Data Science Initiative for Africa (DSI-A) - Deep Learning
Lecture 9
Advanced Topics:
Variational Autoencoders (VAEs), Generative Adversarial Networks (GANs)

Santiago Romero-Brufau
Harvard T.H. Chan School of Public Health

“Far and away the best prize that life offers is the chance to work hard at work worth doing.”

Teddy Roosevelt

Image Generation

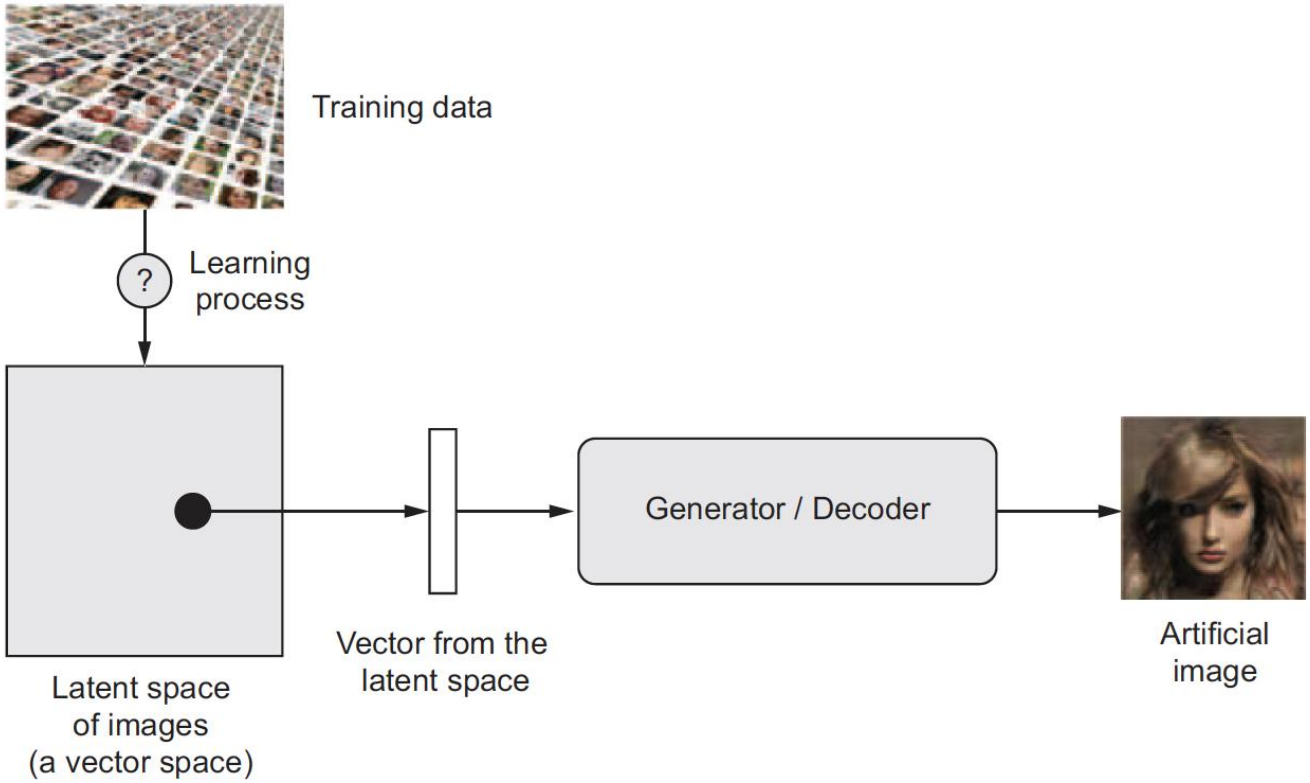
Generating Images

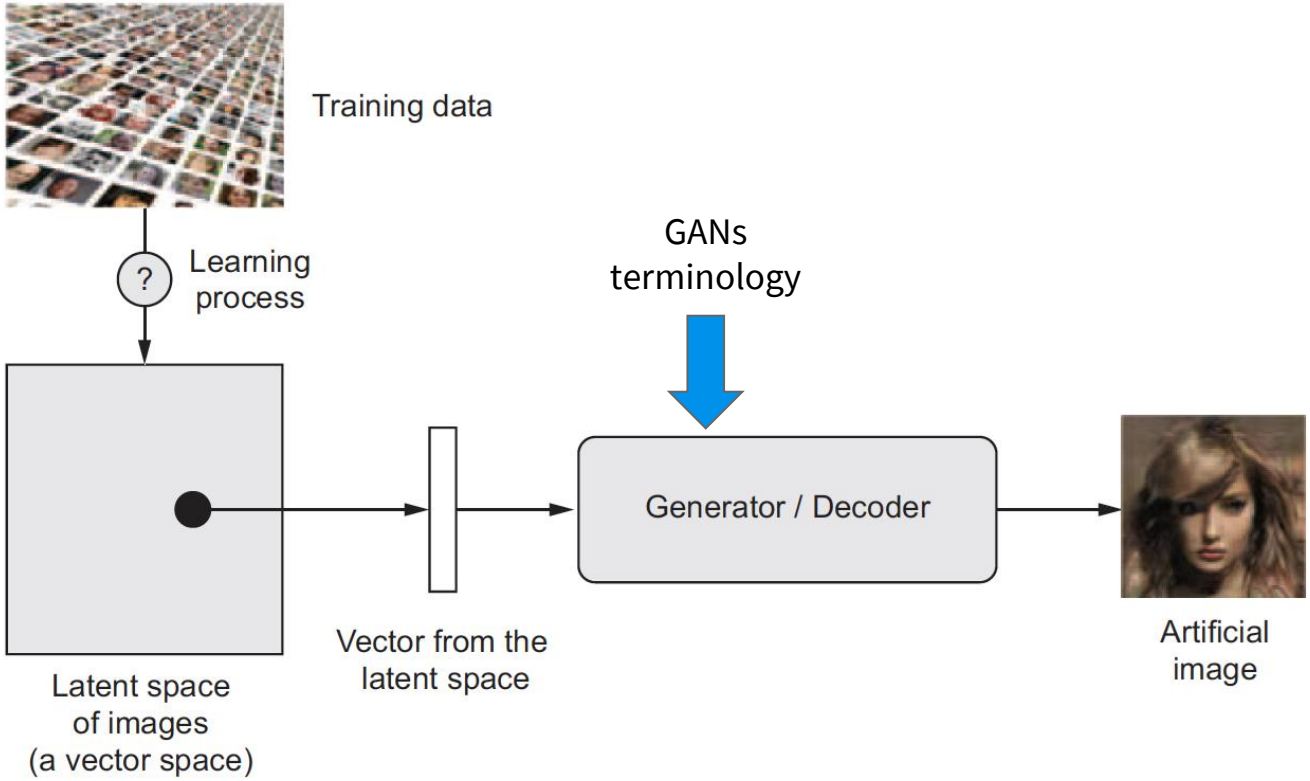
- We'll discuss two different ways of generating images
  - Variational autoencoders (VAEs)
  - Generative adversarial networks (GANs)

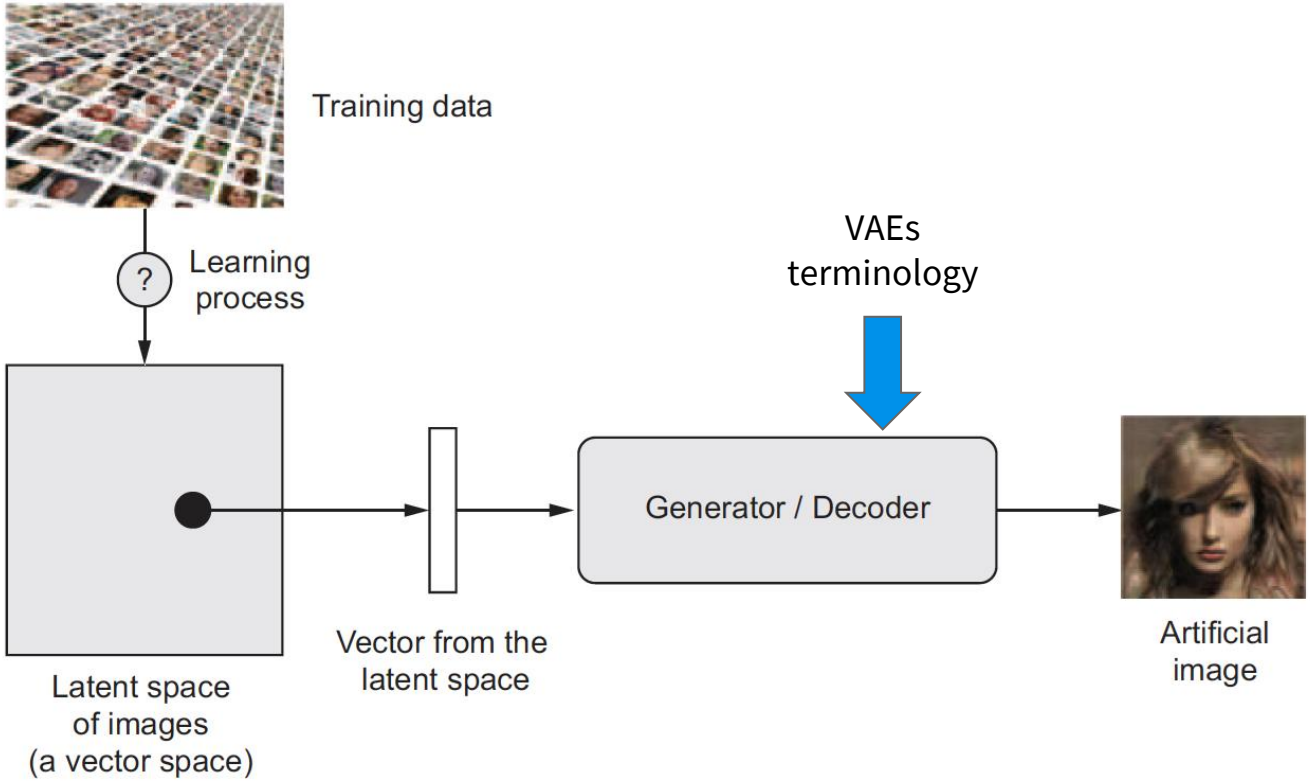
- These methods can also be used to generate sound, music or text, but we'll focus on images

