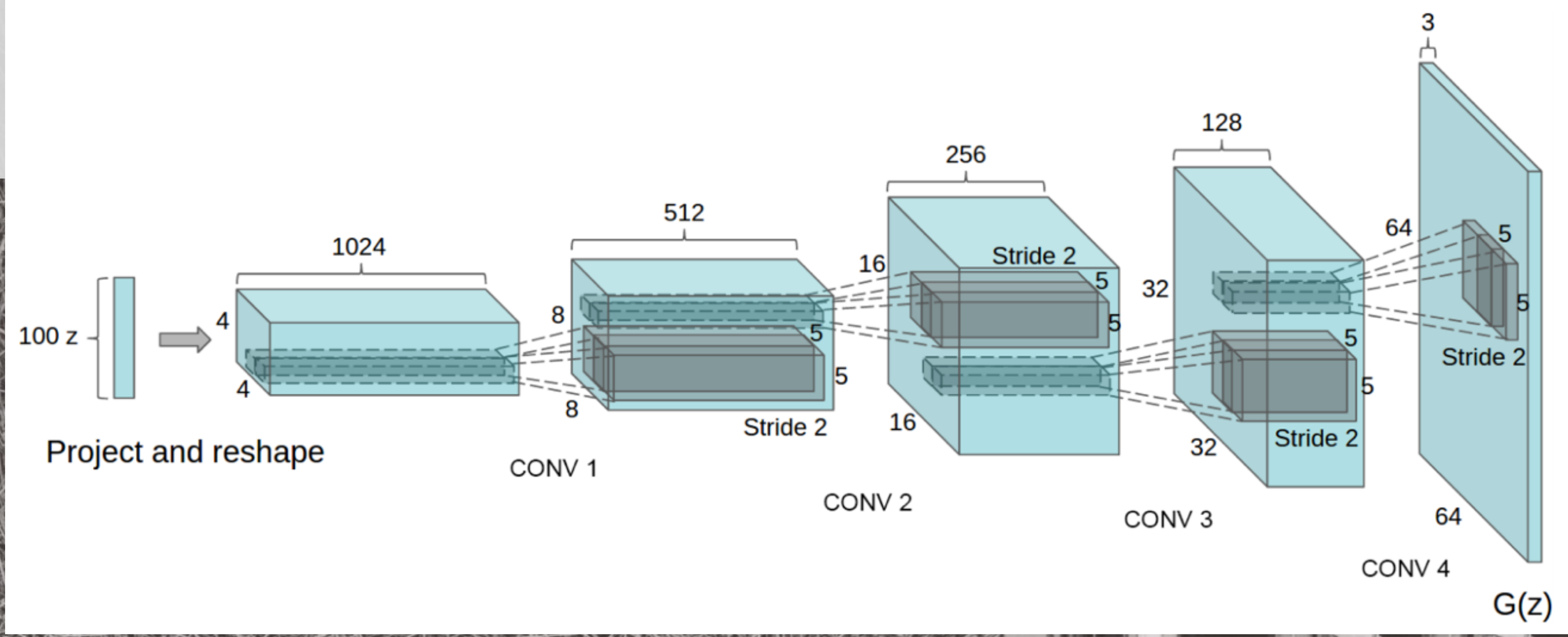


Edges (layer conv2d0) **Textures** (layer mixed3a) **Patterns** (layer mixed4a) **Parts** (layers mixed4b & mixed4c) **Objects** (layers mixed4d & mixed4e)

Olah 2017

Convolutional Generation

- Convolutions learn feature maps
- Upsampling/DeConvolution progressively grow image



Radford et al 2015

GAN Architecture

D(x) - Discriminator

- Given image
- Attempts to classify as fake or real

GAN Architecture

G(z) - Generator

- Given random vector z
- Attempts to generate an image that fools D(·)

