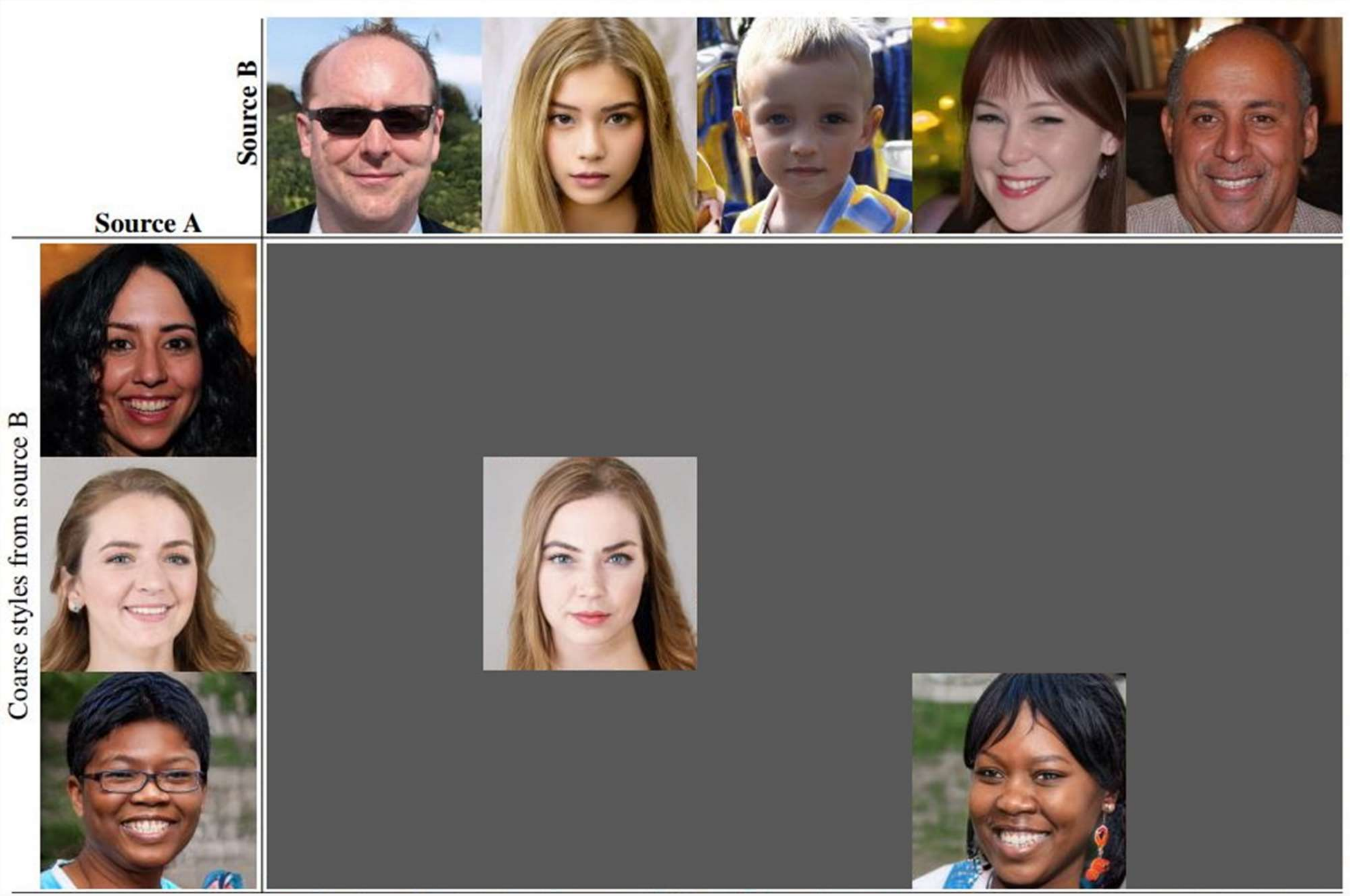
Generative Adversarial Networks

Daniel da Silva Costa
daniel.scosta@edu.unirio.br

A geração automática de novas imagens que sintetizem imagens reais é um importante desafio e útil em diversas áreas.

Alguns exemplos:

- **Área Jurídica**
  - Criação de um sistema que permite às testemunhas oculares de crimes, gerarem a imagem de um suspeito sem a necessidade de um especialista (ZALTRON et al., 2020).
- **Área da Saúde**
  - Geração de novas imagens histopatológicas de câncer de mama (KARRAS et al., 2020).
- **Área do Entretenimento**
  - Criação de fases para o jogo digital Mario (VOLZ et al. 2018) e a criação automática de personagens de *anime* (JIN et al., 2017).



Fonte: Extraído de (KARRAS et al., 2019)

Generative Adversarial Networks (GAN)

- **Paper**
  - **GOODFELLOW, Ian** et al. **Generative Adversarial Nets**. Advances in Neural Information Processing Systems, v. 27, **2014**.
- GAN é uma abordagem recente e promissora de *Deep Learning*.
- Pertence ao subconjunto de modelos generativos.
- Pode gerar amostras de categorias diferentes em paralelo.
- O projeto da rede geradora tem poucas restrições.
- GANs são subjetivamente consideradas como produtores de melhores amostras do que outros métodos.
- A GAN é uma das técnicas mais utilizadas e investigadas atualmente para gerar imagens de rostos que sintetizem um conjunto de dados.

Chapter 12

KROHN, Jon; BEYLEVELD, Grant; BASSENS, Aglaé. Deep Learning Illustrated. Addison-Wesley Professional, 2019.

Essential GAN Theory

“At its highest level, a GAN involves **two deep learning networks pitted against each other in an adversarial relationship**. (...) **one network is a generator that produces forgeries of images**, and **the other is a discriminator that attempts to distinguish the generator’s fakes from the real thing**. (...) the generator is tasked with receiving **a random noise input and turning this into a fake image** (...) The discriminator—**a binary classifier of real versus fake images** (...) (...) As training continues, the two models battle it out, trying to outdo one another, and, in so doing, both models become more and more specialized at their respective tasks. **Eventually** this adversarial interplay can culminate in **the generator producing fakes that are convincing not only to the discriminator network but also to the human eye.**”