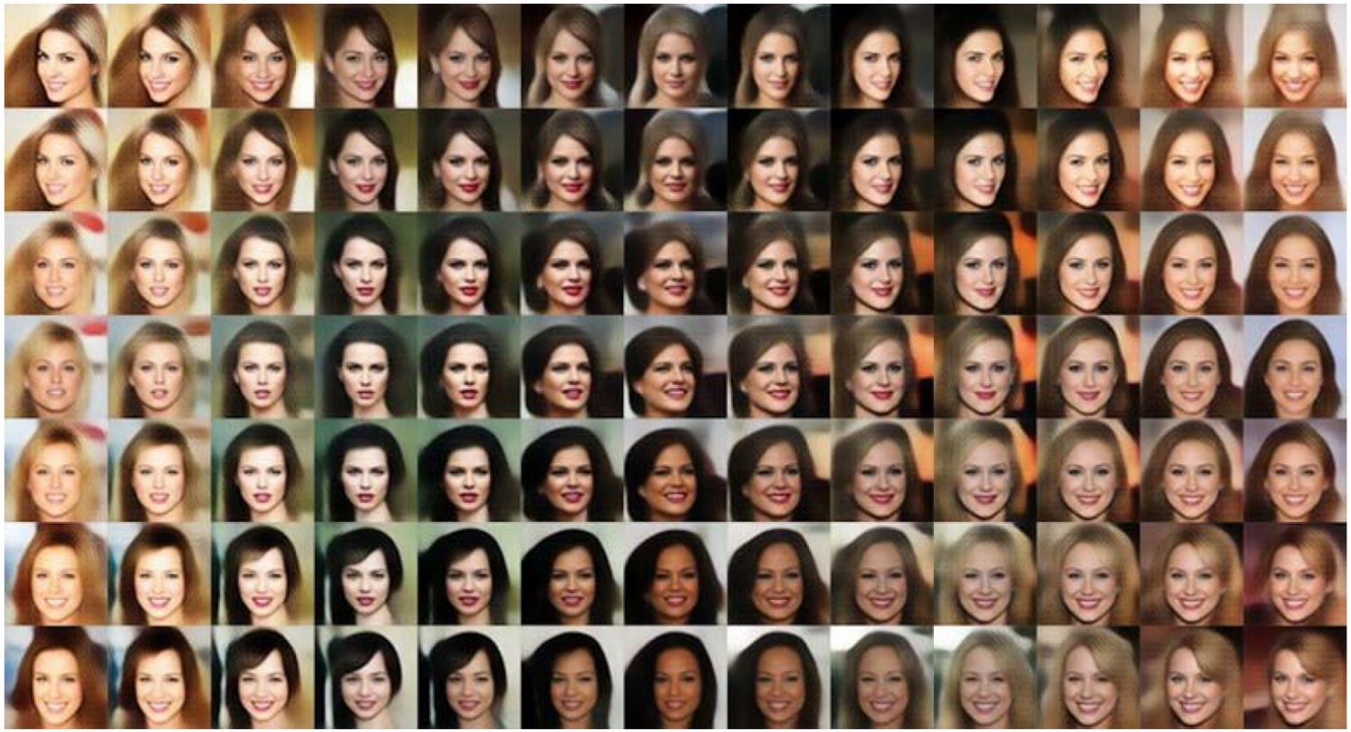
Generating Images

- **Main idea:** sample from a **latent space** of images to create entirely new images
  - Latent space: a low-dimensional representation (vector space) where any point can be mapped to a realistic-looking image
  - The module that takes in a point from the latent space and generates an image is called a **generator** (in the case of GANs) or a **decoder** (in the case of VAEs)
  - Generates images never explicitly seen before


Training data
?
Learning process
Generator / Decoder
Latent space of images (a vector space)
Vector from the latent space
Artificial image


Training data
?
Learning process
GANs terminology
Generator / Decoder
Latent space of images (a vector space)
Vector from the latent space
Artificial image


Training data
?
Learning process
VAEs terminology
Generator / Decoder
Latent space of images (a vector space)
Vector from the latent space
Artificial image



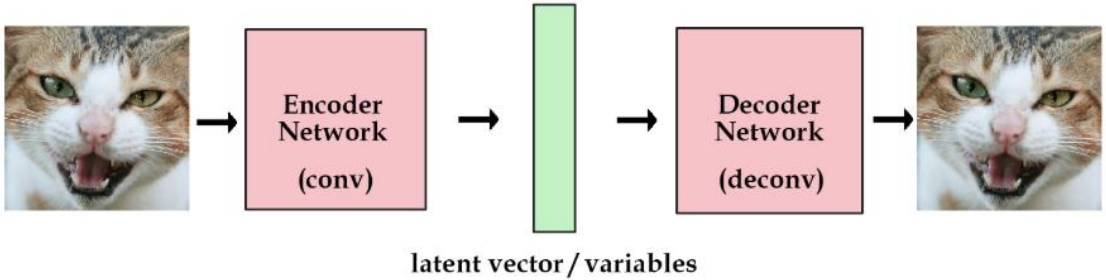
Images generated from a VAE

Concept vectors

- We want to create (learn) a latent vector space
  - We can think of this as embedding - just like what we will do with text and RNNs
  - Certain directions in the space may encode interesting axes of variation in the original data
  - Example: vectors that represent a smile or vectors that represent sunglasses
- Once we create these vectors we can edit images by projecting them into the latent space, moving their representation in a meaningful way, and then decoding them back to image space

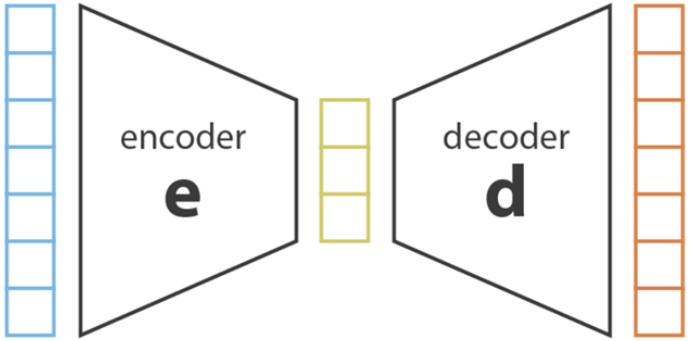
Autoencoders

- Autoencoders are neural networks used to learn efficient representations (encodings) in an **unsupervised** way
  - A type of dimensionality reduction
- At the same time they learn to decode the representations into something very close to the input
  - Reconstructs the original image as much as possible
- Essentially “memorizing” images
  - Not creating (generating) anything new



Autoencoders

- Used for
  - Denoising, face recognition, anomaly detection, dimensionality reduction, etc.

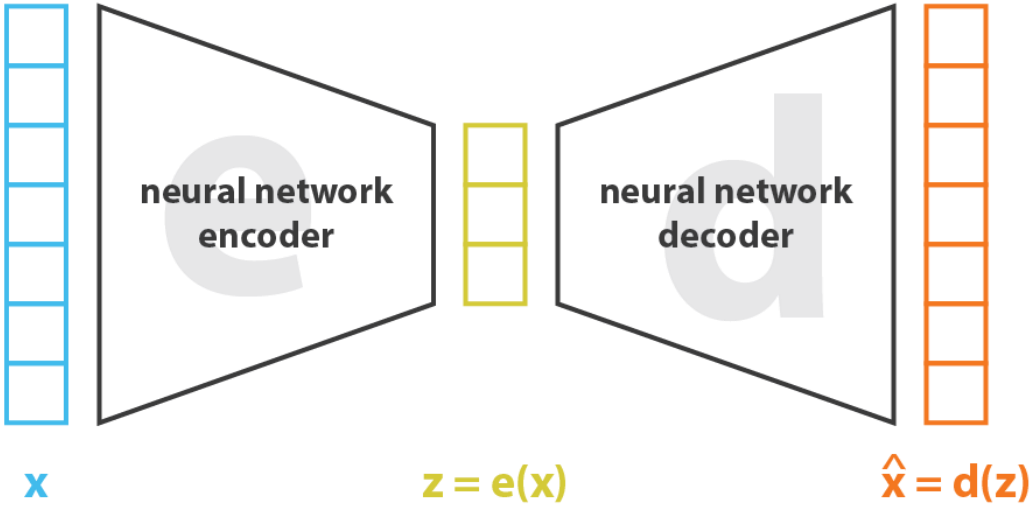


$x = d(e(x))$ ➡ **lossless encoding**
no information is lost when reducing the number of dimensions

$x \neq d(e(x))$ ➡ **lossy encoding**
some information is lost when reducing the number of dimensions and can't be recovered later

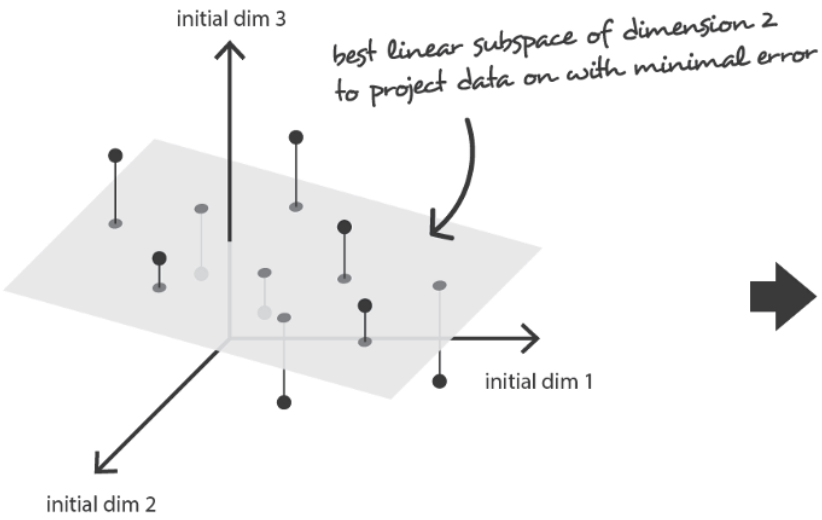
https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Autoencoders



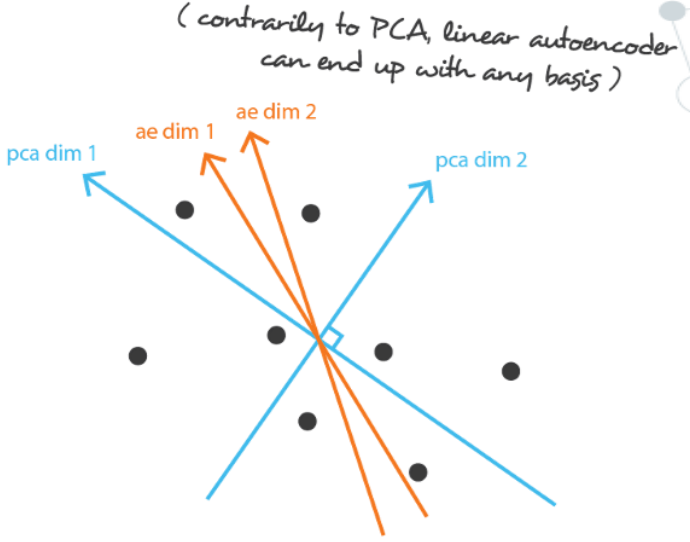
$$\text{loss} = \| x - \hat{x} \|^2 = \| x - d(z) \|^2 = \| x - d(e(x)) \|^2$$

