Computer Graphics: Image inpainting

The problem: image inpainting

- Let us say that you have a treasured old photo of your grandparents, but it has a big scratch on it. You want to repair the photo by filling in the scratch with something that looks like the rest of the photo.
- The problem of **image inpainting** is to take an image with some missing or corrupted parts and fill in those parts with something that looks like the rest of the image.

Image inpainting

- **Inputs**:
  - An image, possibly with some missing or corrupted parts
  - A mask that indicates which parts of the image are missing or corrupted
- **Outputs**:
  - The inpainted image with the missing or corrupted parts filled in
- **Performance measure**:
  - User satisfaction
  - Similarity to original
  - If the original is available, this is relatively easy to measure.

Let us hack together a simple inpainting algorithm

- We have two kind of pixels:
  - Known pixels (the ones that are not masked)
  - Unknown pixels (the ones that are masked)
- We can use the known pixels to fill in the unknown pixels.
- A simple algorithm is to use the average color of the known pixels to fill in the unknown pixels.
- This is a very simple algorithm and it will not produce good results, but it is a good starting point. It should work reasonably well for removing scratches and so on.

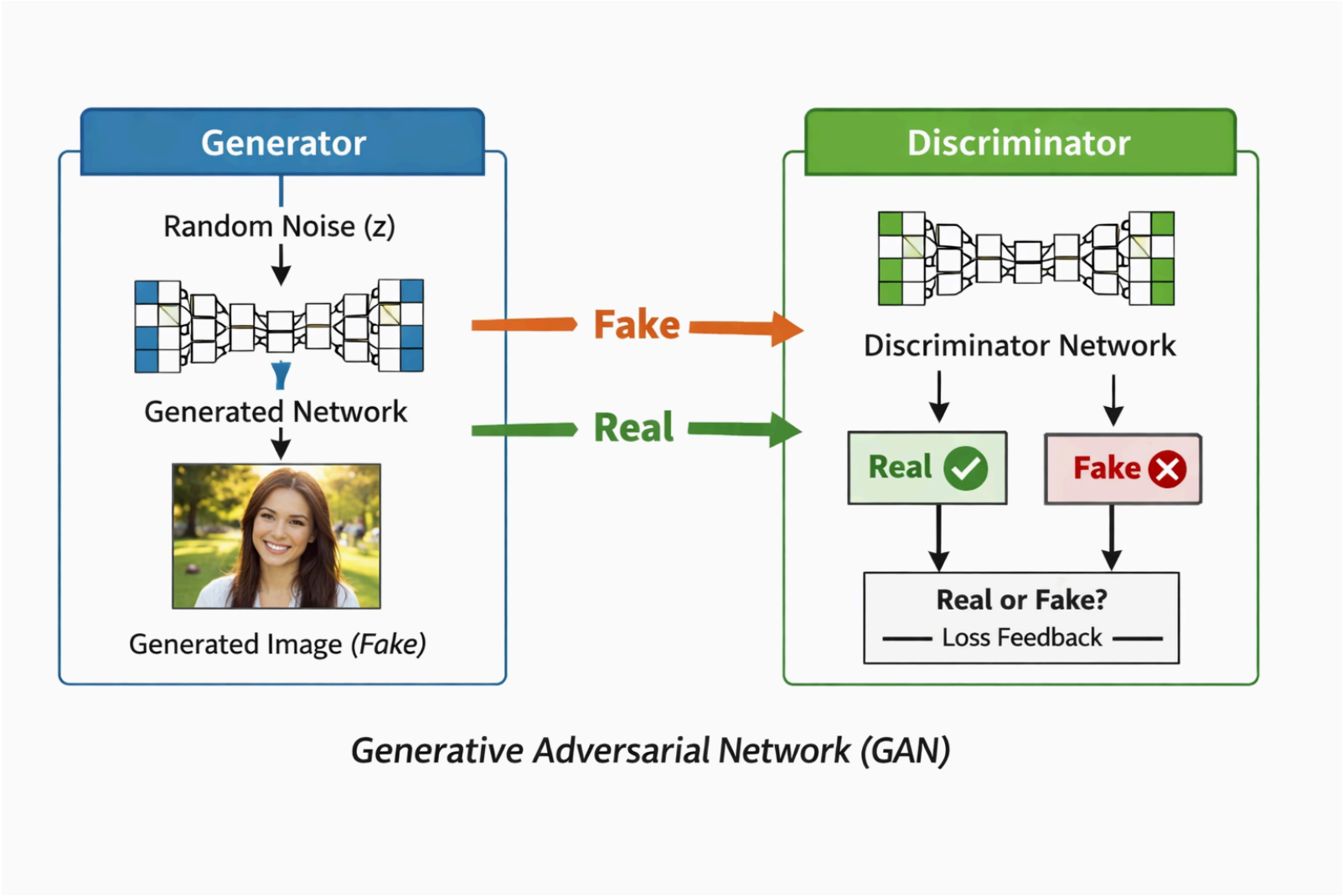
A better algorithm: use what we learned in this class to fill in the unknown pixels