GAN Architecture

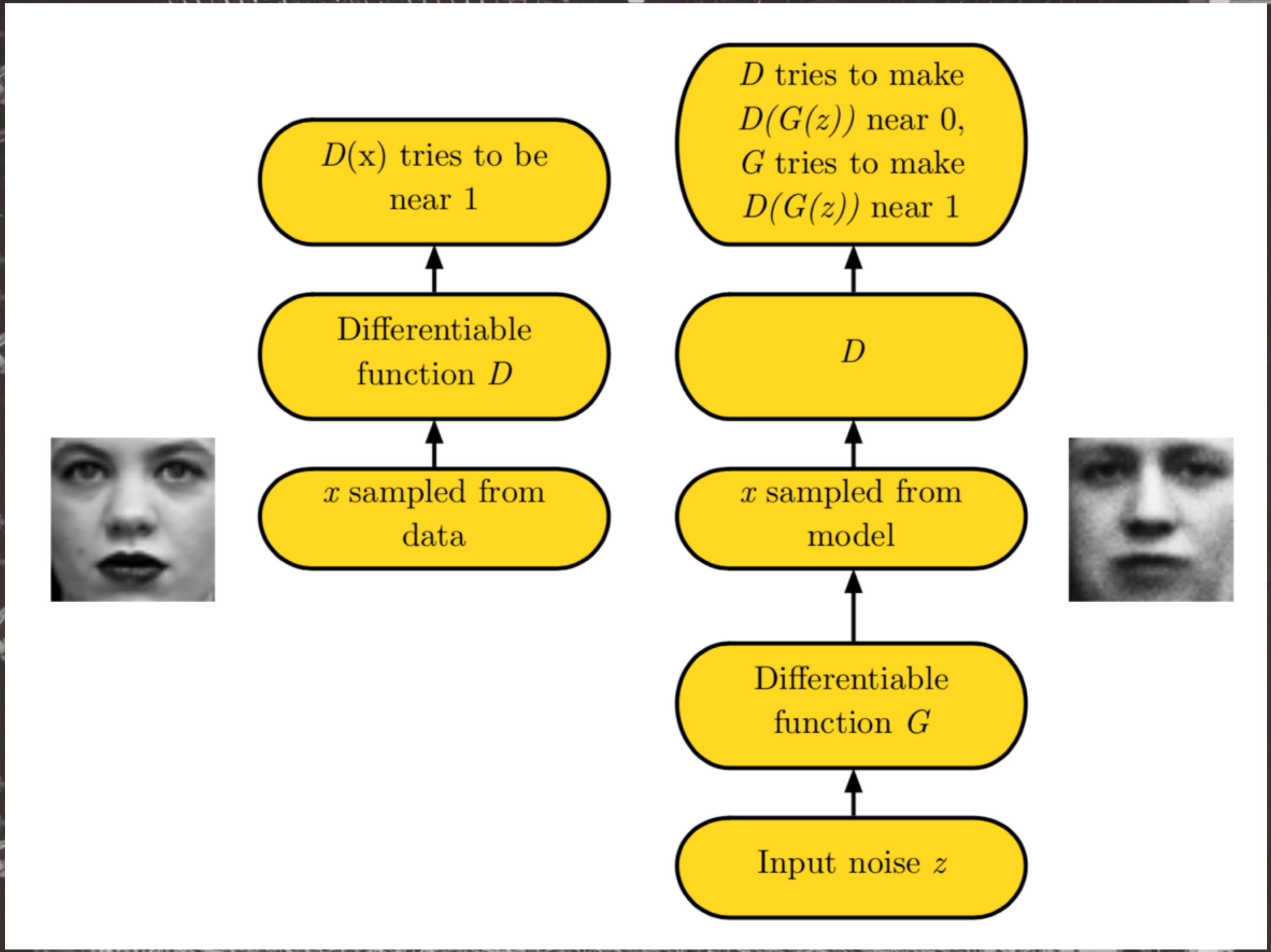
G(z) - Generator

- Given random vector z
- Attempts to generate an image that fools D(·)

D(x) - Discriminator

- Given image
- Attempts to classify as fake or real

GAN Architecture



Goodfellow 2016

Generator task

G(z) - Generator

- Given random vector z
- Attempts to generate an image that fools D(·)

Imagine you are the generator

- CIFAR image data (32x32)
- Generate a frog that will fool the discriminator

Generator task

You are the generator

- Imagine you can see the training data.
- You can learn as much as you want from the training data.
- You have to devise a strategy to trick the Discriminator.
- What is your strategy for fooling the discriminator?
  - i.e. what if you *had* to say/write pseudocode for the best strategy in a minute or so?

Generator task

What is your strategy?

- Memorize training images?
  - You have ~ 1 million parameters but the training data has ~ 100 million pixels

Generator task

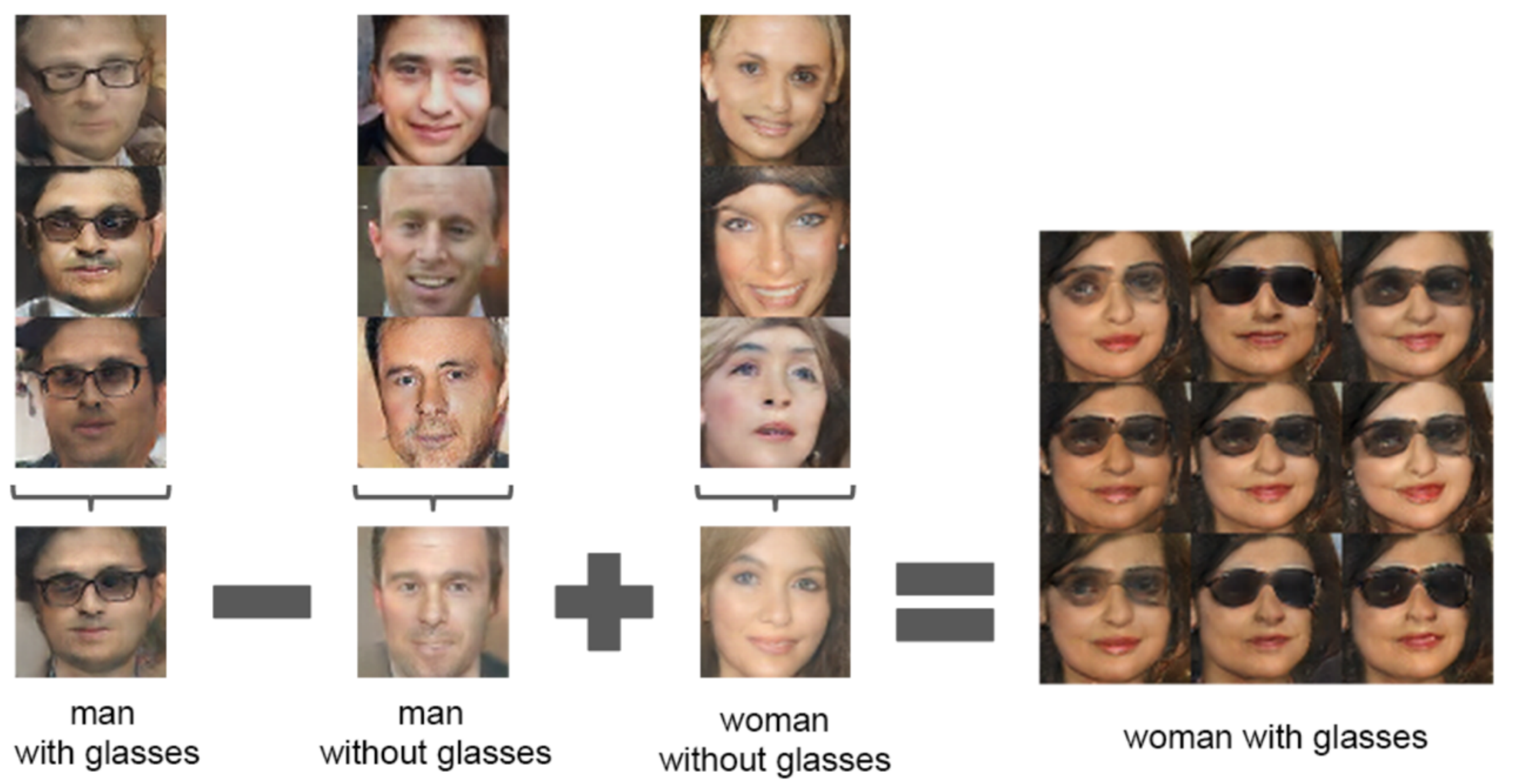
What is your strategy?