Autoencoders



Data in the full initial space

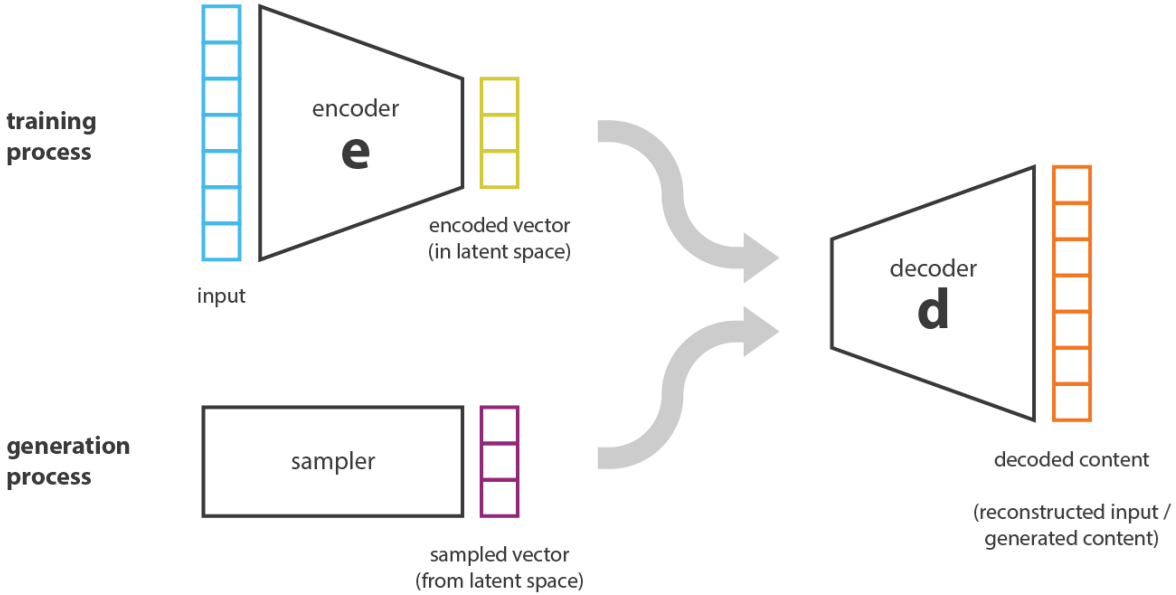
In order to reduce dimensionality, PCA and linear autoencoder target, in theory, the same optimal subspace to project data on...



Data projected on the best linear subspace

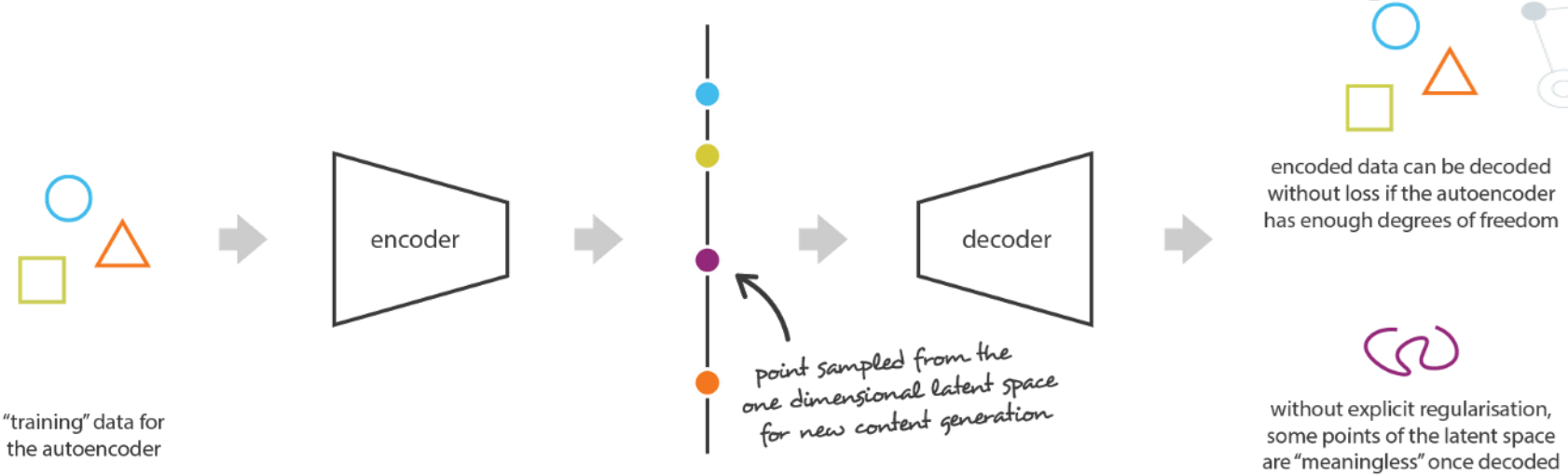
... but not necessarily with the same basis due to different constraints (in PCA the first component is the one that explains the maximum of variance and components are orthogonal)

Autoencoders



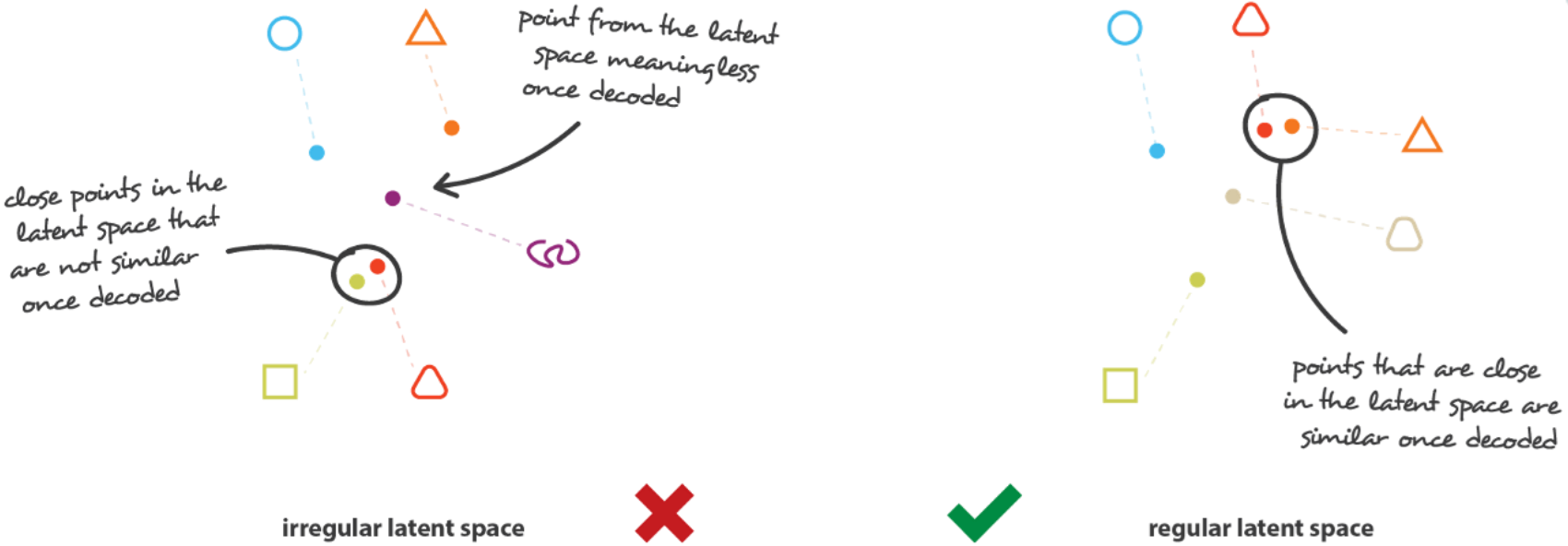
- We can use the decoder portion of the VAE to generate new content (e.g. new images!) by sampling from the latent space

Autoencoders can build irregular latent space



But just an auto-encoder doesn't guarantee that the latent space will be "regular" (that close points will map to close outputs)

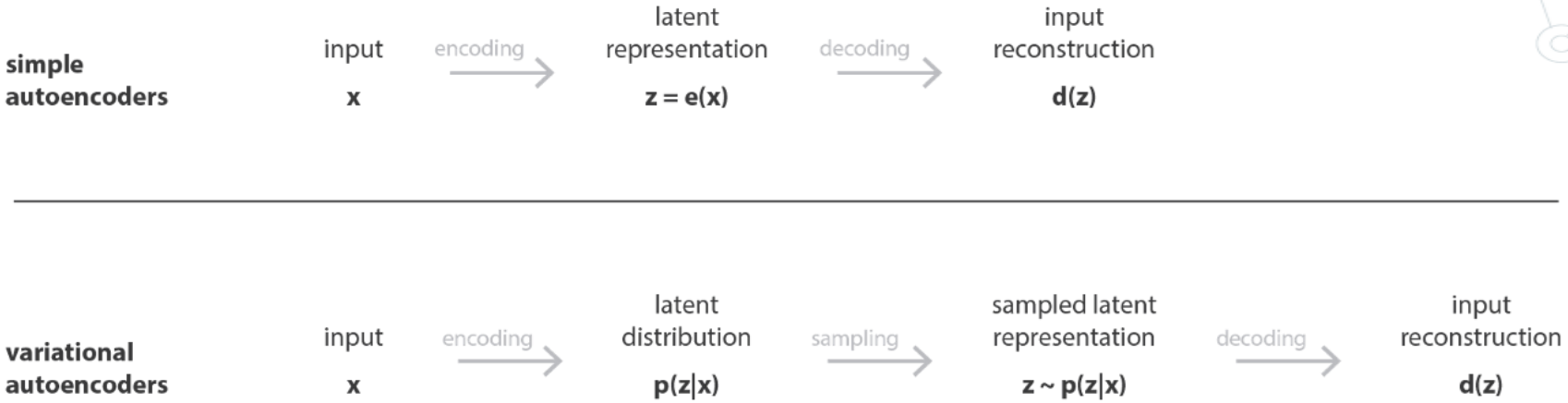
What is a regular latent space?