- We know how to do classification and regression.
- Finding the color of the unknown pixels is a regression problem.
- We can use the neighboring known pixels as input features and the color of the unknown pixel as the target variable.
- Eg. 9 or 25 input features (the colors of the neighboring known pixels) and 3 output variables (the RGB values of the unknown pixel).
- The same code we used for predicting the price of a house can be used to predict the color of the unknown pixel!

Modern approaches: Generative Adversarial Networks (GANs)

- GANs are a type of neural network that can generate new data that is similar to the training data.
- They consist of two networks: a generator and a discriminator.
- The generator takes random noise as input and generates an image.
- The discriminator takes an image as input and tries to determine if it is real (from the training data) or fake (generated by the generator).
- The generator and discriminator are trained together in a game-like setting, where the generator tries to fool the discriminator and the discriminator tries to correctly classify the images.

Generative Adversarial Networks (GANs)

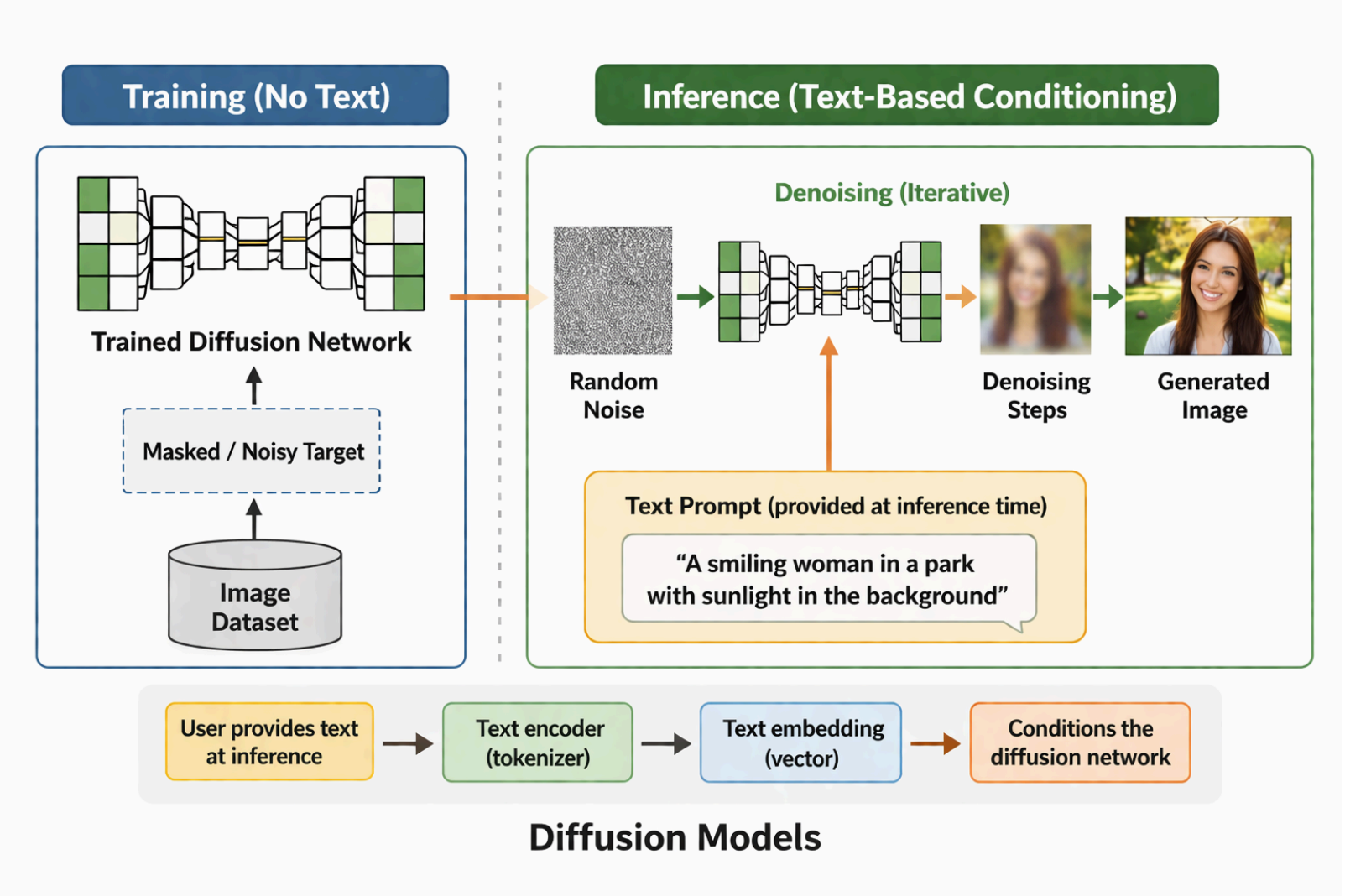


Generative Adversarial Network (GAN)

Diffusion models

- Diffusion models are a type of generative model that can generate new data by iteratively refining a noisy image.
- They consist of a forward process and a reverse process.
- The forward process adds noise to the image, while the reverse process removes noise from the image.
- The reverse process can be used for image inpainting by starting with a noisy image that has the known pixels and iteratively refining it to fill in the unknown pixels.

Diffusion models



Application: repairing old photos

- Old photos often have scratches, stains, and other damage that can be repaired using image inpainting.