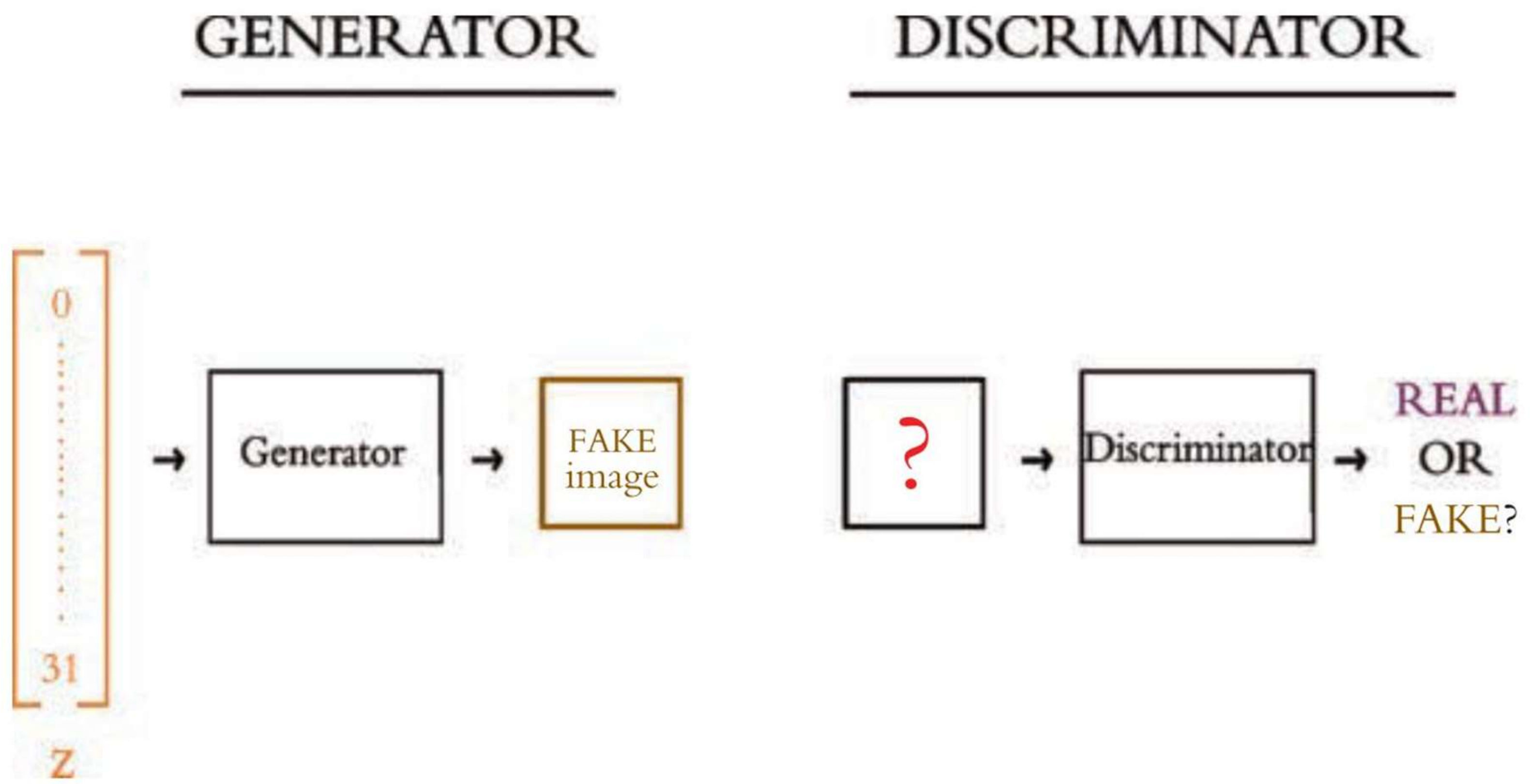


Figure 12.1 Highly simplified schematic diagrams of the two models that make up a typical GAN: the generator (left) and the discriminator (right)

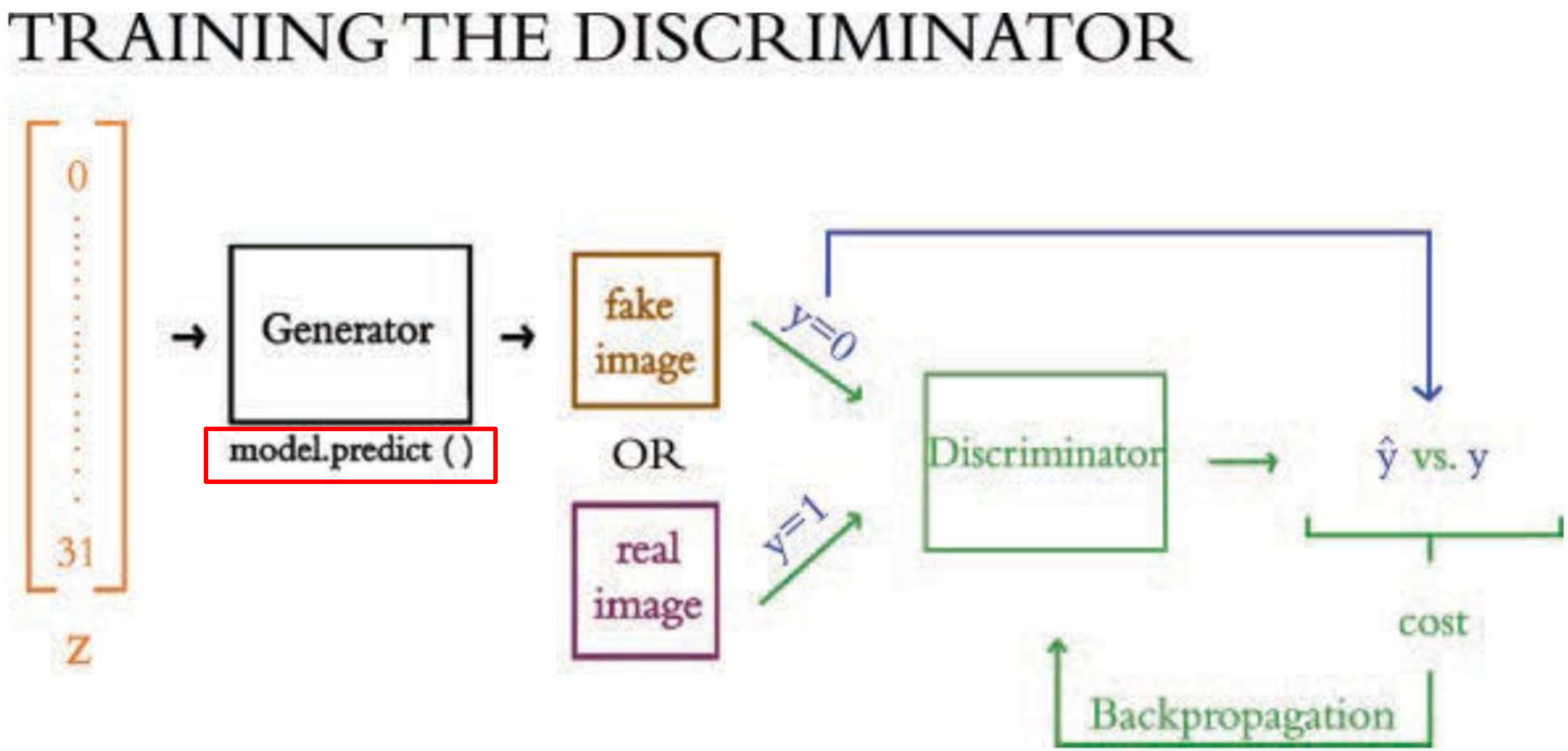


Figure 12.2 This is an outline of the discriminator training loop. Forward propagation through the generator produces **fake images. These are mixed into batches with real images from the dataset and, together with their labels, are used to train the discriminator**. Learning paths are shown in green, while non-learning paths are shown in black and the blue arrow calls attention to the image labels, *y*.