Variational Autoencoders (VAEs)

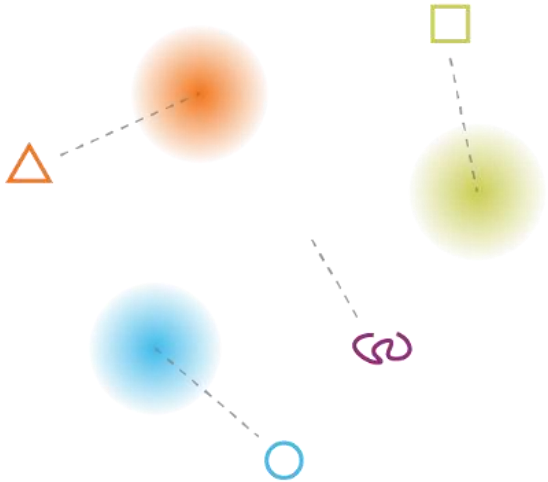
- Simultaneously discovered by Kingma and Welling in December 2013 and Rezende, Mohamed, and Wierstra in January 2014
- A generative model especially appropriate for the task of image editing using concept vectors
- Modern version of **autoencoders**
  - Autoencoders are networks that encode an input to a low-dimensional latent space and then decode it back
  - The decoder learns how to reconstruct the original inputs with some added noise to create new images
  - Learn to compress the input data into fewer bits of information

Variational autoencoders are not deterministic

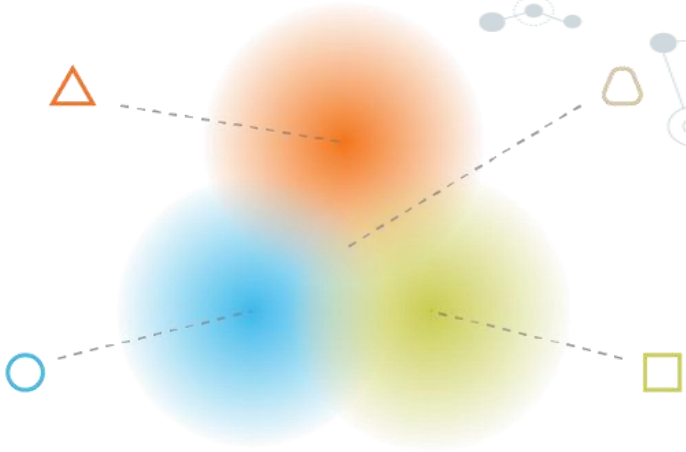


$$\text{loss} = ||x - \hat{x}||^2 + KL[N(\mu_x, \sigma_x), N(0, I)] = ||x - d(z)||^2 + KL[N(\mu_x, \sigma_x), N(0, I)]$$

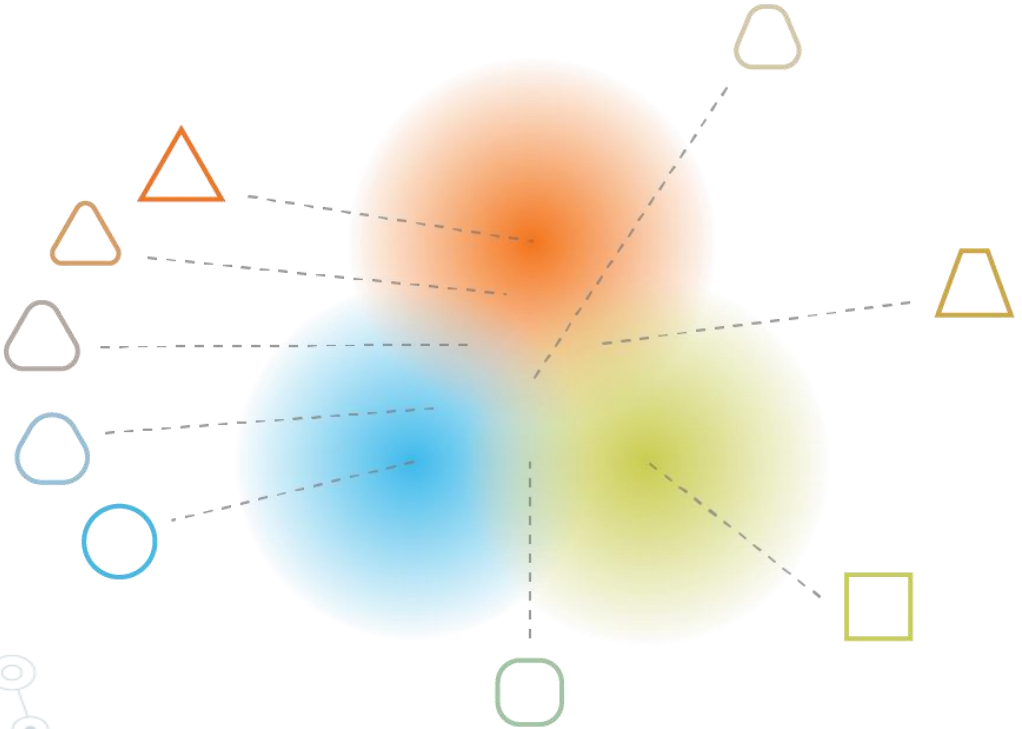
Intuition of a regular latent space



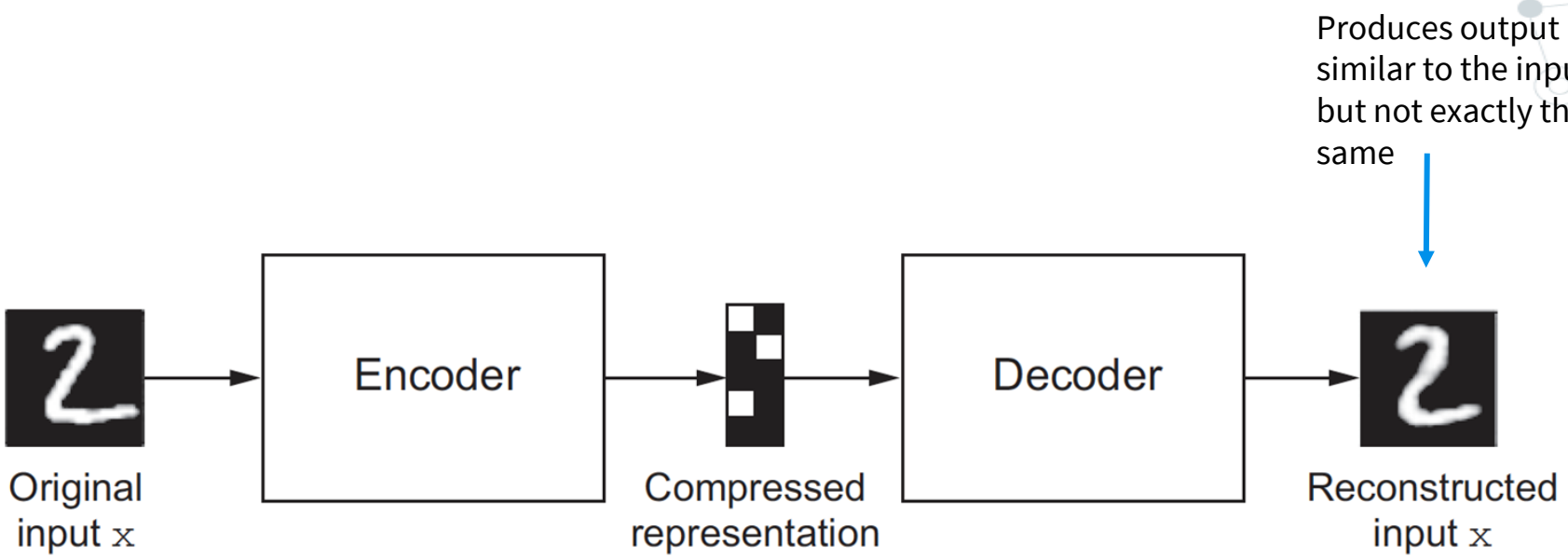
what can happen without regularisation



what we want to obtain with regularisation



Variational Autoencoders (VAEs)



VAEs

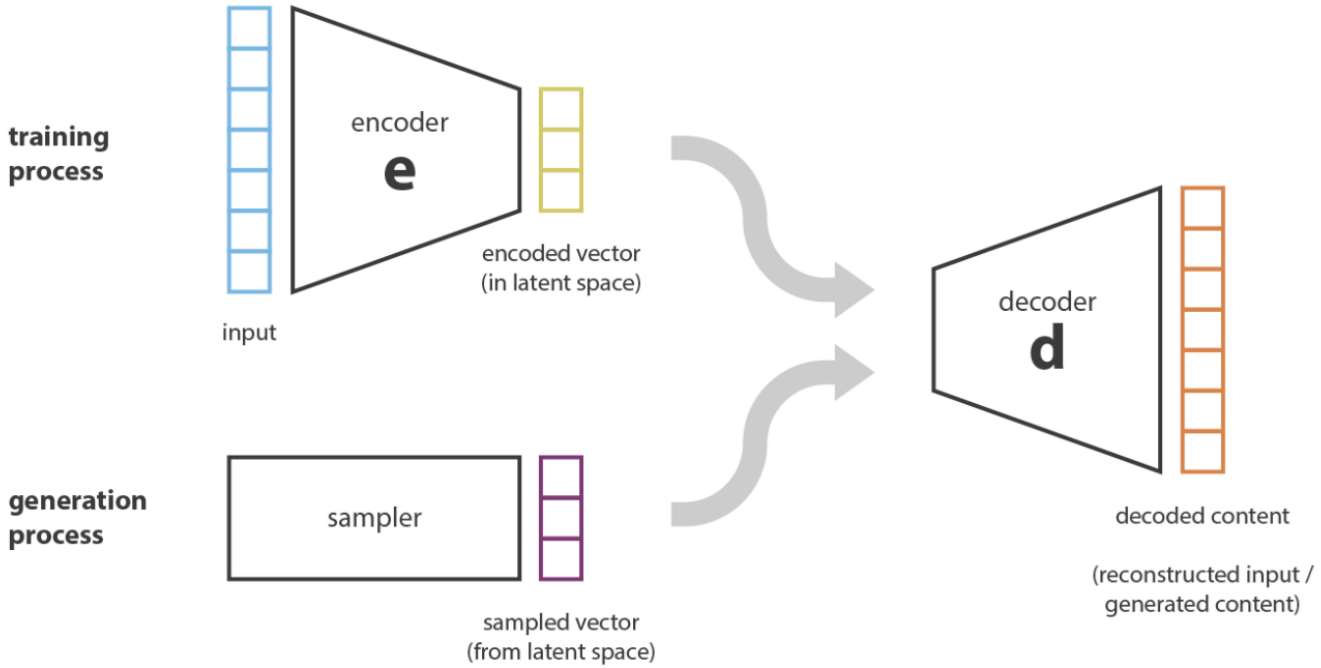
- Autoencoders aren’t great at creating nicely structured latent spaces and they also don’t generate anything new
- VAEs augment autoencoders to learn highly structured latent spaces and are able to generate new images
- Are regularized versions of autoencoders