- The known pixels are the undamaged parts of the photo, and the unknown pixels are the damaged parts of the photo.
- Image inpainting can be used to fill in the damaged parts of the photo with something that looks like the rest of the photo.

Application: removing distracting details from pictures

- Let's say you have a nice photo of a landscape, but there is pile of garbage in the foreground that you want to remove.
- You can black out the garbage with a mask - but now you have a big black hole in your photo.
- Image inpainting can be used to fill in the black hole with something that looks like the rest of the landscape.

Application: missing in details in data

- Imagine that you have a dataset of sattelite images of the Earth, but some of the images are missing due to cloud cover.
- You can use image inpainting to fill in the missing parts of the images with something that looks like the rest of the image.
- This can be useful for tasks such as land cover classification, where you want to classify the type of land (e.g., forest, urban, water) in the image.

Pitfalls and dangers: Censorship

- Early example of image inpainting used for censorship: the original photo of Lenin in 1897, which was altered to remove a person (Nikolai Malchenko) who was later executed by the Soviet government.

Pitfalls and dangers: Censorship

- The ability to alter images can be used for nefarious purposes, such as spreading misinformation or propaganda.
- We need to be aware that every image we see may have been altered in some way.
- It is very difficult to detect if an image has been altered:
  - We can try cryptographic techniques to sign images
  - We can try to detect inconsistencies in the image (e.g., lighting, shadows, reflections), but this is not always reliable.

Try it out: impainting with Stable Diffusion