- Memorize training images?
  - You have ~ 1 million parameters but the training data has ~ 100 million pixels
- Instead the generator learns the *distribution* of the training data.
  - Attempts to generate an example from that distribution

Distribution Learning

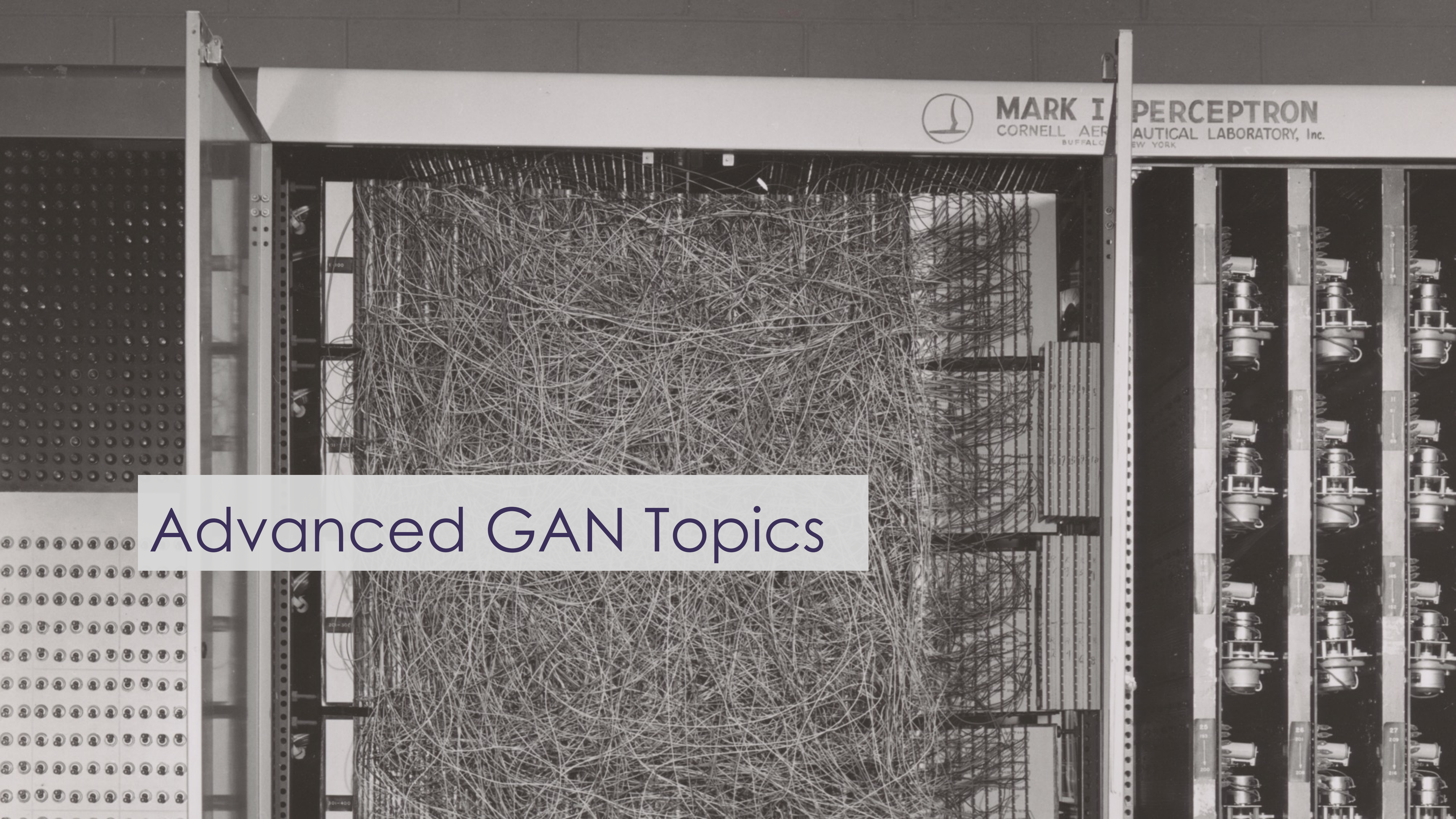
Generator learns *distribution* of training data

