Aprendizado Profundo (Deep Learning)

Autoencoders

Dario Oliveira (dario.oliveira@fgv.br)

Overview

Introduction

Autoencoders

Convolutional autoencoders

Denoising autoencoders

Variational autoencoders

Scarce labeled data

DL demands many labeled data

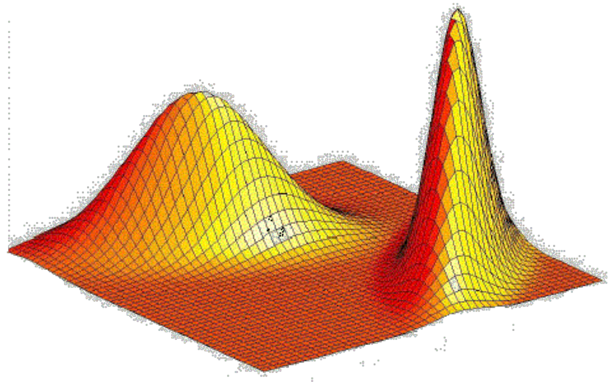
Lack of annotated RS data

Wealth of unlabeled RS data

Go unsupervised!

Generative Models

learn the distribution underlying the training samples



generate new samples from the learned distribution

real training samples

generated samples

Generative Models

learn the distribution underlying the training samples

p_{model}

generate new samples from the learned distribution

Question: How can we learn $p_{model}(x)$ similar to $p_{data}(x)$?

$x \sim p_{data}$

$y \sim p_{model}$

training samples

generated samples

Why generative models

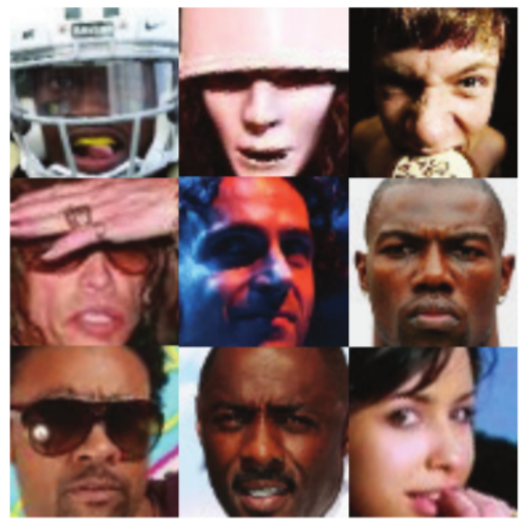
Debiasing: uncovers under and over represented training data.

Random Batch Sampling During Standard Face Detection Training

Homogenous skin color, pose

Mean Sample Prob: 7.57 x 10^{-6}

Batch Sampling During Training with Learned Debiaising



Diverse skin color, pose, illumination

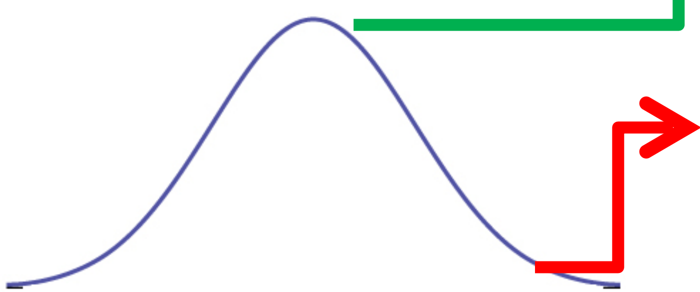
Mean Sample Prob: 1.03 x 10^{-4}

Source: Amini, et al., 2019 , [Uncovering and Mitigating Algorithmic Bias through Learned Latent Structure,]

Why generative models

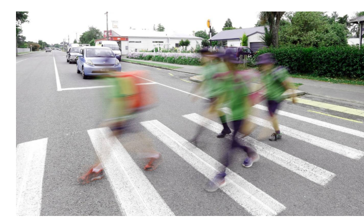
- **Problem**: detect something new/ rare
- **Strategy:** detect outliers in the distribution
- Use outliers during training to improve the model.