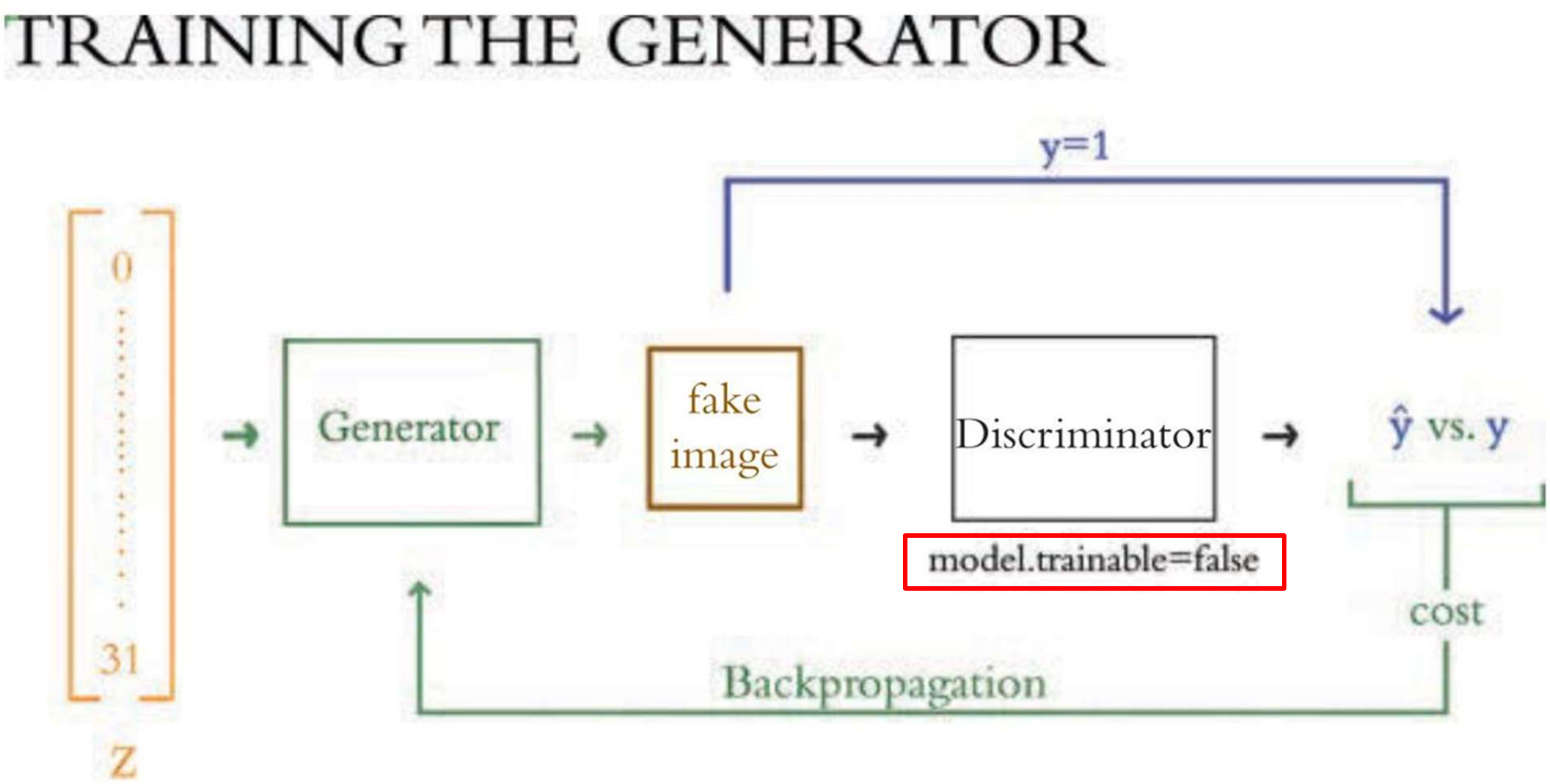


Figure 12.3 An outline of the generator training loop. Forward propagation through the generator produces fake images, and inference with the discriminator scores these images. The generator is improved through backpropagation. As in Figure 12.2, learning paths are shown in green, and non-learning paths are shown in black. The blue arrow calls attention to the relationship between the image and its label *y* which, **in the case of generator training, is always equal to 1**.

“Thus, in each of these two processes, one of the models creates its output (either a fake image or a prediction of whether the image is fake) **but is not trained, and the other model uses that output to learn to perform its task better**.”

“**The generator receives a random noise vector *z* as input** and produces a fake image as an output.”

“**Crucially to this process**, we lie to the discriminator and **label all of these fake images as real (y = 1)**.”

Relação *Minimax* entre uma Rede Geradora *G* e uma Discriminadora *D* (GOODFELLOW et al., 2014)

$$\min_{G} \max_{D} V(D, G) = \mathrm{E}_{\mathrm{X} \sim \mathrm{PData}}[\log \mathrm{D}(\mathrm{x})] + \mathrm{E}_{\mathrm{Z} \sim \mathrm{P_z(z)}}[\log (1 - \mathrm{D}(\mathrm{G}(\mathrm{z})))]$$

onde:

- $D(x)$ representa a probabilidade de que x pertence ao conjunto de dados reais;
- $\mathrm{E}_{\mathrm{X} \sim \mathrm{PData}}[log\ D(x)]$ é a expectativa de $log\ D(x)$ com relação a x ter sido sorteado a partir do conjunto de dados reais;
- $D(G(z))$ representa a probabilidade de $G(z)$ pertencer ao conjunto dos dados reais. Logo, $1 - D(G(z))$ representa a probabilidade de $G(z)$ não pertencer ao conjunto dos dados reais;
- A expectativa de $log\ (1 - D(G(z)))$ com relação a $G(z)$ ter sido produzido por G, a partir de um ruído aleatório z, é representada por $\mathrm{E}_{\mathrm{Z} \sim \mathrm{P_z(z)}}\ [log\ (1 - D(G(z)))]$.

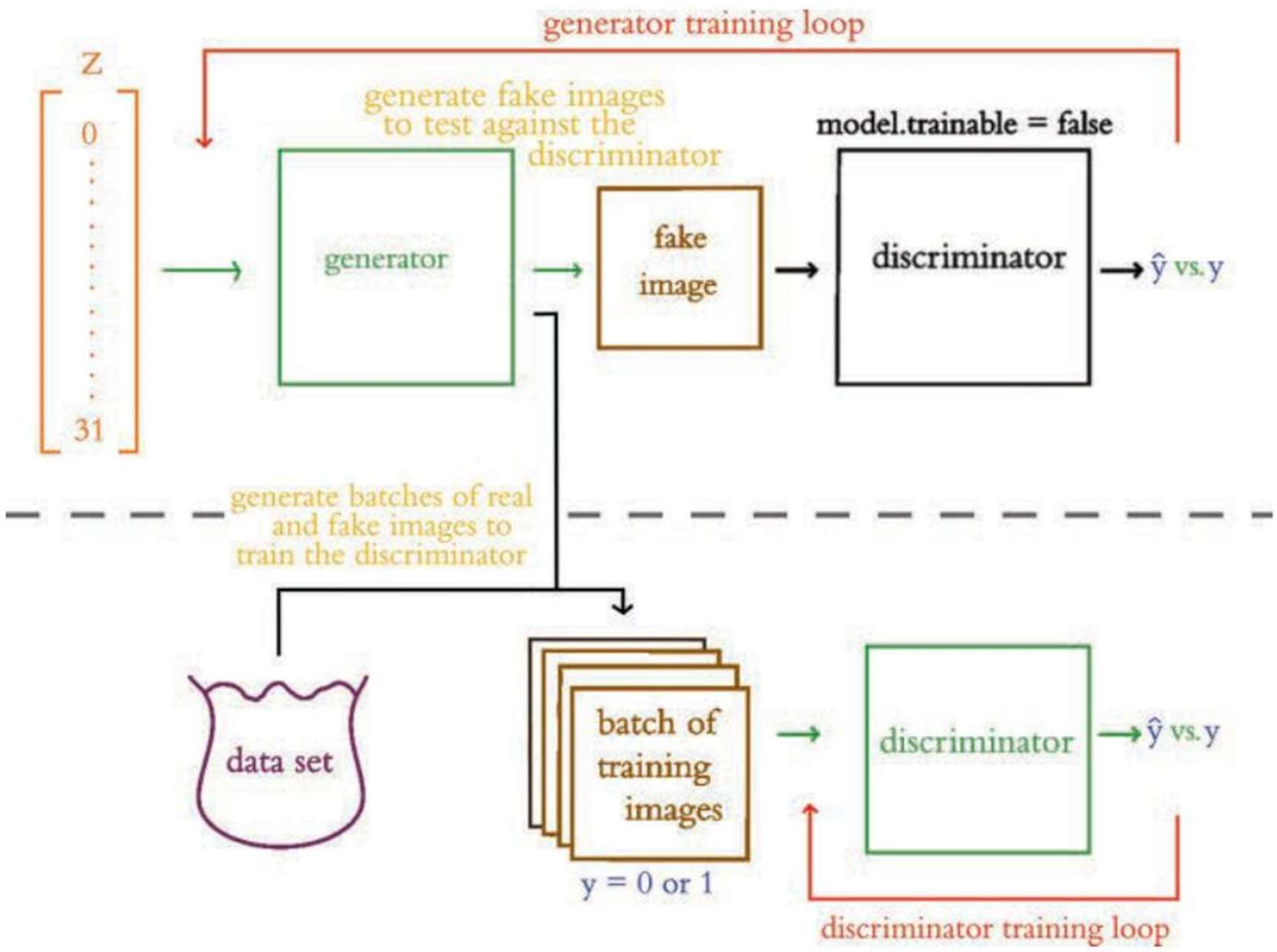
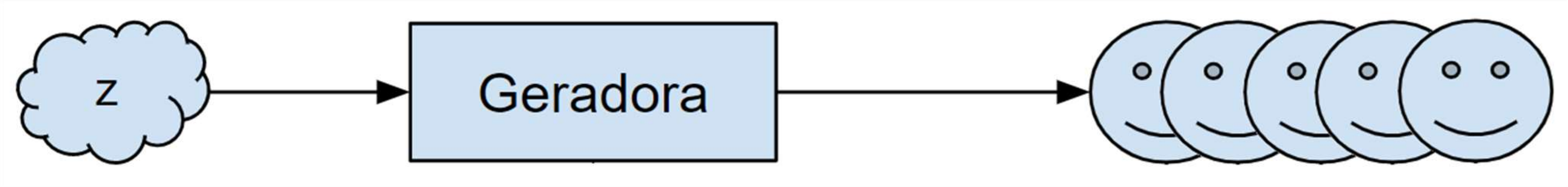


Figure 12.9 Shown here is a summary of the whole adversarial network. The horizontal dashes visually separate generator training from discriminator training. Green lines indicate trainable paths, whereas inference-only paths are in black. The red arrows above and below indicate the path of the backpropagation step during the respective training processes.