- Meaningful understanding of that training data



Radford et al 2015

Advanced GAN Topics

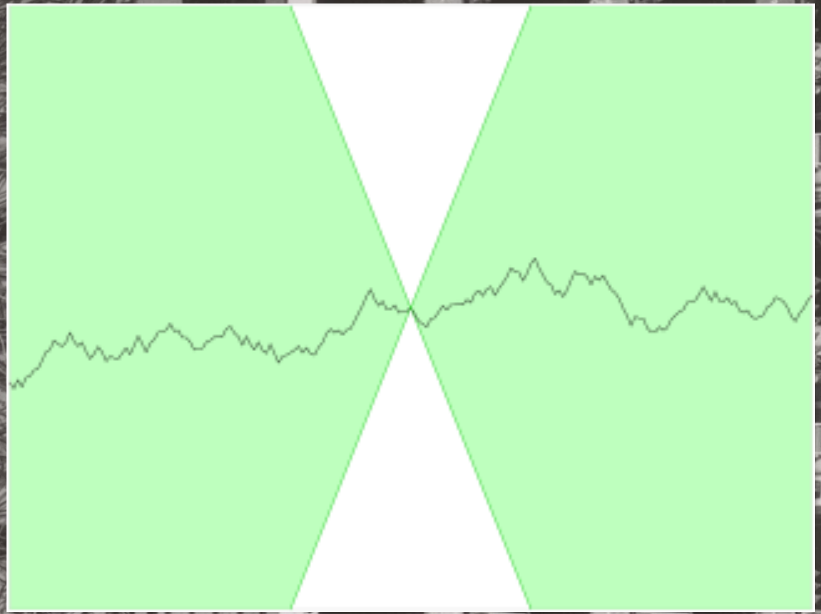


Lipschitz Continuity

- Problem: in many cases, the discriminator can be essentially impossible for the generator to beat.
  - Impossible to win → zero gradient → no learning

Lipschitz constant: Maximum rate of change of a function

Spectral Normalization (Miyato et al 2018) constrains the Lipshitz constant of the discriminator, ensuring stable training of generator.