Most driving data
sunny, wide straight road



Edge Cases

Harsh Weather



Pedrestrians

What is a latent variable?

The myth of the cave

Can we learn the true explanatory factors, e.g., latent variables, from only observed data?

Source: https://www.studiobinder.com/blog/platos-allegory-of-the-cave/

Overview

Introduction

Autoencoders

Convolutional autoencoders

Denoising autoencoders

Variational autoencoders

Autoencoders

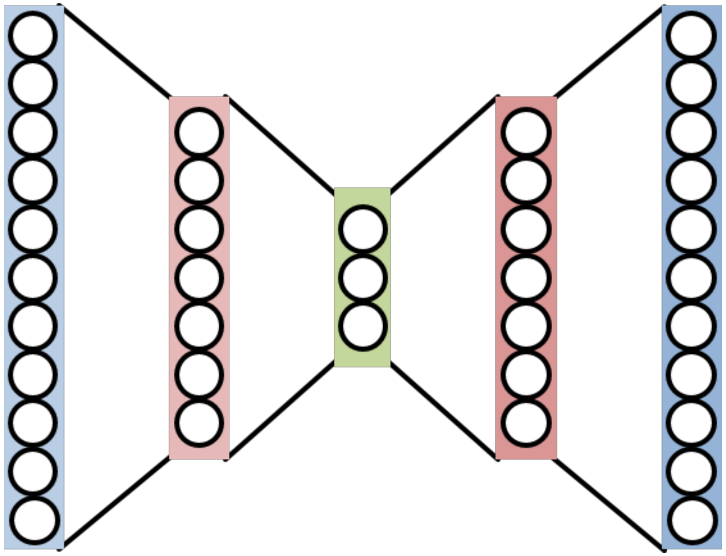
Unsupervised

Dimensionality Reduction