VAEs

- Rather than compressing the image into a fixed code in the latent space, VAEs turn the image into the **parameters of a statistical distribution** (a mean and variance)
  - Assume the input image has been generated by a statistical process and that the **randomness** of this process should be taken into account when encoding and decoding
  - A VAE uses the mean and variance parameters to randomly sample one element from the distribution and decodes that element back to the original input
    - This process improves robustness and forces the latent space to encode meaningful representations everywhere: every point in the latent space is decoded to a valid output

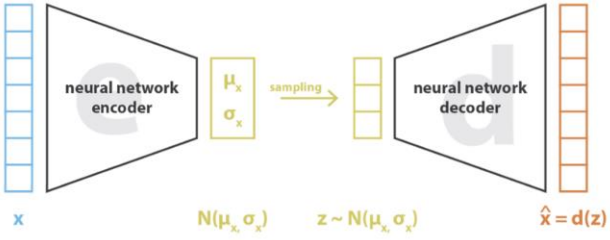
Autoencoders vs VAEs

VAEs



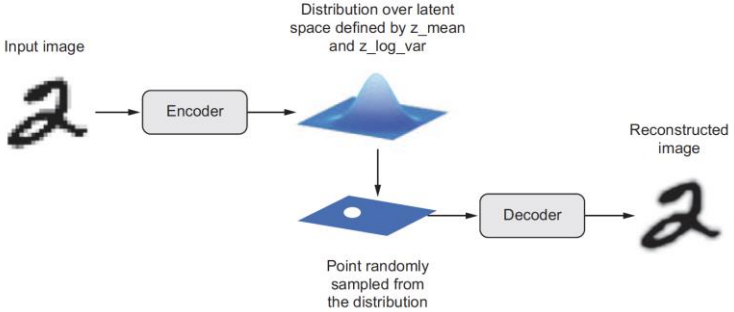
We can generate new data by decoding points that are randomly sampled from the latent space. The quality and relevance of generated data depend on the regularity of the latent space.

VAEs



$$\text{loss} = ||x - \hat{x}||^2 + KL[N(\mu_x, \sigma_x), N(0, I)] = ||x - d(z)||^2 + KL[N(\mu_x, \sigma_x), N(0, I)]$$

In variational autoencoders, the loss function is composed of a reconstruction term (that makes the encoding-decoding scheme efficient) and a regularisation term (that makes the latent space regular).



VAEs

- Algorithm steps:
  - An encoder module turns the input samples into two parameters in a latent space of representations, **z_mean** and **z_log_variance**
  - Randomly sample a point z from the latent normal distribution that's assumed to generate the input image, via

    z = z_mean + exp(z_log_variance) *epsilon

    where epsilon is a random tensor of small values
  - A decoder module maps this point in the latent space back to the original input image

- Having epsilon be random ensures every point that's close to the latent location where you encoded the input image can be decoded to something similar to the input image - but not exactly the same
  - Also forces every direction in the latent space to encode a meaningful axis of variation of the data, making the latent space very structured and highly suitable to manipulation via concept vectors

VAE Training

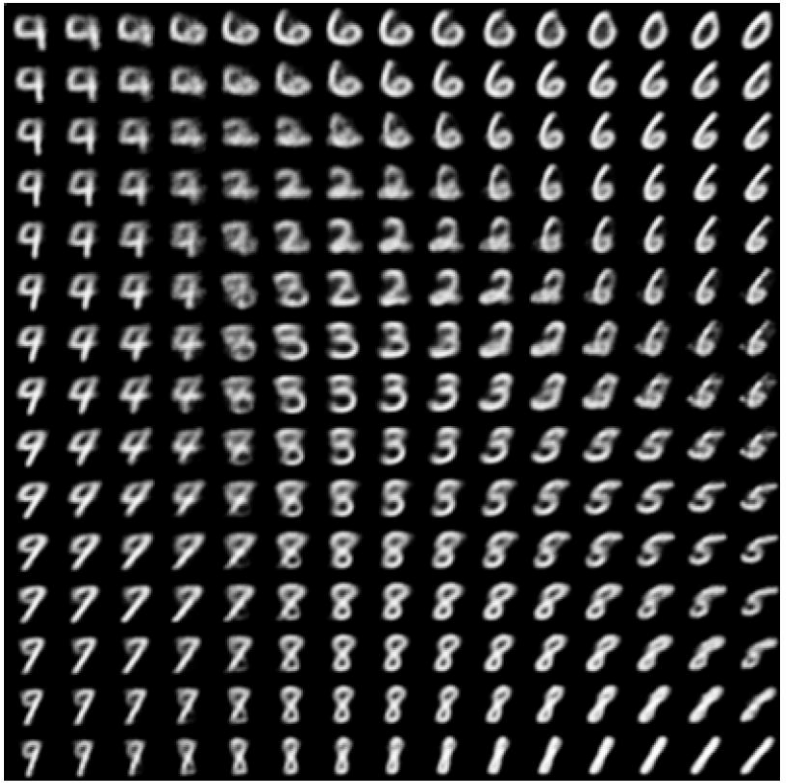
- Parameters are trained using two loss functions
  - **Reconstruction (generative) loss**: measures how accurately the network reconstructed the images; forces the decoded samples to match the initial inputs
  - **Regularization (latent) loss**: measures how closely the latent variables match a unit Gaussian distribution; helps learn well-formed latent spaces and reduce overfitting to the training data

- Because this is a different type of loss, Keras allows you to have a custom loss function and add it as a layer to your network using the **add_loss** layer

VAE on MNIST Data

Digits decoded from the latent space

Note how one digit morphs into another and that directions in this space have specific meaning (one direction for “four-ness”, one direction for “one-ness”, etc.



VAEs for Medicine

Extracting a biologically relevant latent space from cancer transcriptomes with variational autoencoders

Gregory P. Way and
Genomics and Computational Biology Graduate Program, University of Pennsylvania, Philadelphia, PA 19104, USA

Casey S. Greene*
Department of Systems Pharmacology and Translational Therapeutics, University of Pennsylvania, Philadelphia, PA 19104, USA

Abstract

The Cancer Genome Atlas (TCGA) has profiled over 10,000 tumors across 33 different cancer-types for many genomic features, including gene expression levels. Gene expression measurements capture substantial information about the state of each tumor. Certain classes of deep neural network models are capable of learning a meaningful latent space. Such a latent space could be used to explore and generate hypothetical gene expression profiles under various types of molecular and genetic perturbation. For example, one might wish to use such a model to predict a tumor's response to specific therapies or to characterize complex gene expression activations existing in differential proportions in different tumors. Variational autoencoders (VAEs) are a deep

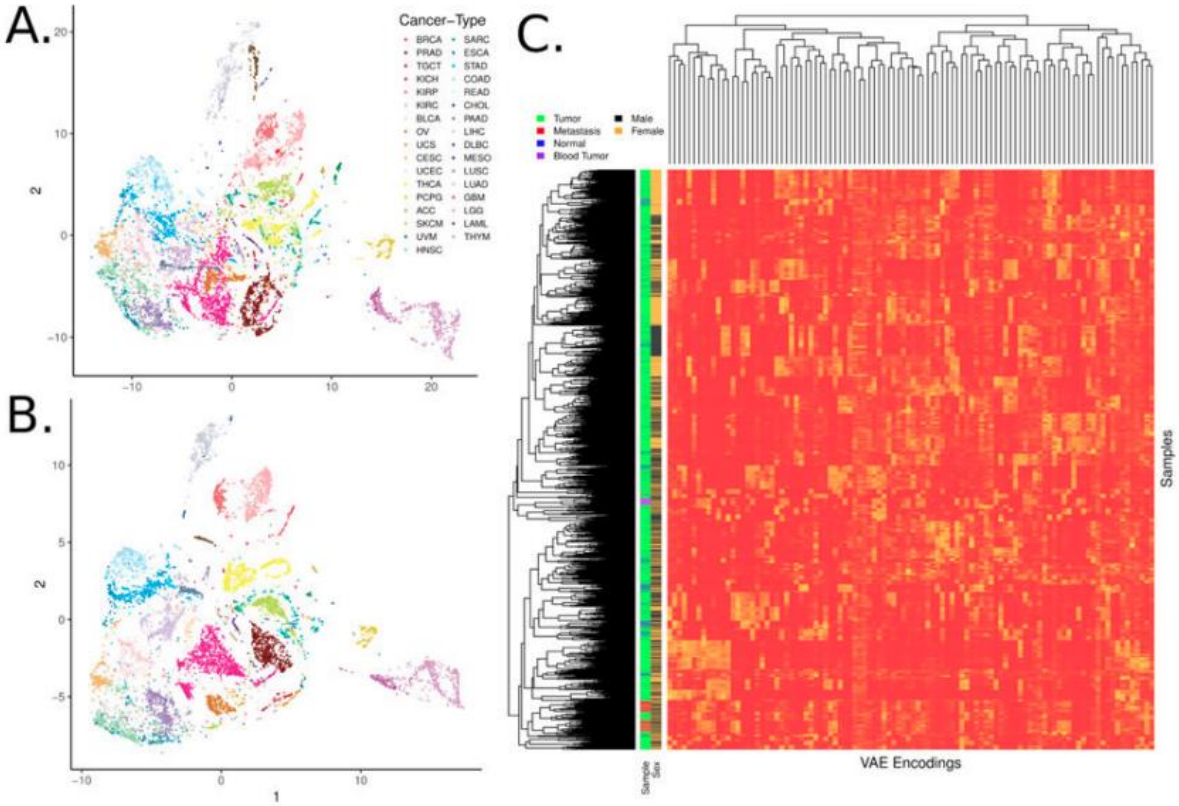


Fig. 2. Samples encoded by a variational autoencoder retain biological signals
(**A**) t-distributed stochastic neighbor embedding (t-SNE) of TCGA pan-cancer tumors with Tybalt encoded features. (**B**) t-SNE of 0-1 normalized gene expression features. Tybalt retains similar signals as compared to uncompressed gene expression data. (**C**) Full Tybalt encoding features by TCGA pan-cancer sample heatmap. Given on the y axis are the patients sex and type of sample.

setup.py	add util import to init (#133)	a year ago
tsne_tybalt_features.ipynb	Adding ADAGE model to tsne visualization (#114)	a year ago
tybalt_twohidden.ipynb	Add Two Hidden Layer Model (#81)	2 years ago
tybalt_vae.ipynb	Update conda environments (#108)	a year ago

README.md

Tybalt 

A Variational Autoencoder trained on Pan-Cancer Gene Expression

Gregory Way and Casey Greene 2017

DOI 10.5281/zenodo.1047069

The repository stores scripts to train, evaluate, and extract knowledge from a variational autoencoder (VAE) trained on 33 different cancer-types from The Cancer Genome Atlas (TCGA).