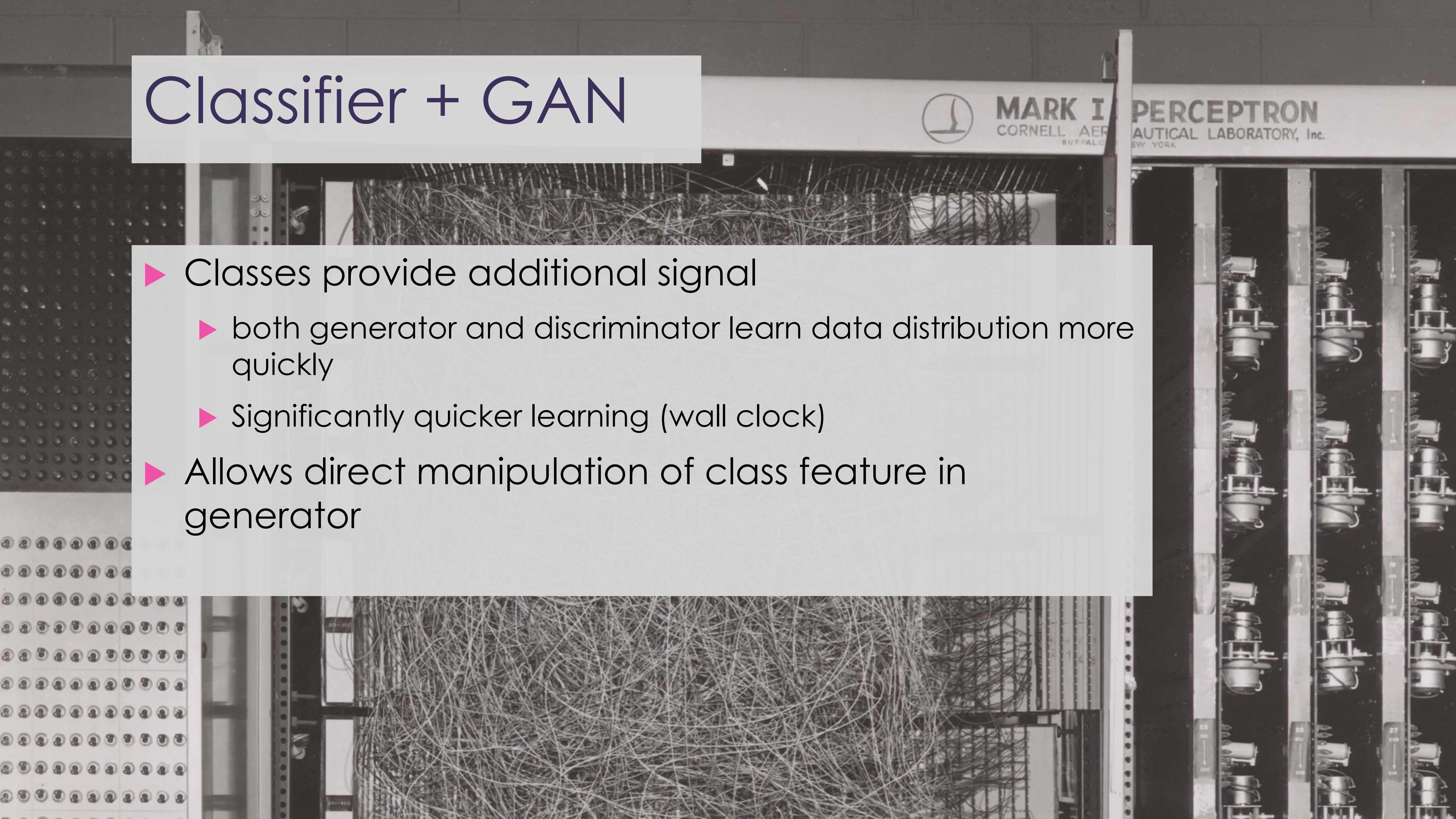


source

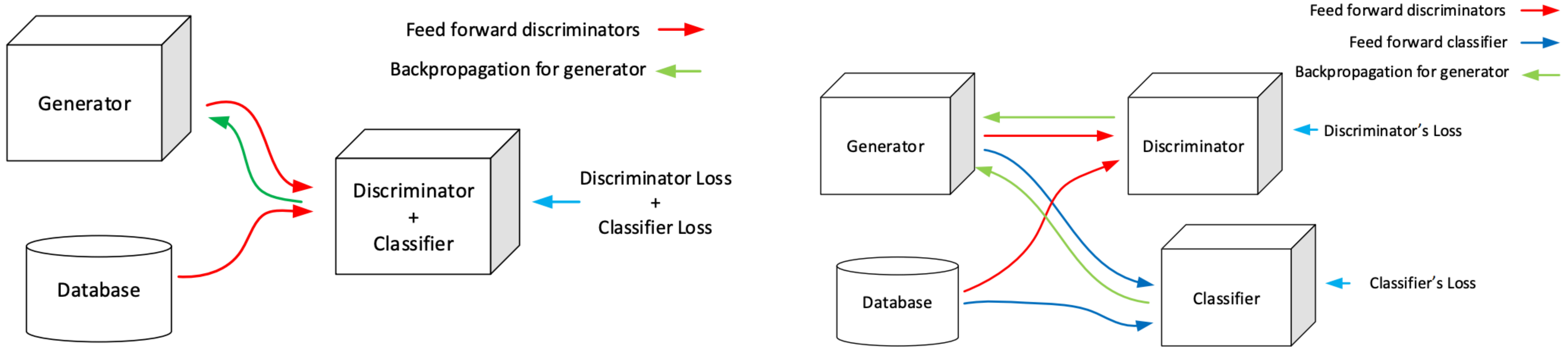
Multilabel Conditional GAN

Classifier + GAN

- Classes provide additional signal
  - both generator and discriminator learn data distribution more quickly
  - Significantly quicker learning (wall clock)
- Allows direct manipulation of class feature in generator



VAC GAN

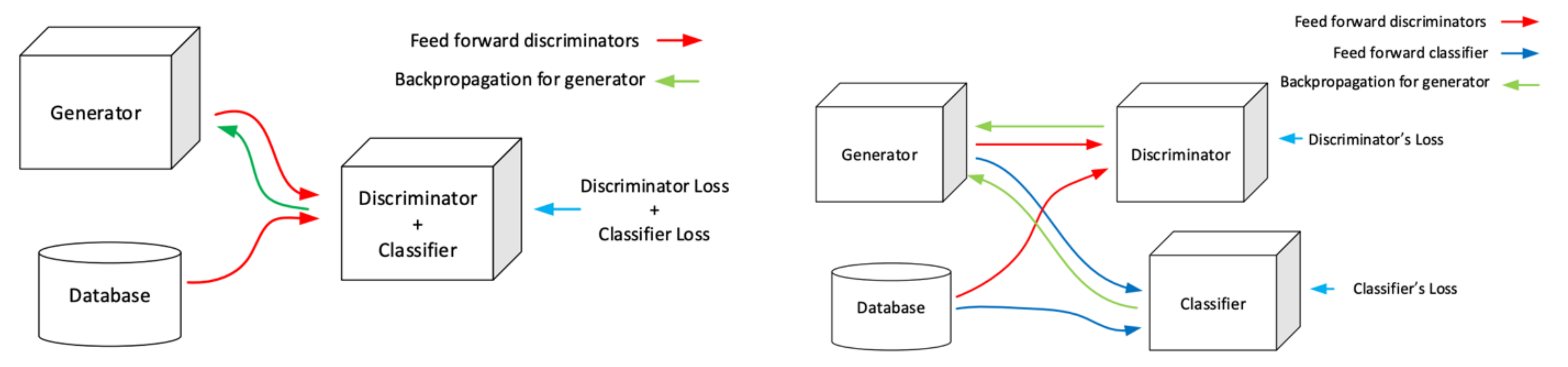


(a) The ACGAN scheme. (b) The presented scheme (VAC+GAN)