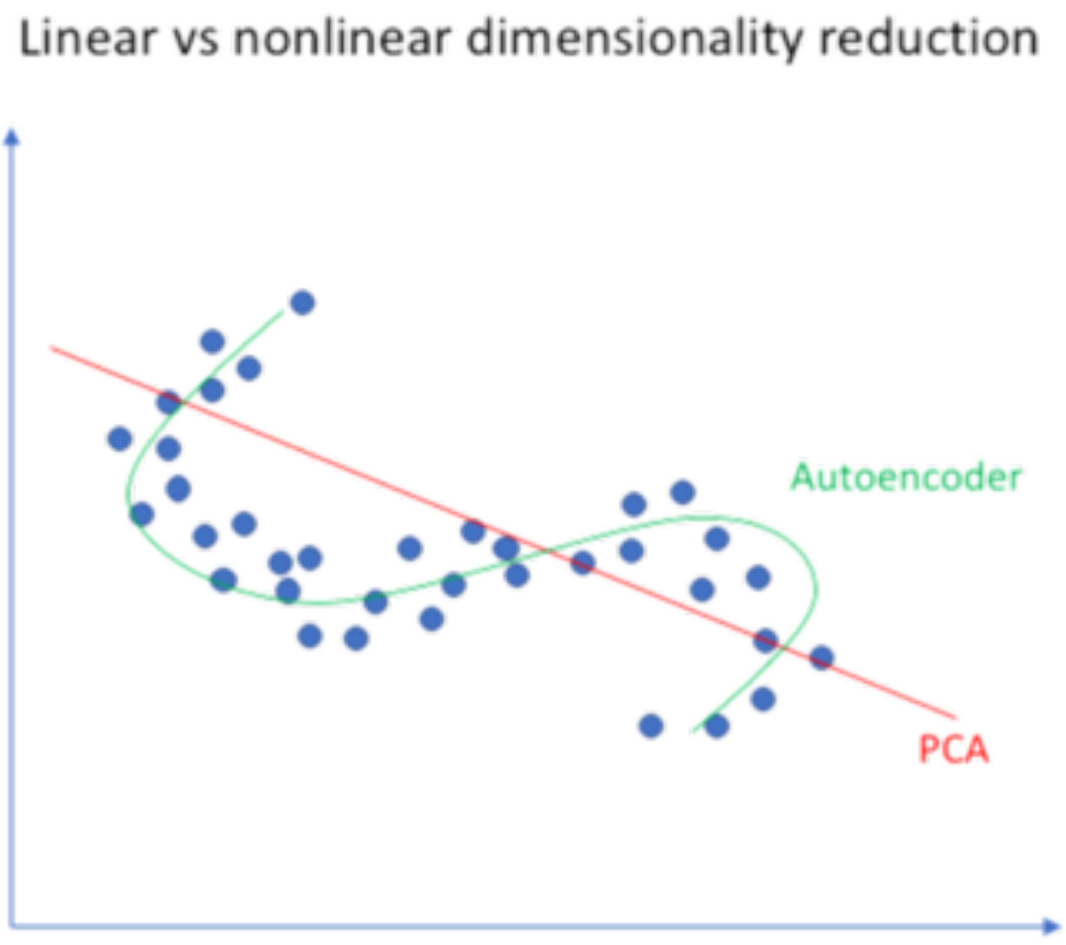
Feature learning

A feedforward network

Same training machinery



AE vs PCA



Source https://www.edureka.co/blog/autoencoders-tutorial/

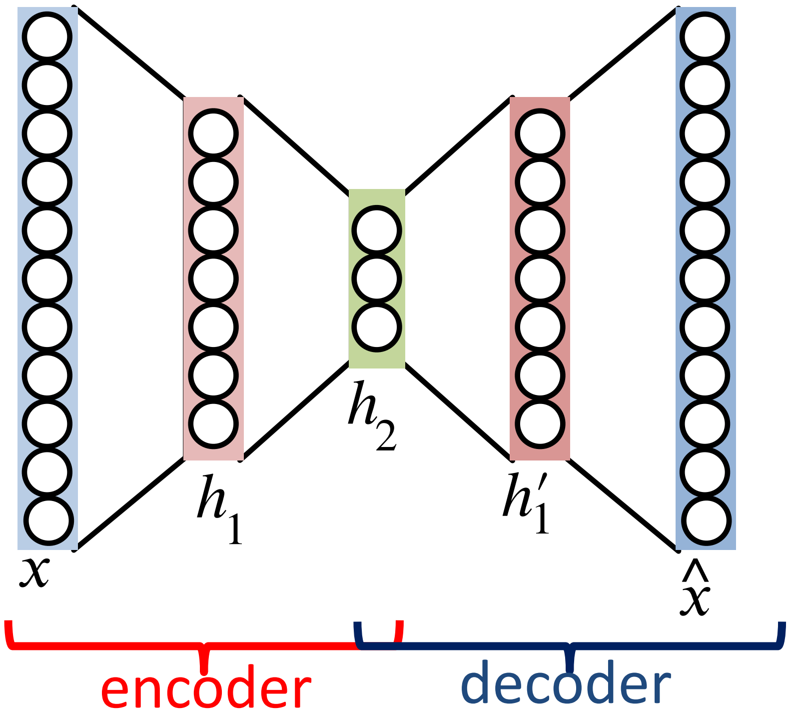
AE for dimensionality reduction

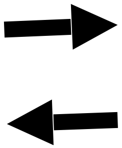
Aims at reproducing the input at the output.

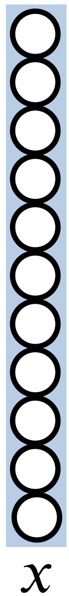
It comprises an encoder that contracts the data ...

followed by a decoder that recovers the original dimension.

At the bottleneck we have a latent space(?)