“the discriminator initially has no trouble at all learning to distinguish real from fake. As the generator trains, however, it gradually learns how to replicate some of the structure of the real images. **Eventually, the generator becomes crafty enough to fool the discriminator, and thus in turn the discriminator learns more-complex and nuanced features from the real images such that outwitting the discriminator becomes trickier**. (...) **At some point, the two adversarial models arrive at a stalemate: They reach the limits of their architectures, and learning stalls on both sides**. At the conclusion of training, **the discriminator is discarded** and **the generator is our final product**. **We can feed in random noise, and it will output images. (...) In this sense, the generative capacity of GANs could be considered creative**.”

GAN - Produção de Novas Imagens com a Rede Geradora Treinada



(Extraído da dissertação de mestrado do Daniel)

The Quick, Draw! Dataset

From *https://github.com/googlecreativelab/quickdraw-dataset*

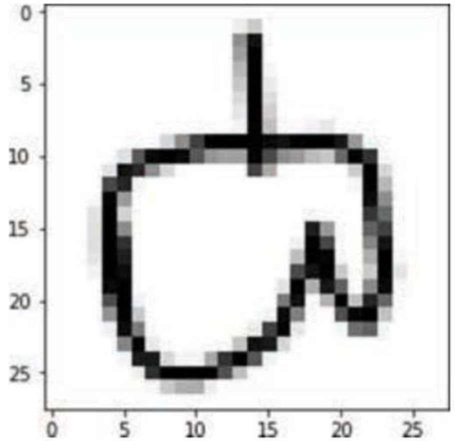


Figure 12.6 This example bitmap is the 4,243rd sketch from the apple category of the Quick, Draw! dataset.

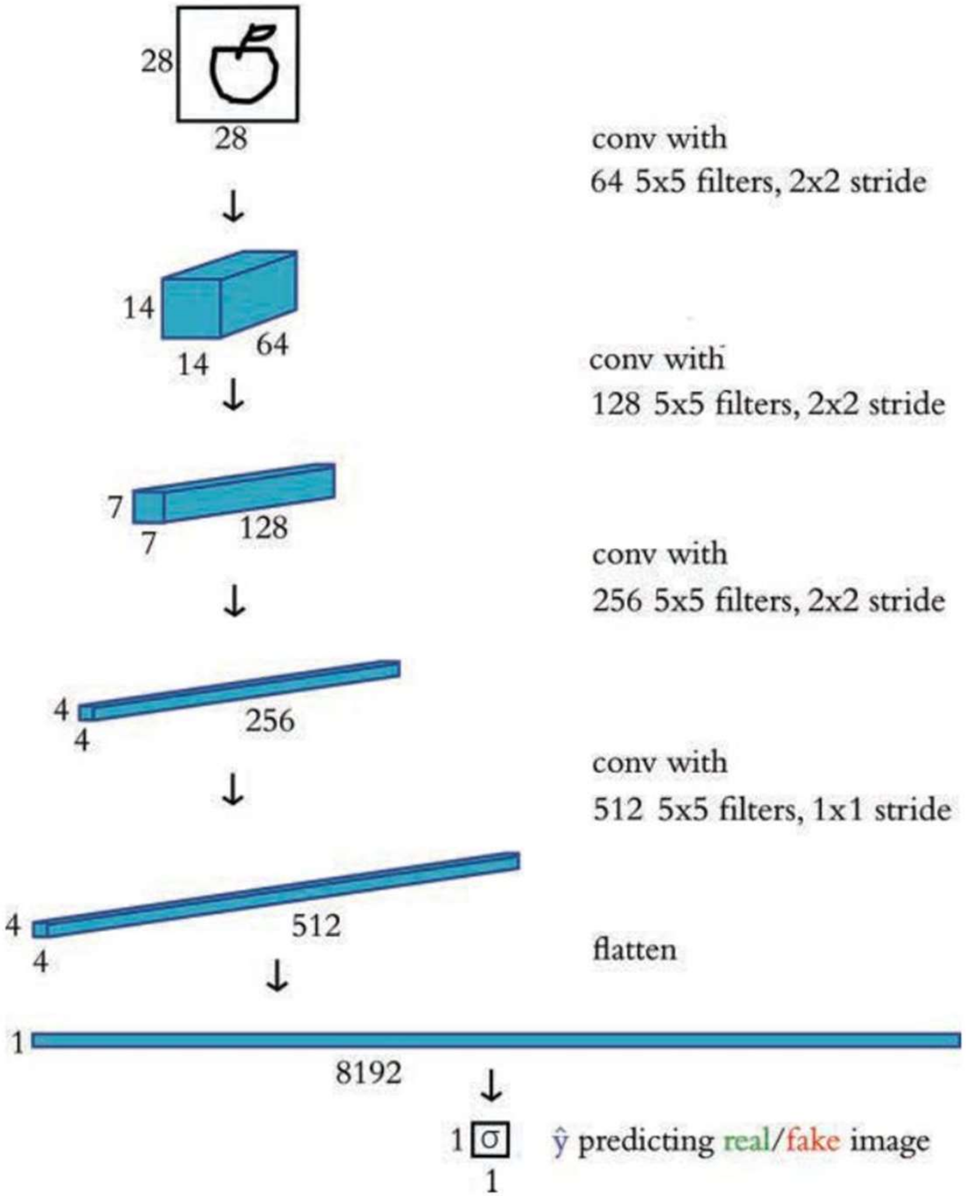
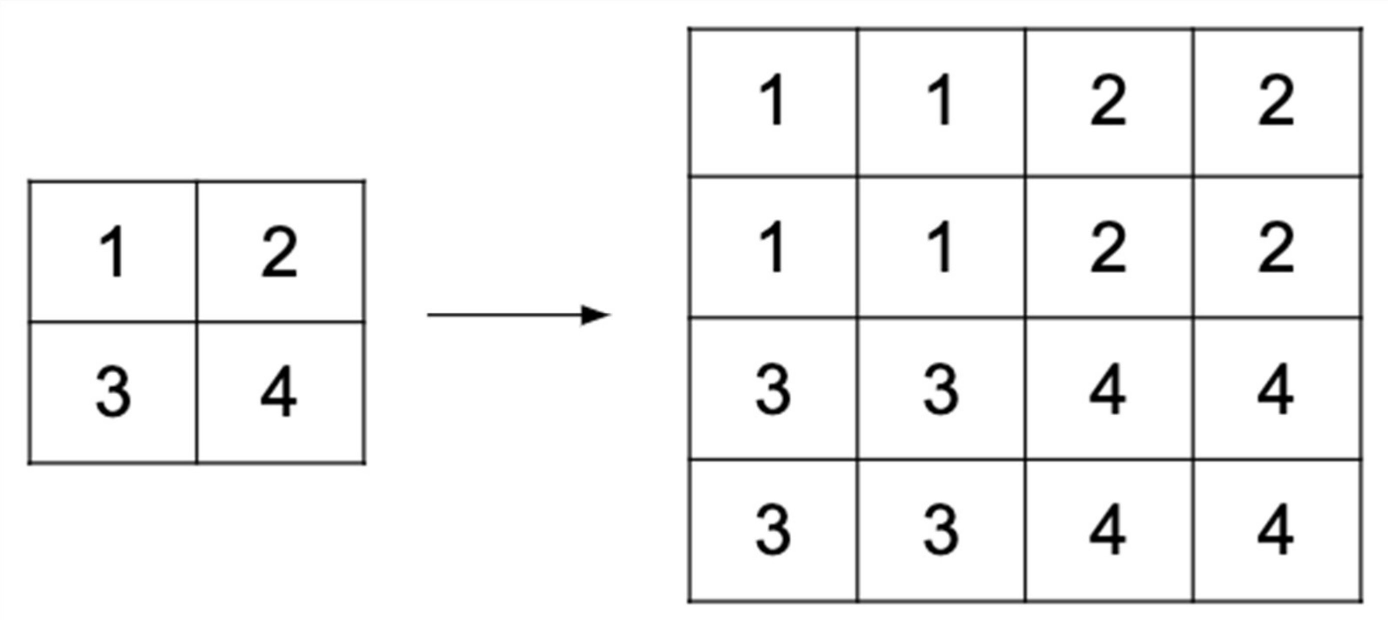