Bazrafkan 2018

VAC GAN

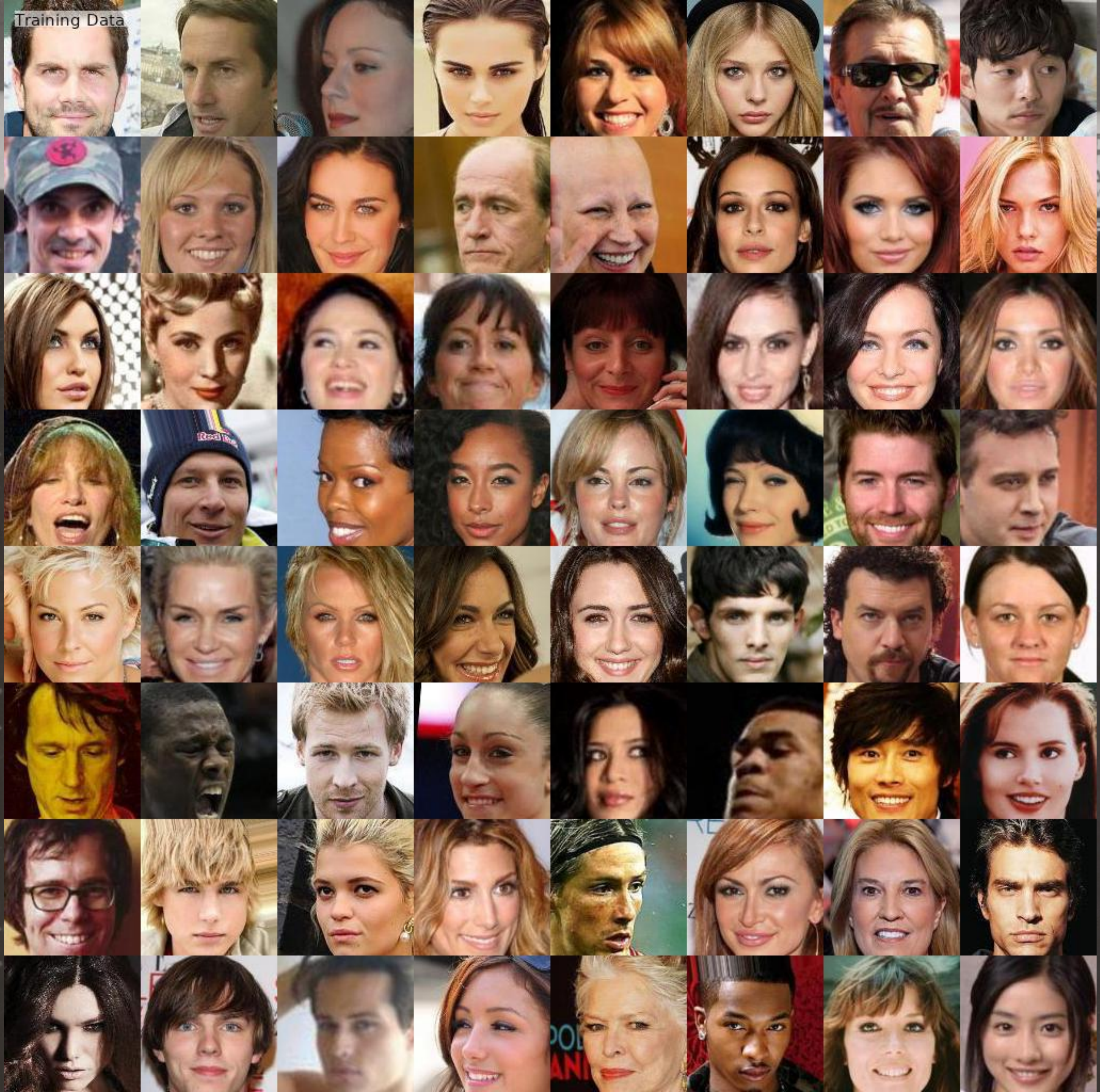
- VAC GAN
  - Good: Versatile classification with GAN
  - Bad: Requires a 3rd model
- Today's demo: VAC-GAN variant that combines Discriminator and Classifier