The specific VAE model is named *Tybalt* after an instigative, cat-like character in Shakespeare's "Romeo and Juliet". Just as the character Tybalt sets off the series of events in the play, the model Tybalt begins the foray of VAE manifold learning in transcriptomics. Also, deep unsupervised learning likes cats.

We discuss the training and evaluation of Tybalt in our PSB paper:

Extracting a Biologically Relevant Latent Space from Cancer Transcriptomes with Variational Autoencoders.

VAEs for Medicine

Dr.VAE: Drug Response Variational Autoencoder

Ladislav Rampasek*†‡ Daniel Hidru†‡ Petr Smirnov§
Benjamin Haibe-Kains§¶‖ Anna Goldenberg*‡†

Abstract

We present two deep generative models based on Variational Autoencoders to improve the accuracy of drug response prediction. Our models, Perturbation Variational Autoencoder and its semi-supervised extension, Drug Response Variational Autoencoder (Dr.VAE), learn latent representation of the underlying gene states before and after drug application that depend on: (i) drug-induced biological change of each gene and (ii) overall treatment response outcome. Our VAE-based models outperform the current published benchmarks in the field by anywhere from 3 to 11% AUROC and 2 to 30% AUPR. In addition, we found that better reconstruction accuracy does not necessarily lead to improvement in classification accuracy and that jointly trained models perform better than models that minimize reconstruction error independently.

Generative Adversarial Networks (GANs)

GANs

- Introduced by Goodfellow et al in 2014
- An alternative to VAEs for learning latent spaces of images
- Enable generation of fairly realistic (have been increasingly realistic over time) synthetic images by forcing the generated images to be statistically almost indistinguishable from real ones
- Made of 2 parts:
  - **Generator network**: takes as input a random vector (a random point in the latent space) and decodes it into a synthetic image - trained to fool the discriminator network
  - **Discriminator network (or adversary)**: takes as input an image (real or synthetic) and predicts whether the image came from the training set or was created by the generator network

Intuition Examples

Think of someone trying to create counterfeit money who has a spy inside a bank. They can create the fake money and try to slip it past back employees. The bank employees will notice the fake money easily in the beginning, but as the spy relays information to the counterfeiter, they will make better fake versions of money that will be more difficult for bank employees to distinguish as fake. Meanwhile, the bank will come up with new ways of detecting the updated counterfeit money.

Intuition Examples

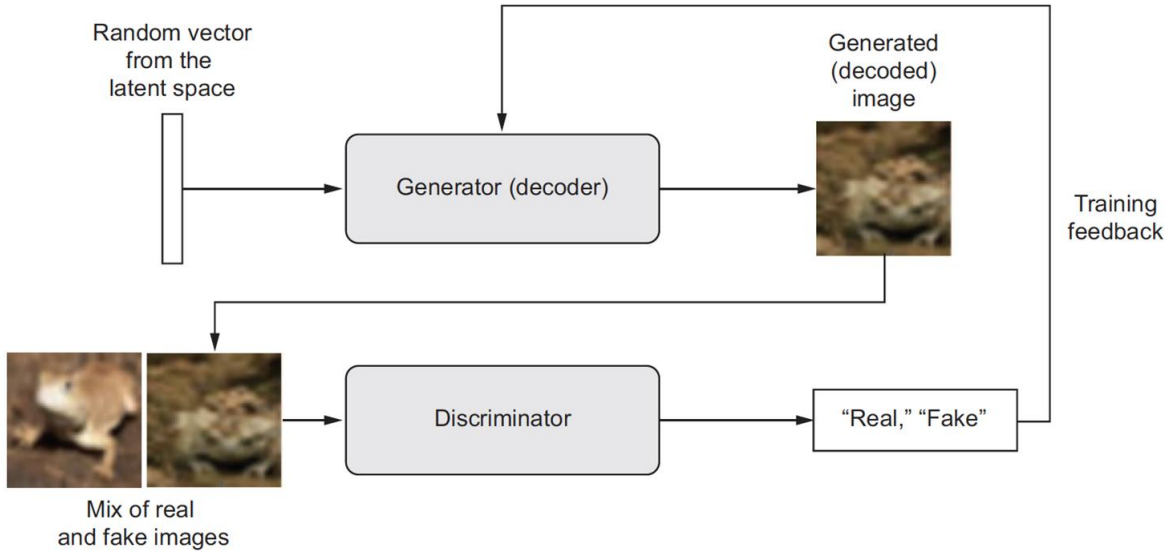
Think of someone trying to create counterfeit money who has a spy inside a bank. They can create the fake money and try to slip it past back employees. The bank employees will notice the fake money easily in the beginning, but as the spy relays information to the counterfeiter, they will make better fake versions of money that will be more difficult for bank employees to distinguish as fake. Meanwhile, the bank will come up with new ways of detecting the updated counterfeit money.

Intuition Examples

Now think of someone trying to forge Picasso paintings. They will be bad in the beginning, but will improve over time with feedback from an expert art dealer on how they detected the fake art from the real paintings. At the same time, new, more advanced techniques will be discovered to spot forged paintings.

GANs

- The generator learns to generate increasingly realistic images as it trains
- At the same time, the discriminator network is constantly adapting to the gradually improving capabilities of the generator
- After training, the generator is able to turn any point in its input space into a believable image
- Unlike VAEs, this latent space has fewer explicit guarantees of meaningful structure



Training

- The optimization minimum isn't fixed for GANs
  - Every step taken during gradient descent changes the entire landscape - the optimization process is dynamic
  - Need to find an equilibrium, not a minimum
    - There are 2 opposing forces here - the generator and the discriminator

- Very difficult to train
  - Many parameters to tune
  - Complex model architecture

Training

- Implementation of a deep convolutional GAN (DCGAN) in Keras
  - The generator and discriminator are deep CNNs
  - A **generator** network maps vectors to images
  - A **discriminator** network maps images to a binary score estimating the probability that the image is real
  - A gan network chains the generator and the discriminator together:

    gan(x) = discriminator(generator(x))

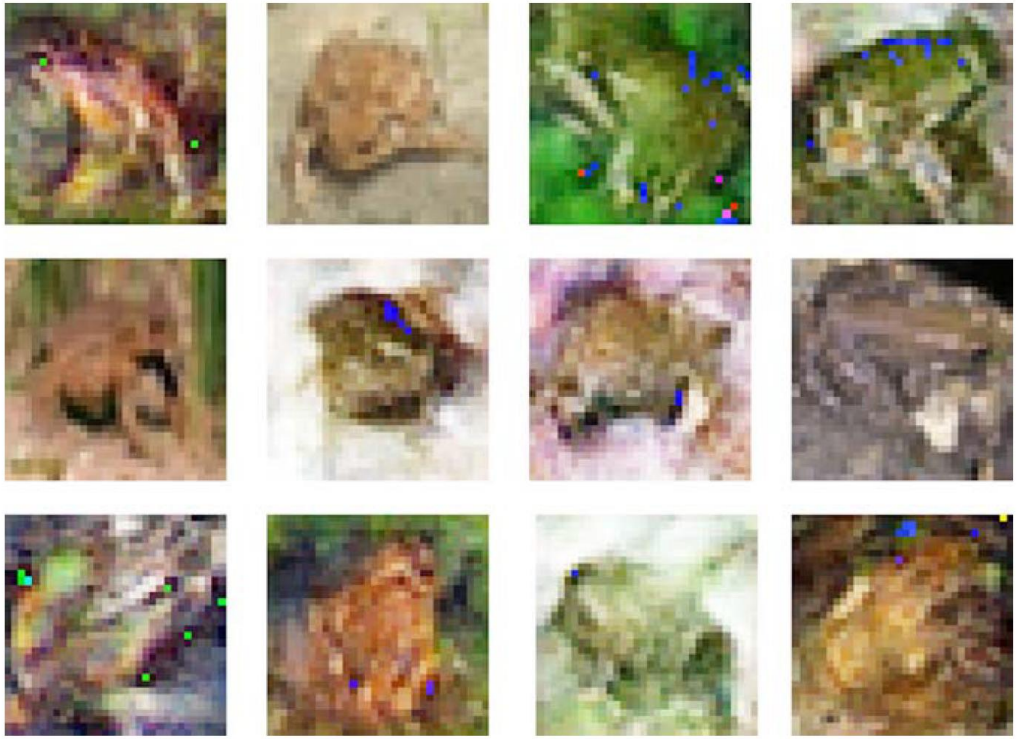
    Thus this gan network maps latent space vectors to the discriminator's assessment of the realism of these latent vectors as decoded by the generator

Training

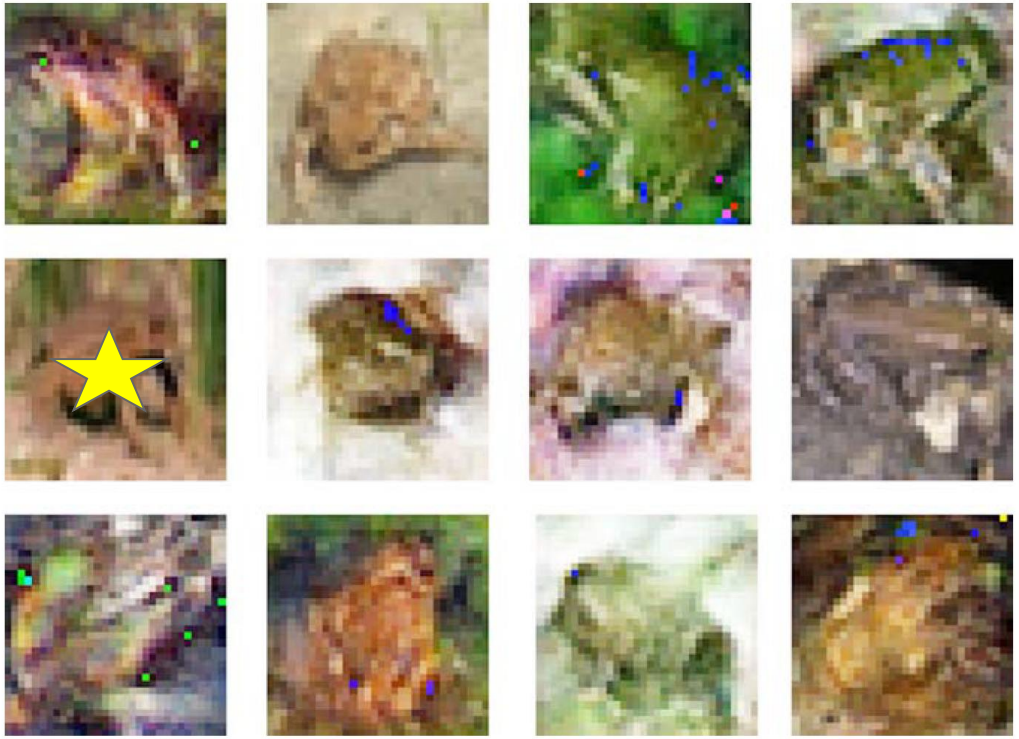
- You train the discriminator using examples of real and fake images along with “real”/“fake” labels, just as you train any regular image-classification model