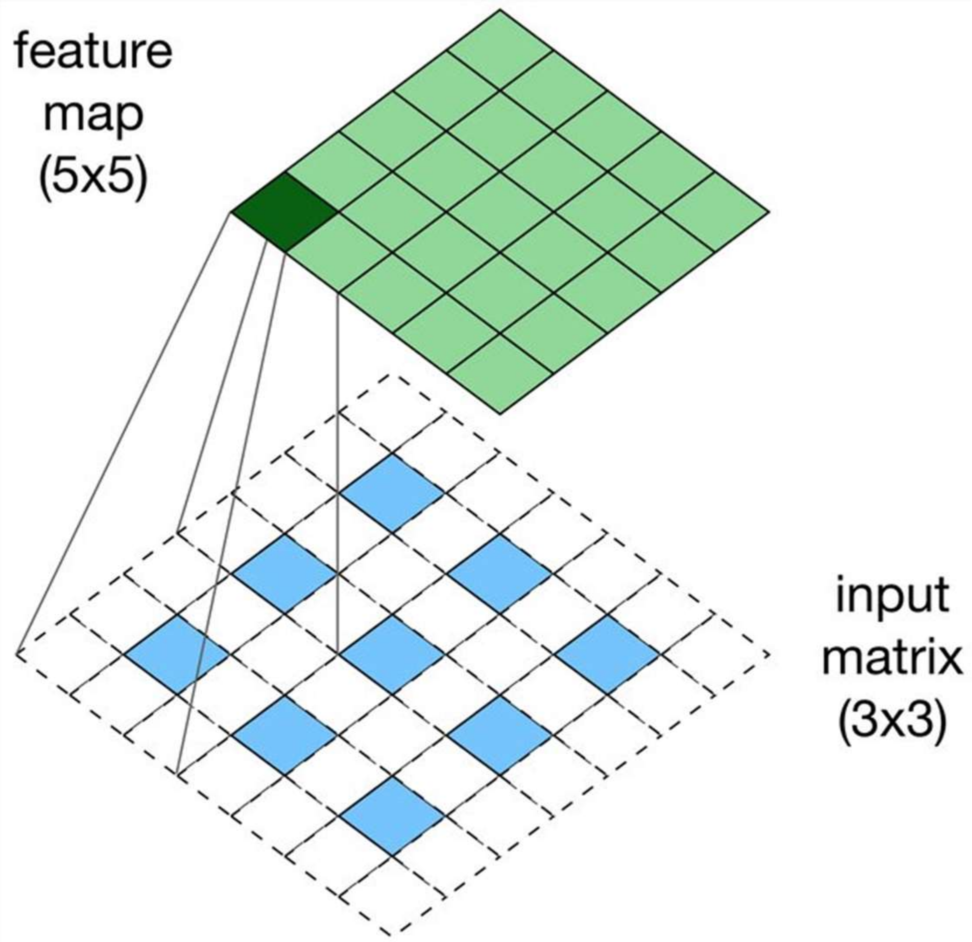
Figure 12.7 A schematic representation of our **discriminator** network for predicting whether an input image is real (in this case, a hand-drawn apple from the Quick, Draw! dataset) or fake (produced by an image generator)

Upsampling Layer



“Simple layer used to increase the size of its input by repeating its entries. **Does not have any parameters**.”
[Layers — ML Compiled]

Transposed Convolutional Layer



- *dilation_rate=(1, 1) # Keras*
- Layers — ML Compiled
  - "Sometimes referred to as a **deconvolutional layer**."
  - "Can be used for upsampling."
  - **"Pads the input with zeros and then applies a convolution."**
  - "**Has parameters which must be learned**, unlike the upsampling layer."
- How to use the UpSampling2D and Conv2DTranspose Layers in Keras
  - https://machinelearningmastery.com/upsampling-and-transpose-convolution-layers-for-generative-adversarial-networks/

Extraído de
https://www.researchgate.net/publication/331258691_Deep_Learning_Methods_for_Drum_Transcription_and_Drum_Pattern_Generation/figures?lo=1

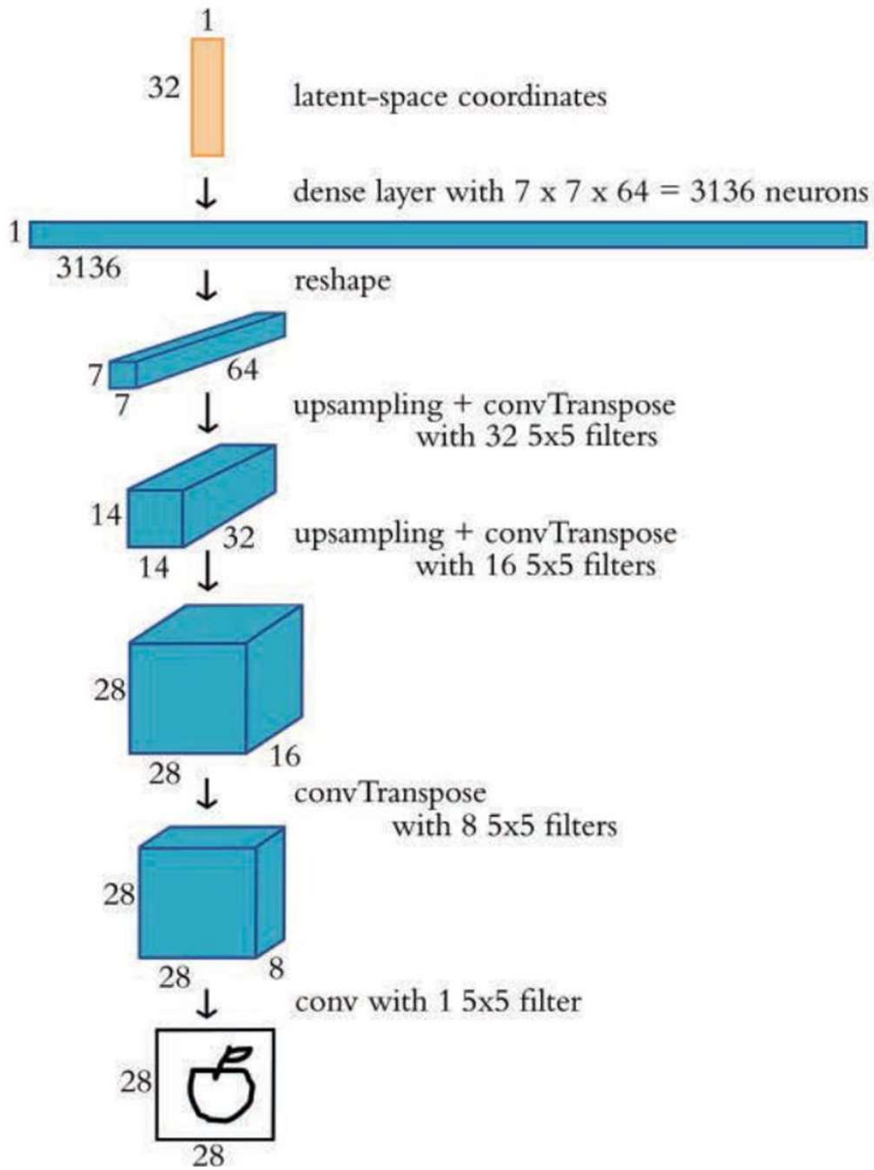


Figure 12.8 A schematic representation of our **generator** network, which takes in noise (in this case, representing 32 latent-space dimensions) and outputs a 28×28-pixel image. After training as part of an adversarial network, these images should resemble images from the training dataset (in this case, hand-drawn apples).