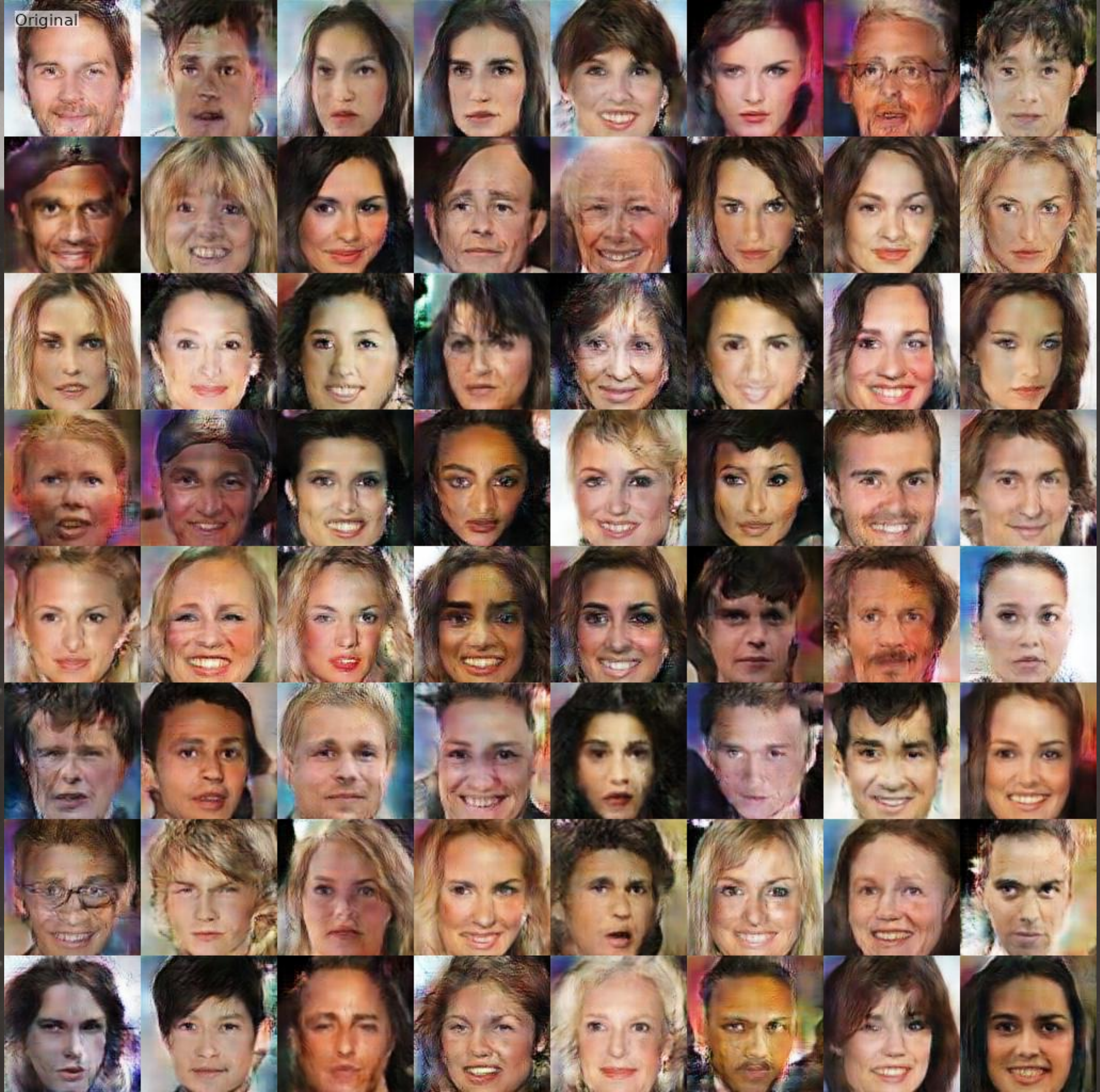
(a) The ACGAN scheme. (b) The presented scheme (VAC+GAN)