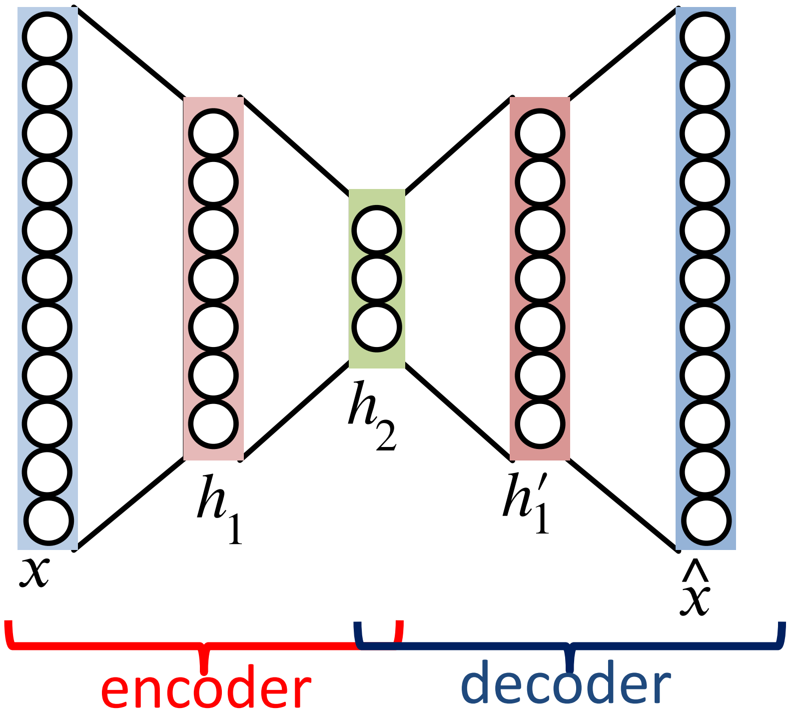


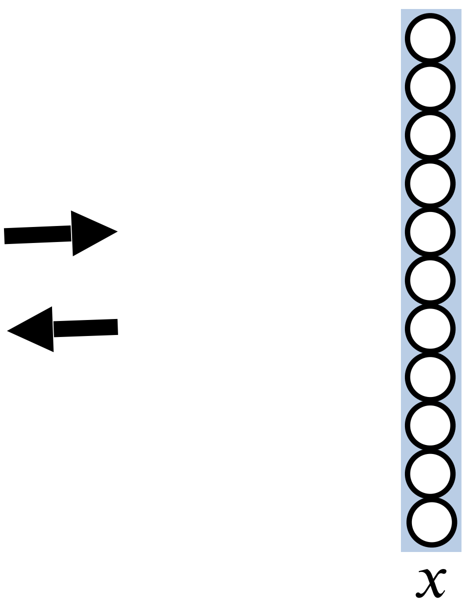
AE for dimensionality reduction

Aims at reproducing the input at the output.

The loss function relates to the similarity input ↔ output (e.g., L_2)

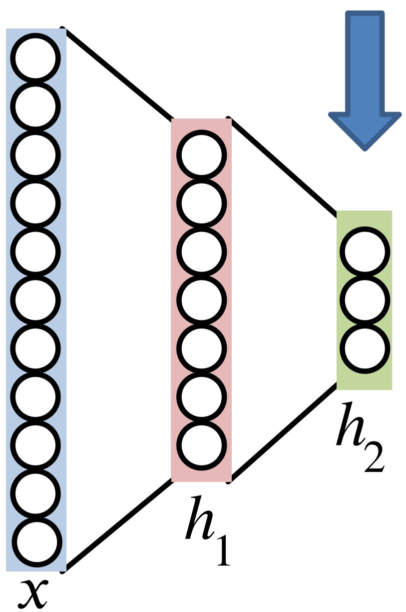
$$L(\boldsymbol{W}) = \frac{1}{N}\sum_i \left\| x_i - \hat{x}_i \right\|^2 + \lambda\, R(\boldsymbol{W})$$