Notebook e Considerações sobre Resultados e Treinamento

https://colab.research.google.com/drive/1X4AGvKe7QGU-JQWrp7t7mwULcZ3b24be

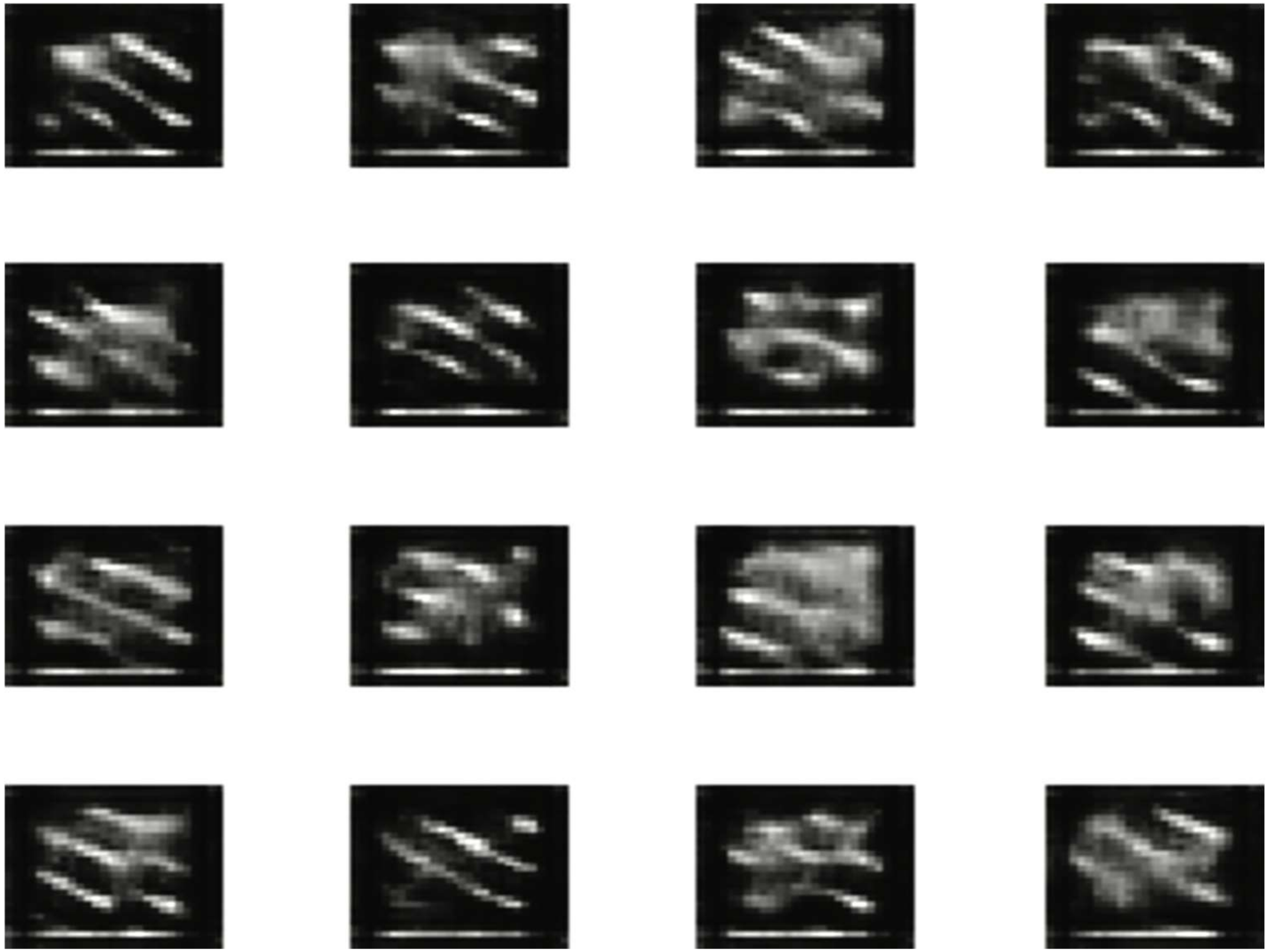


Figure 12.10 Fake apple sketches generated after **100 epochs** of training our GAN