Bazrafkan 2018

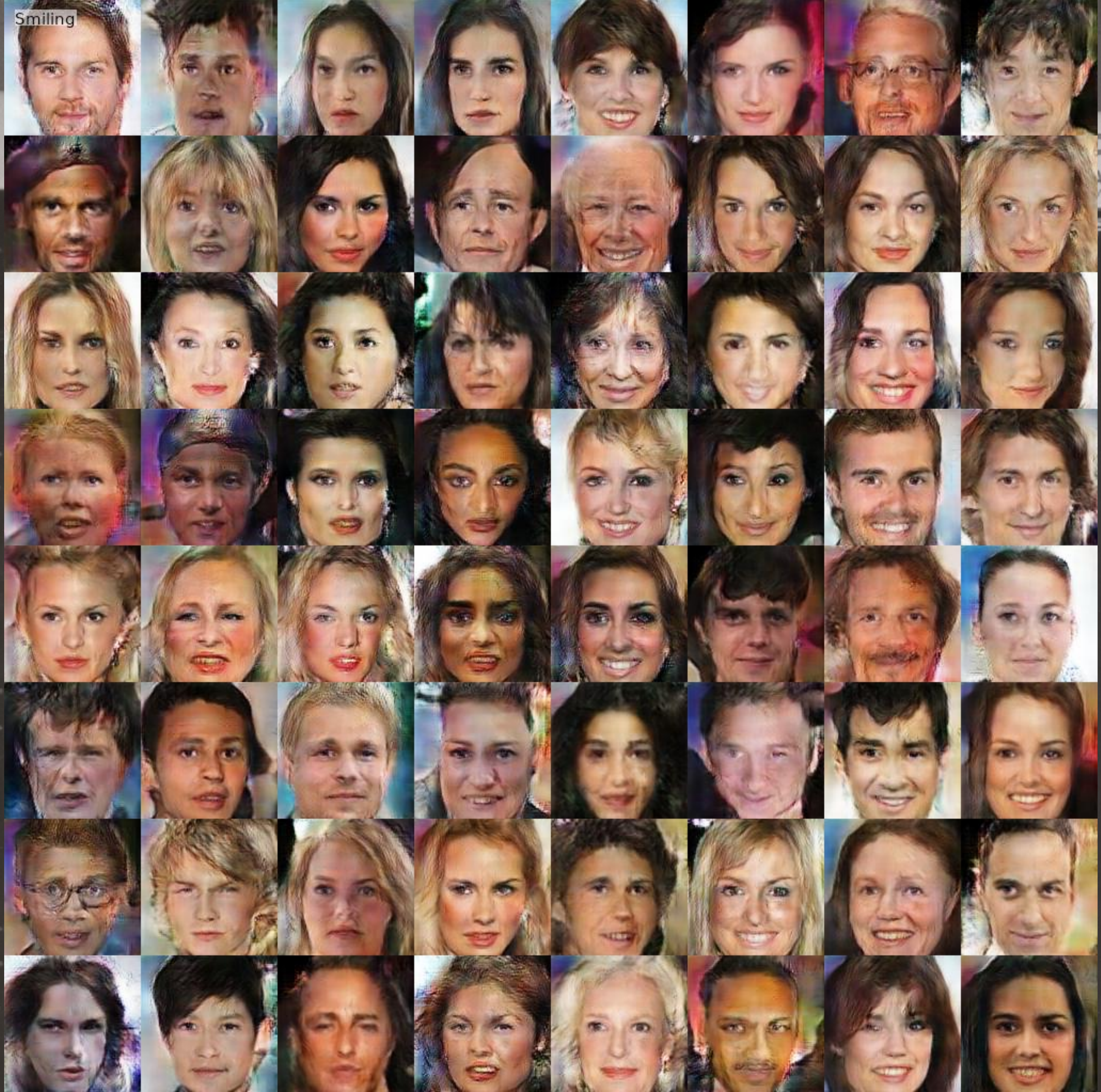
Results



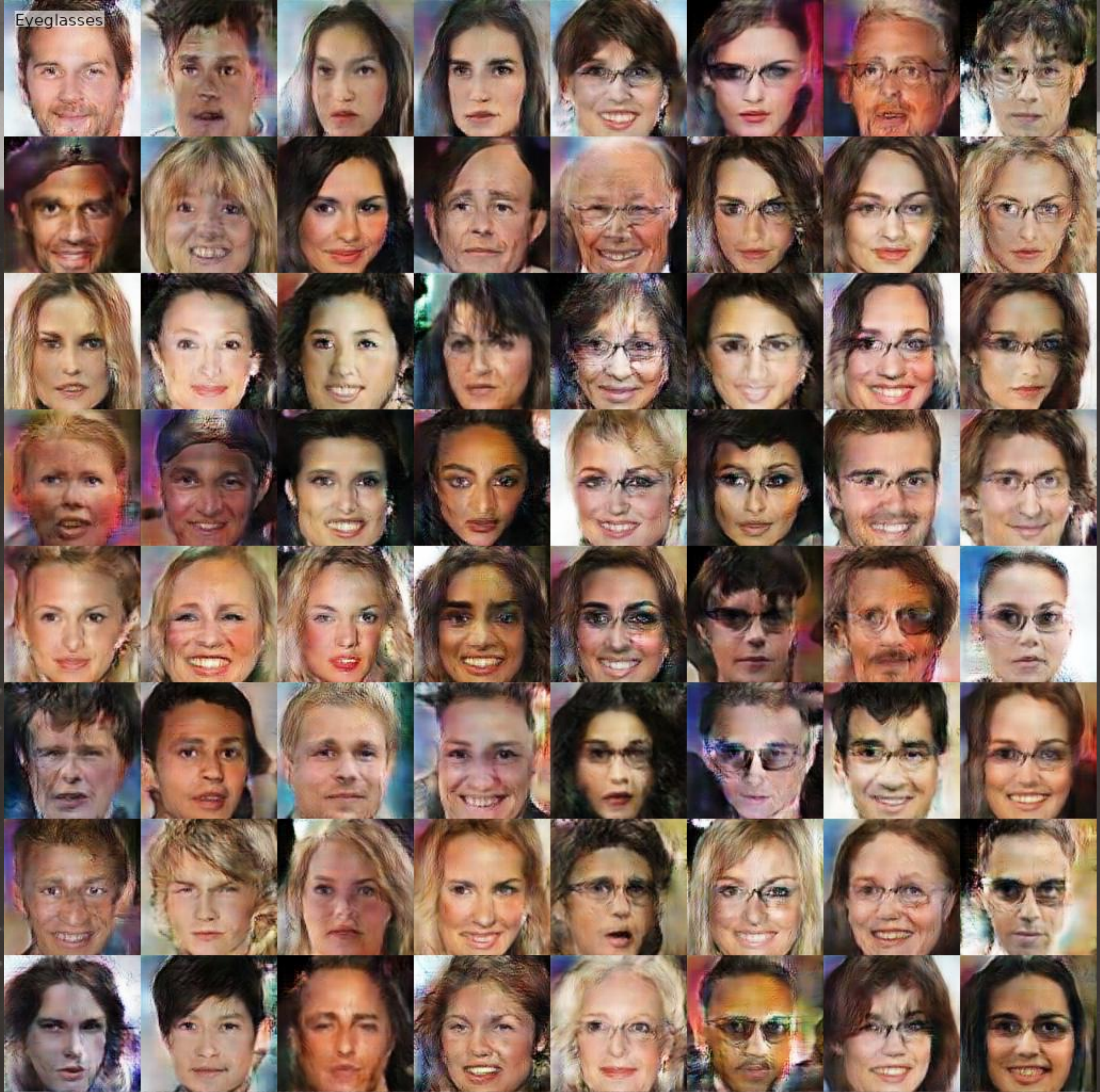
Results



Results



Results



Results