- To train the generator, you use the gradients of the generator’s weights with regard to the loss of the GAN model. This means, at every step, you move the weights of the generator in a direction that makes the discriminator more likely to classify as “real” the images decoded by the generator. In other words, you train the generator to fool the discriminator

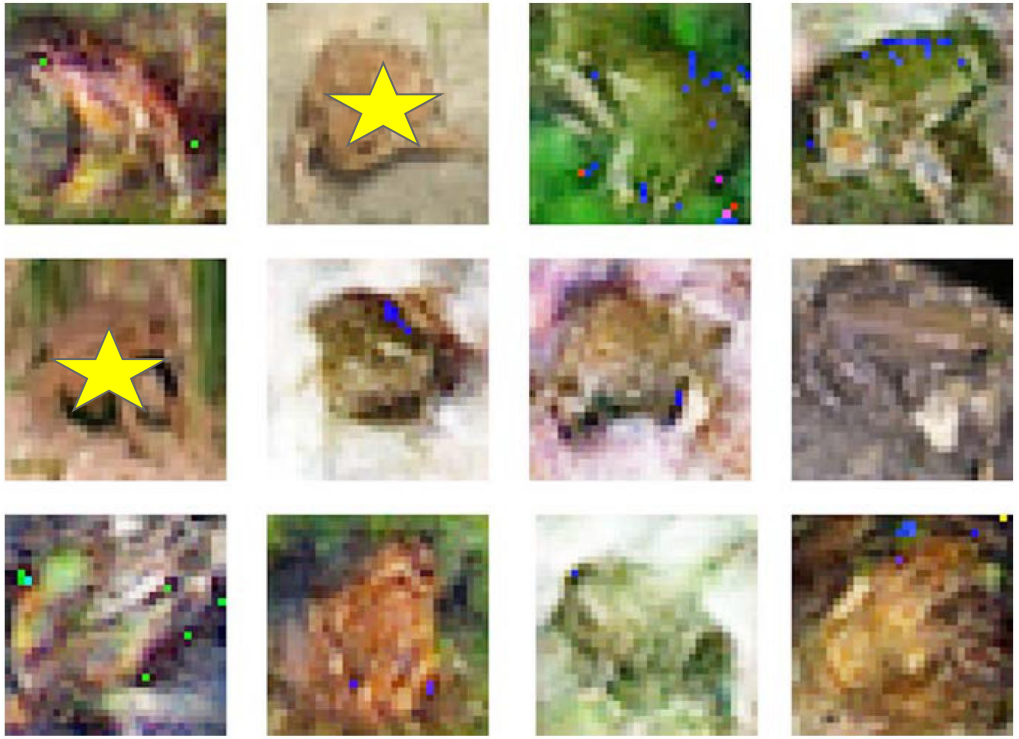
- Can you guess which image is from the training set (not created by the generator) in each column?



- Can you guess which image is from the training set (not created by the generator) in each column?



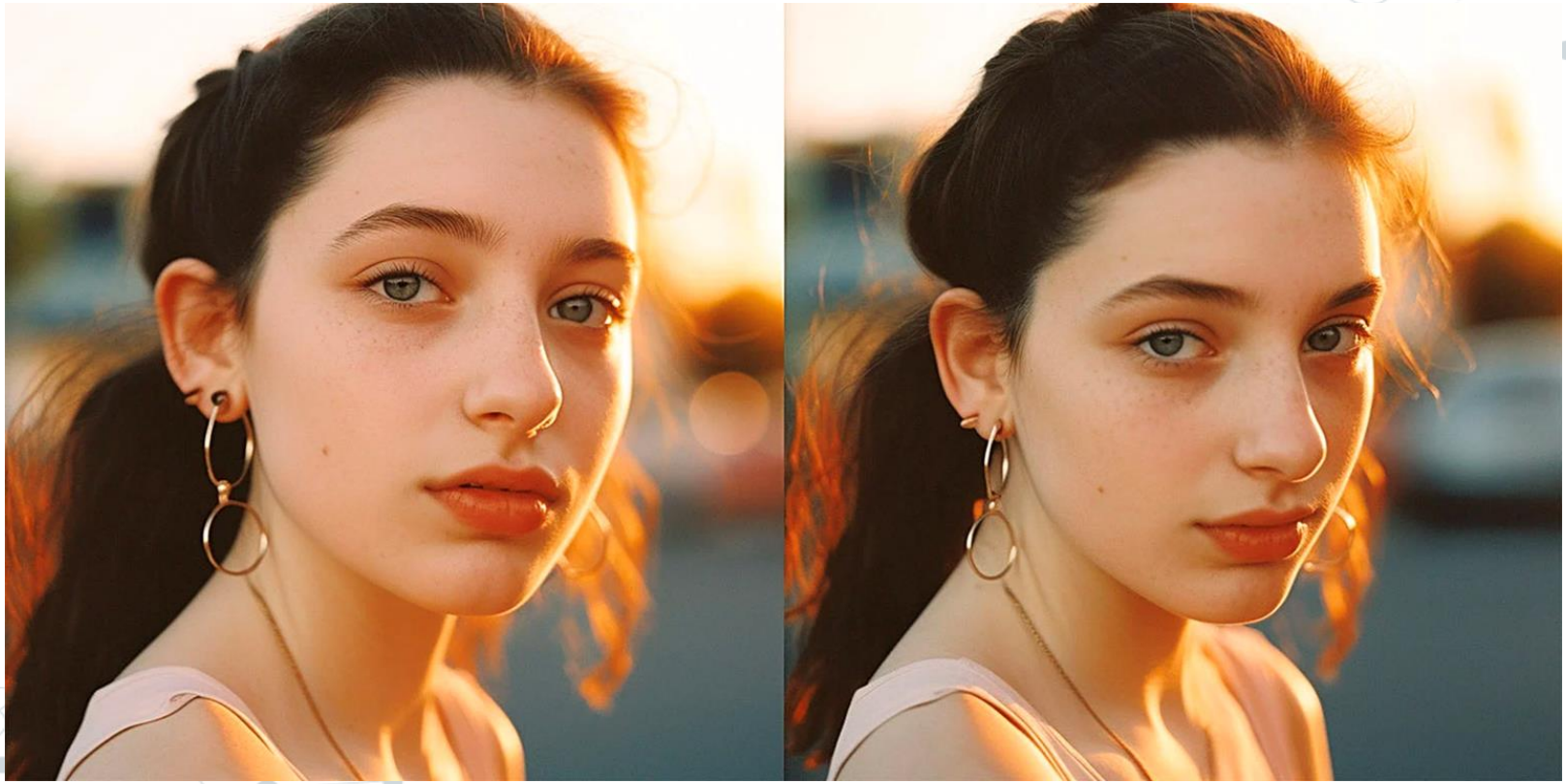
- Can you guess which image is from the training set (not created by the generator) in each column?



- Can you guess which image is from the training set (not created by the generator) in each column?

- Can you guess which image is from the training set (not created by the generator) in each column?


/imagine prompt::a mutant umbrella tree shaped like a nuclear bomb explosion



GANs for Medicine

https://arxiv.org/pdf/1809.06222.pdf

GANs for Medical Image Analysis

Salome Kazeminia[a,1], Christoph Baur[b,1], Arjan Kuijper[c], Bram van Ginneken[d], Nassir Navab[b], Shadi Albarqouni[b], Anirban Mukhopadhyay[a]

[a]*Department of Computer Science, TU Darmstadt, Germany*
[b]*Computer Aided Medical Procedures (CAMP), TU Munich, Germany*
[c]*Fraunhofer IGD, Darmstadt, Germany*
[d]*Radboud University Medical Center, Nijmegen, The Netherlands*

Abstract

Generative Adversarial Networks (GANs) and their extensions have carved open many exciting ways to tackle well known and challenging medical image analysis problems such as medical image de-noising, reconstruction, segmentation, data simulation, detection or classification. Furthermore, their ability to synthesize images at unprecedented levels of realism also gives hope that the chronic scarcity of labeled data in the medical field can be resolved with the help of these generative models. In this review paper, a broad overview of recent literature on GANs for medical applications is given, the shortcomings and opportunities of the proposed methods are thoroughly discussed and potential future work is elaborated. We review the most relevant papers published until the submission date. For quick access, important details such as the underlying method, datasets and performance are tabulated. An interactive visualization categorizes all papers to keep the review alive[2].