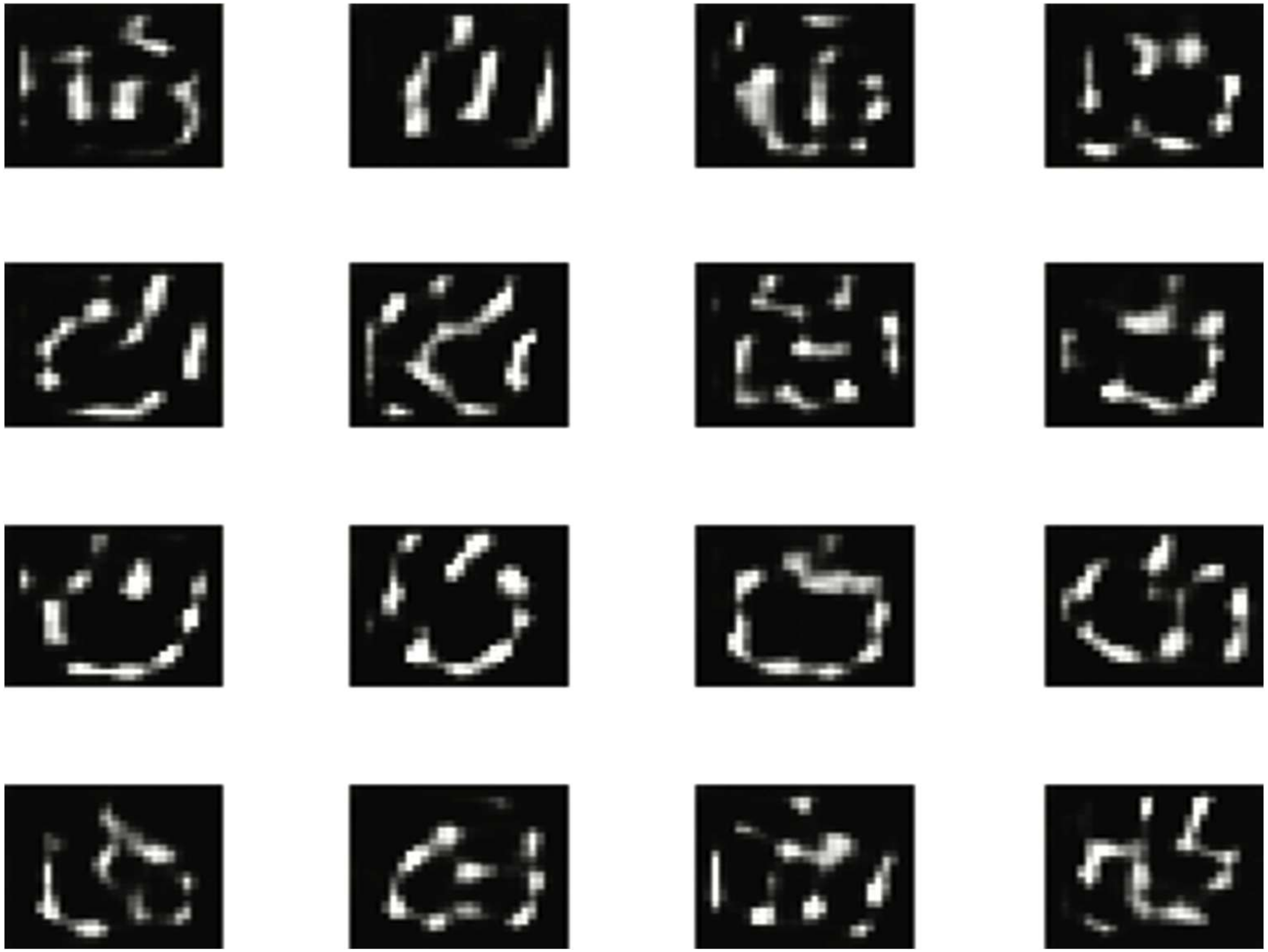


Figure 12.11 Fake apple sketches after **200 epochs** of training our GAN

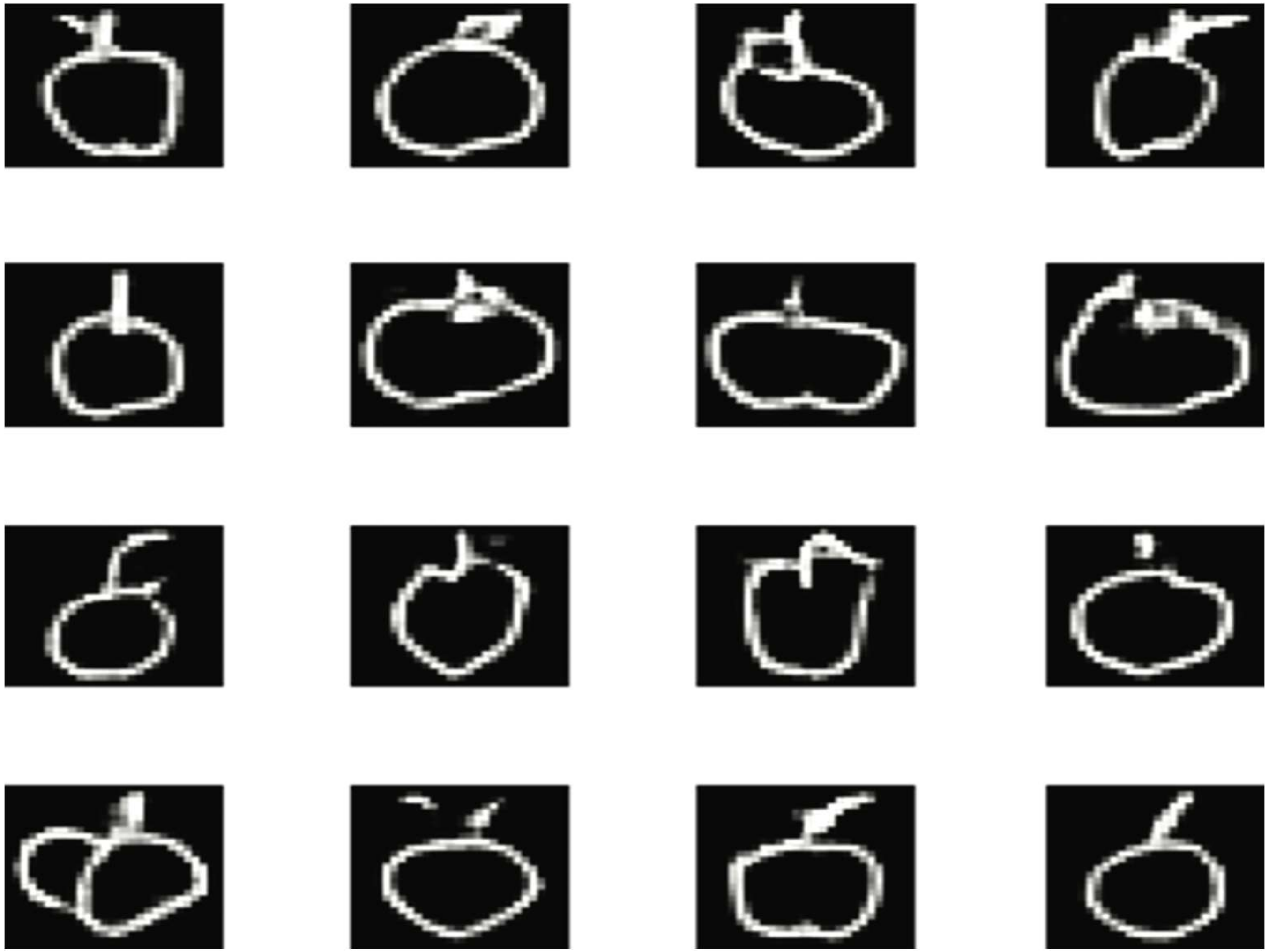


Figure 12.12 Fake apple sketches after **1,000 epochs** of training our GAN

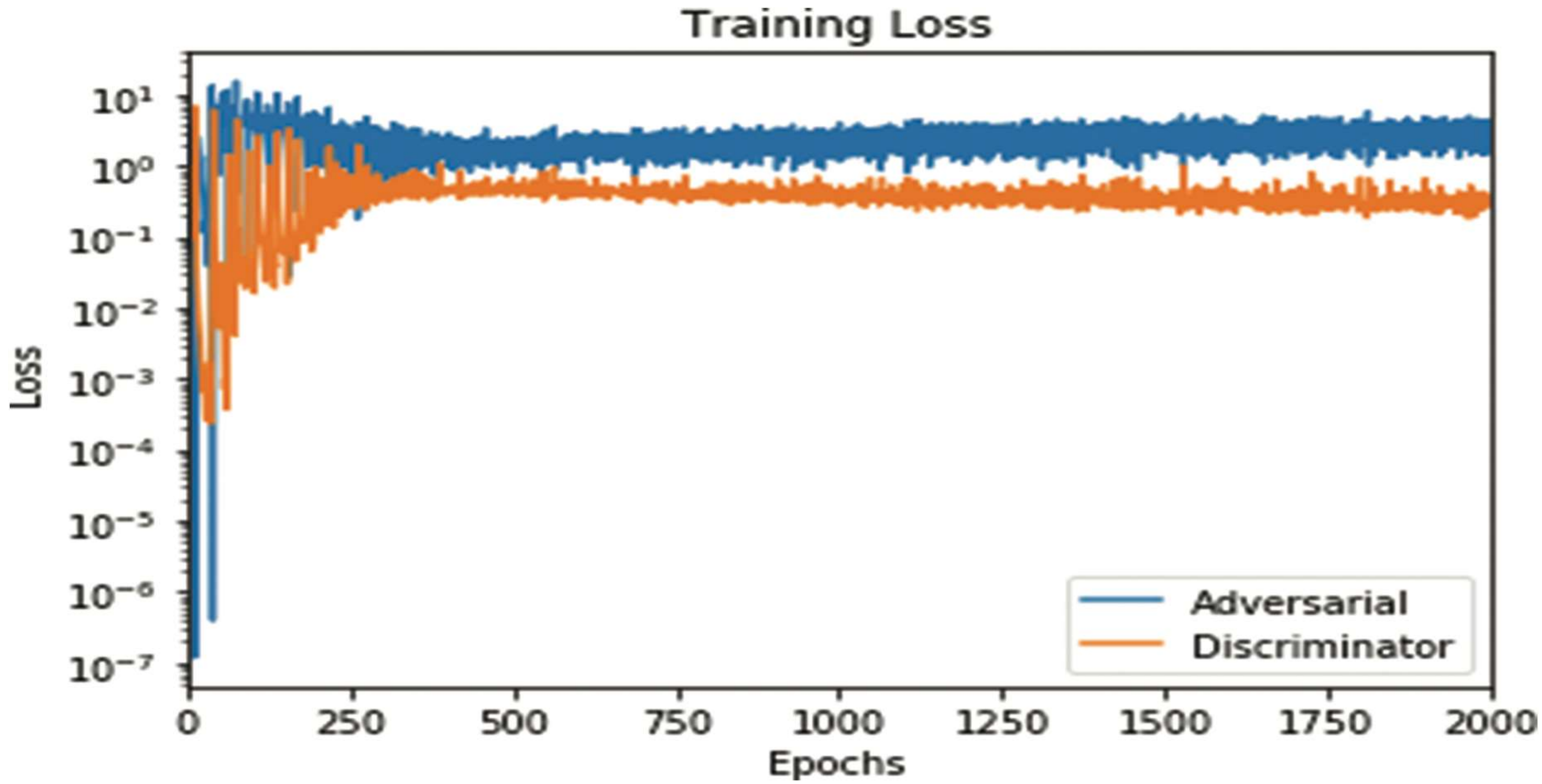


Figure 12.13 GAN training loss over epochs

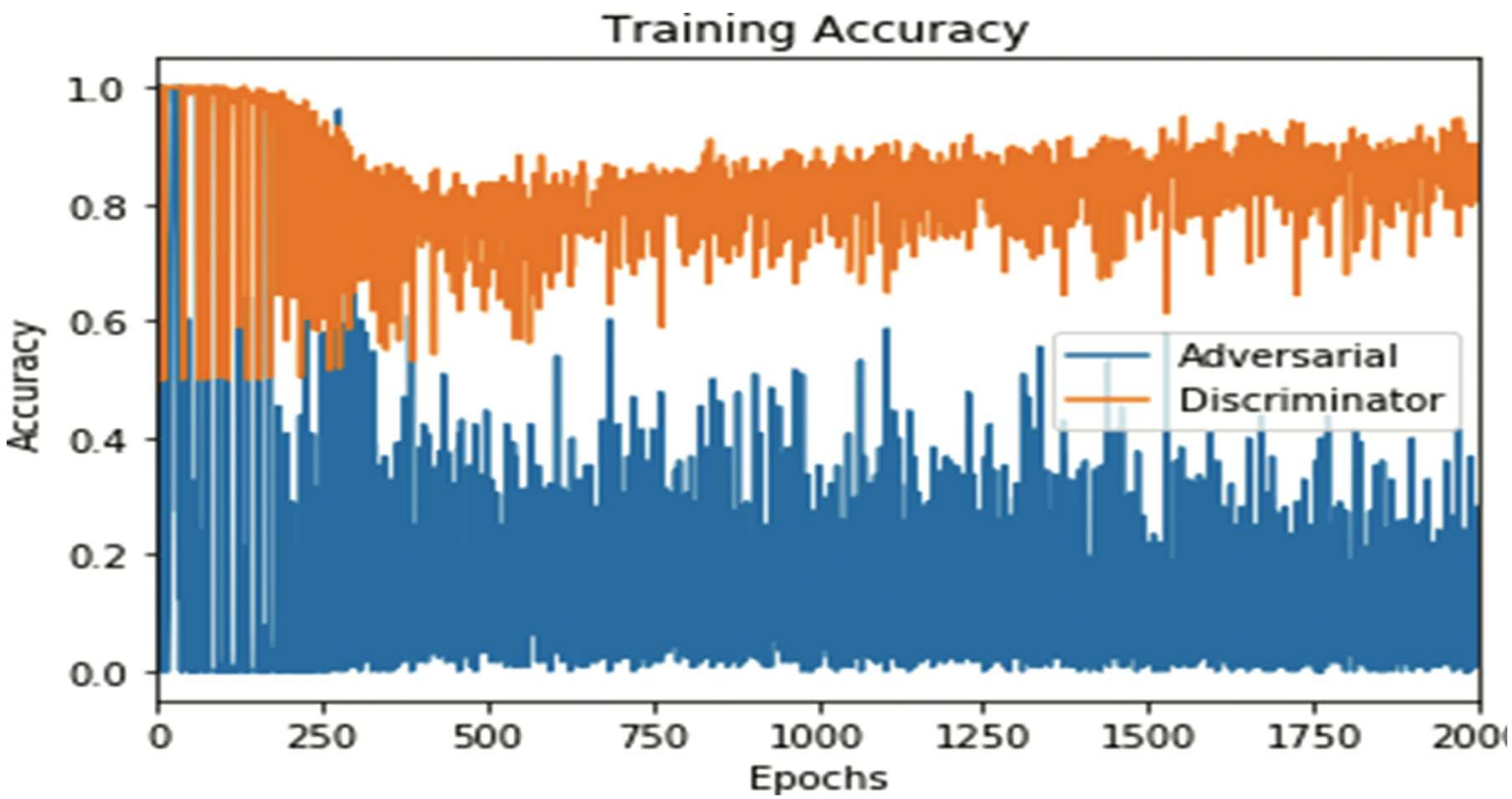
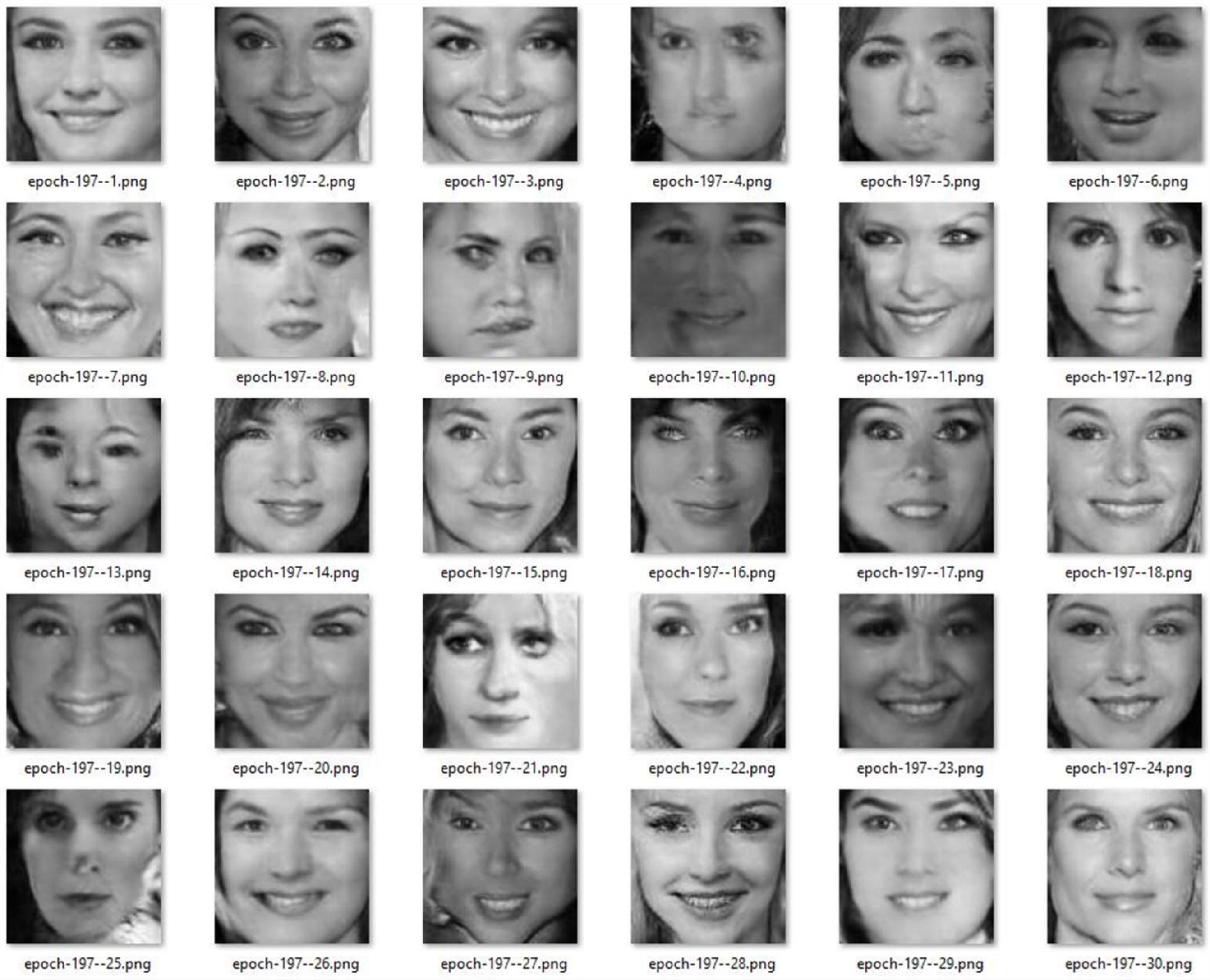


Figure 12.14 GAN training accuracy over epochs

Imagens sintéticas geradas pelo modelo do Tipo 2 (5000) treinado com imagens do CelebA filtradas pelo atributo rosto oval.

(Extraído da dissertação de mestrado do Daniel)



Dificuldades do Treinamento dos Modelos GAN

- Convergência das redes geradora e discriminadora (*Nash equilibrium*).
- Treinar GANs requer **uma grande quantidade de dados de treinamento**.
- Treinar GANs com menos dados, **pode reduzir a variabilidade dos resultados ou dificultar a convergência do modelo**.
- Atualmente, as técnicas buscam, em geral, **eliminar os resultados de baixa qualidade após o modelo ser treinado**.

Imagens geradas pelo modelo do **Tipo 1** treinado com **15000** imagens. Esse modelo parece ter entrado em ***Mode Collapse*. Observa-se que os olhos e a boca das imagens destacadas parecem os mesmos.**

(Extraído da dissertação de mestrado do Daniel)

Heurísticas que Podem Ajudar a Estabilizar o Treinamento dos Modelos

- Referências: GOODFELLOW, 2016; SALIMANS et al., 2016; CHINTALA et al., 2020.