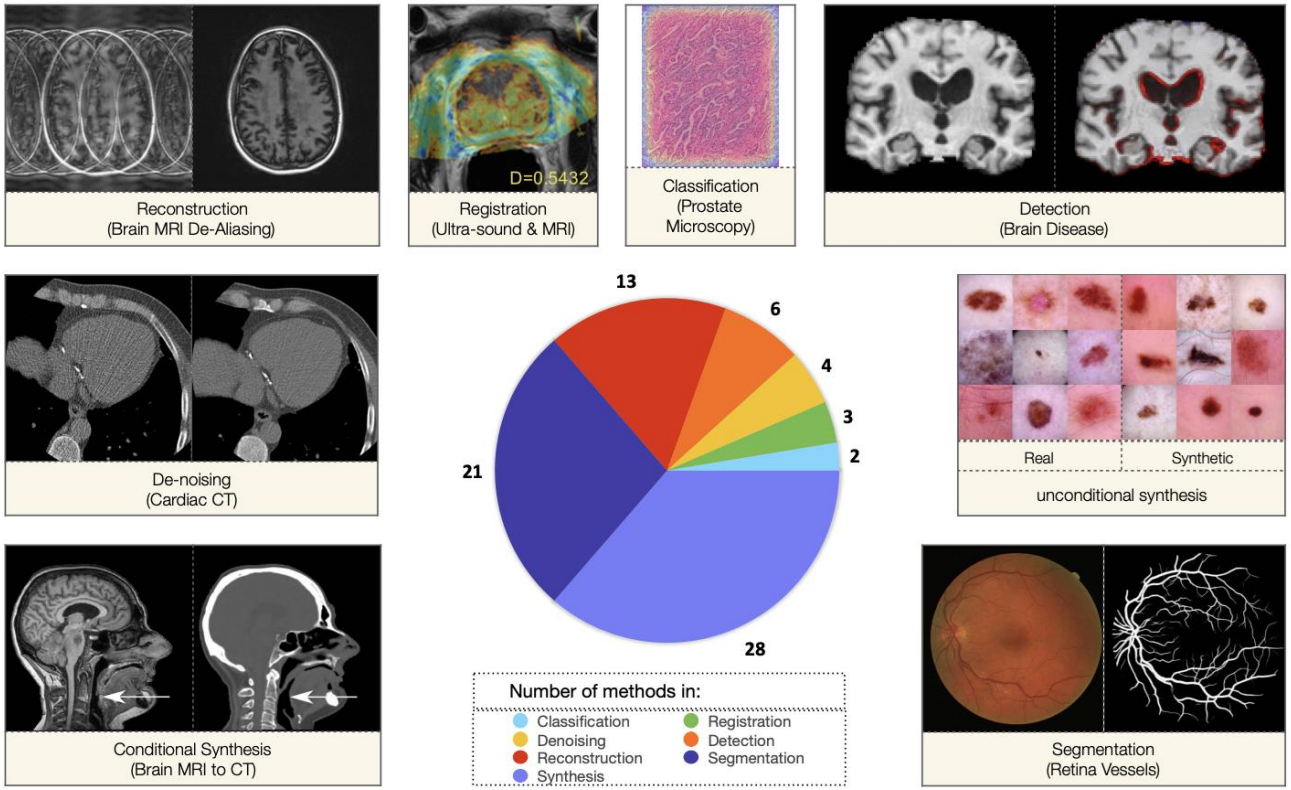


Figure 1: The pie chart of distribution of papers and visual examples of GAN functionality among the different applications. Examples are taken from papers as the following: Conditional synthesis Wolterink et al. (2017a), Denoising Wolterink et al. (2017b), Reconstruction Zhang et al. (2018), Registration Yan et al. (2018), Classification Ren et al. (2018), Detection Baumgartner et al. (2018), Unconditional synthesis Baur et al. (2018b), and Segmentation Son et al. (2017).

*GAN

- A ton of different GAN architectures have been created
  - cGAN
  - SeGAN
  - DCGAN
  - ACGAN
  - CycleGAN
  - MGAN
  - LSGAN
  - VAE-GAN
  - WGAn
  - ...

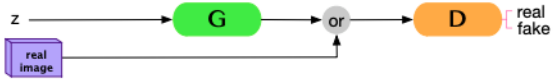


Figure 2: GAN

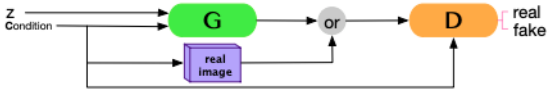


Figure 3: cGAN

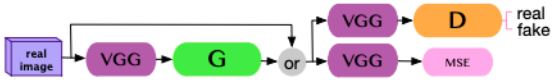


Figure 4: MGAN

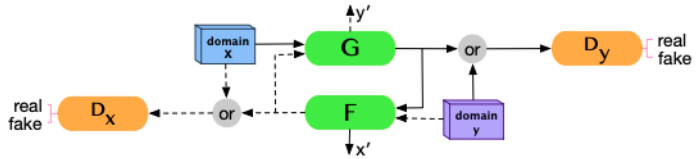


Figure 5: cycleGAN

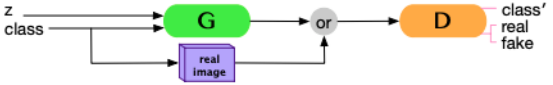
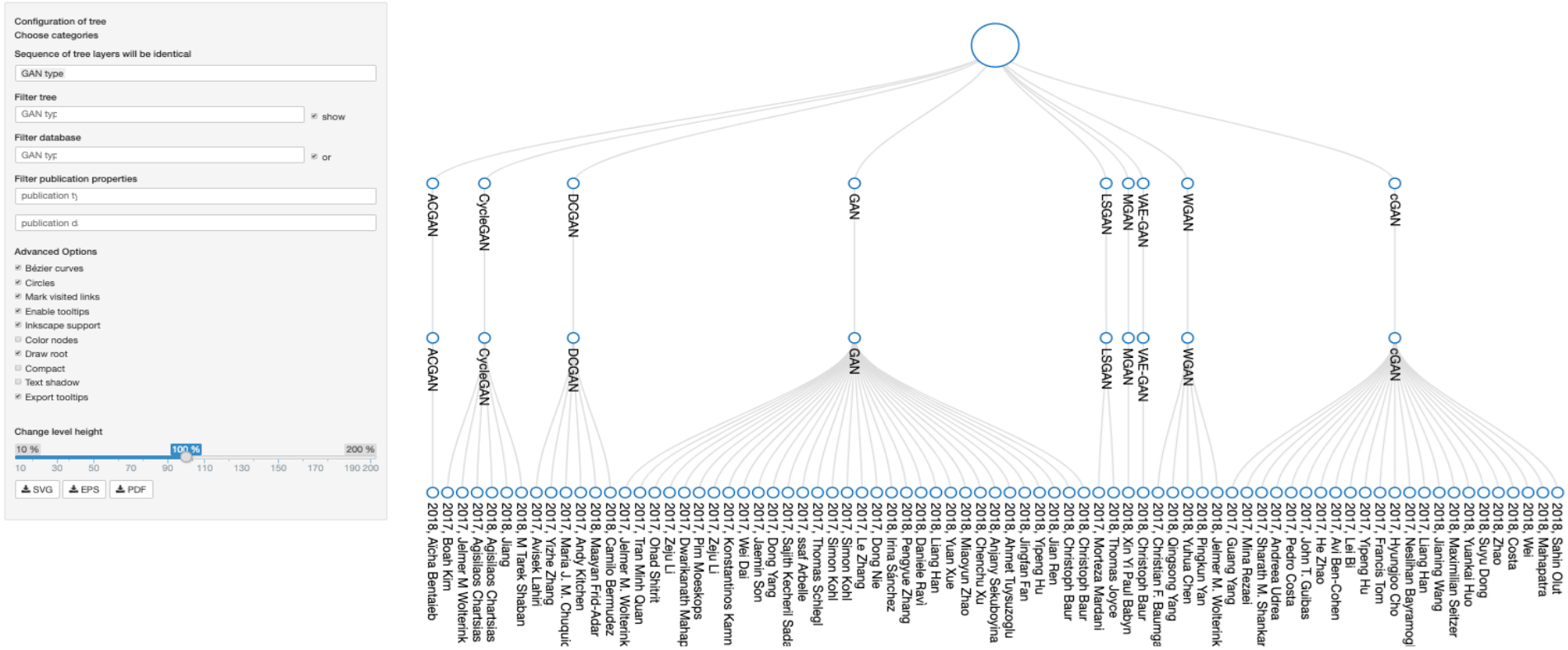


Figure 6: AC-GAN



http://livingreview.in.tum.de/GANs_for_Medical_Applications/

GANs for Medicine

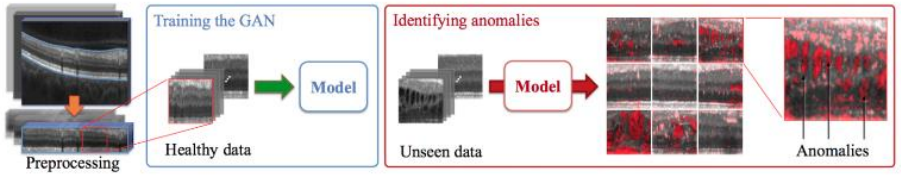
Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery

Thomas Schlegl[1,2] *, Philipp Seeböck[1,2], Sebastian M. Waldstein[2], Ursula Schmidt-Erfurth[2], and Georg Langs[1]

[1]Computational Imaging Research Lab, Department of Biomedical Imaging and Image-guided Therapy, Medical University Vienna, Austria
thomas.schlegl@meduniwien.ac.at
[2]Christian Doppler Laboratory for Ophthalmic Image Analysis, Department of Ophthalmology and Optometry, Medical University Vienna, Austria

Abstract. Obtaining models that capture imaging markers relevant for disease progression and treatment monitoring is challenging. Models are typically based on large amounts of data with annotated examples of known markers aiming at automating detection. High annotation effort and the limitation to a vocabulary of known markers limit the power of such approaches. Here, we perform unsupervised learning to identify anomalies in imaging data as candidates for markers. We propose *AnoGAN*, a deep convolutional generative adversarial network to learn a manifold of normal anatomical variability, accompanying a novel anomaly scoring scheme based on the mapping from image space to a latent space. Applied to new data, the model labels anomalies, and scores image patches indicating their fit into the learned distribution. Results on optical coherence tomography images of the retina demonstrate that the approach correctly identifies anomalous images, such as images containing retinal fluid or hyperreflective foci.

https://arxiv.org/pdf/1703.05921.pdf



AnoGAN

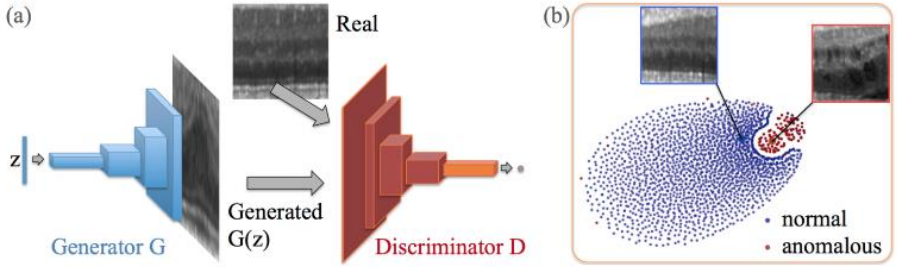


Fig. 2. (a) Deep convolutional generative adversarial network. (b) t-SNE embedding of normal (blue) and anomalous (red) images on the feature representation of the last convolution layer (orange in (a)) of the discriminator.

AnoGAN