- **Normalização dos *pixels*** das imagens de treinamento **para valores entre -1 e 1**.
- Utilização de **camadas de ativação LeakyReLU na rede geradora e na discriminadora** (A DCGAN original propõe usar a ReLU na geradora).
- Utilização de ***label smoothing*** dos rótulos das imagens reais e das imagens sintéticas **antes de treinar a discriminadora**. (Por exemplo: 1 -> 0.9)
- Utilização de ***noisy labels*** para as imagens reais e para as sintéticas **antes do treinamento da discriminadora**. (Por exemplo: 1 -> 0)
- Utilização de **otimizador Adam na discriminadora e na DCGAN propriamente**, com ***learning rate* igual a 0,0002** e ***decay rate* exponencial para as estimativas do primeiro momento igual a 0,5**.
- A ***loss function*** utilizada na discriminadora e na DCGAN foi a ***Binary Cross Entropy***.

Root Mean Square Propagation (RMSProp)

- "RMSProp, root mean square propagation, is an optimization algorithm/method designed for Artificial Neural Network (ANN) training. **And it is an unpublished algorithm first proposed in the Coursera course. "Neural Network for Machine Learning" lecture six by Geoff Hinton**.[9] RMSProp lies in the realm of adaptive learning rate methods, which have been growing in popularity in recent years because it is the extension of Stochastic Gradient Descent (SGD) algorithm, momentum method, **and the foundation of Adam algorithm**." (https://optimization.cbe.cornell.edu/index.php?title=RMSProp)

- **Visualizing Optimization Algos**
    - https://imgur.com/a/Hqolp#NKsFHJb