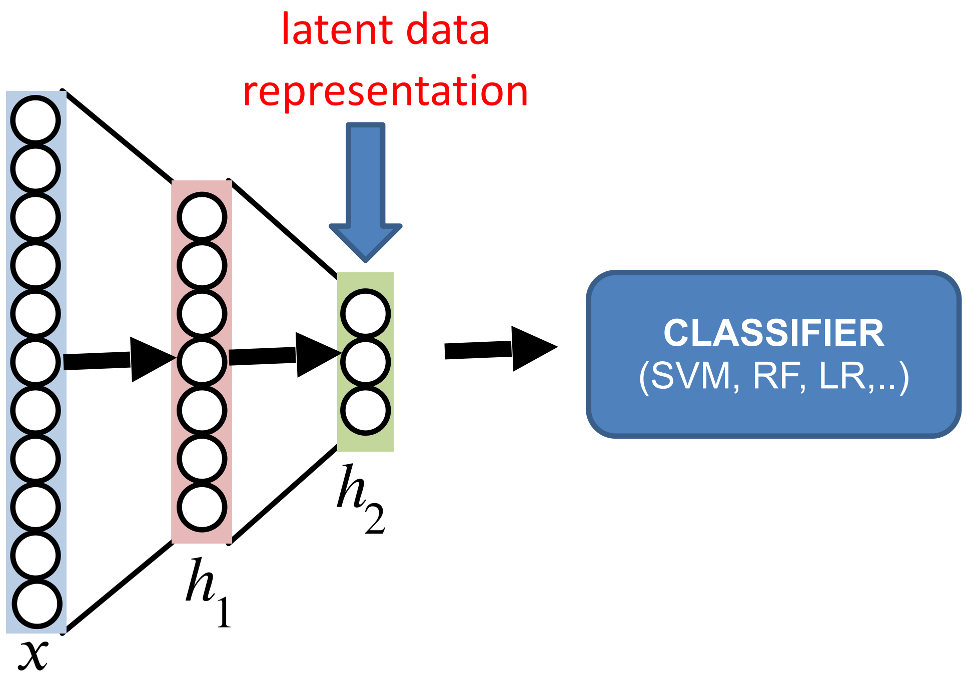
AE for dimensionality reduction

Actually, we are not interested in the output but in the "bottleneck that captures the most salient features of the training data, the latent representation.



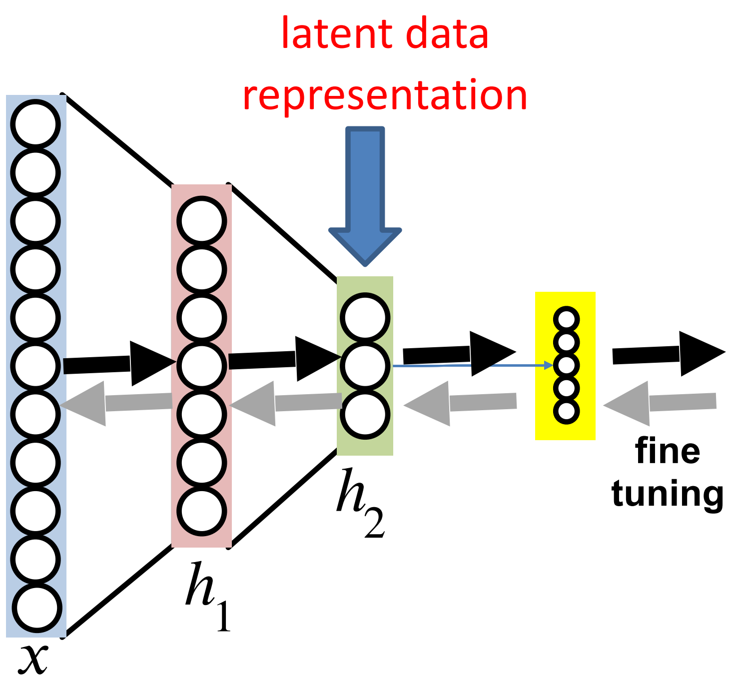
Use of AE features

as input to some ML classifier



Use of AE features

as initialization for subsequent fine tuning



Overview

Introduction

Autoencoders

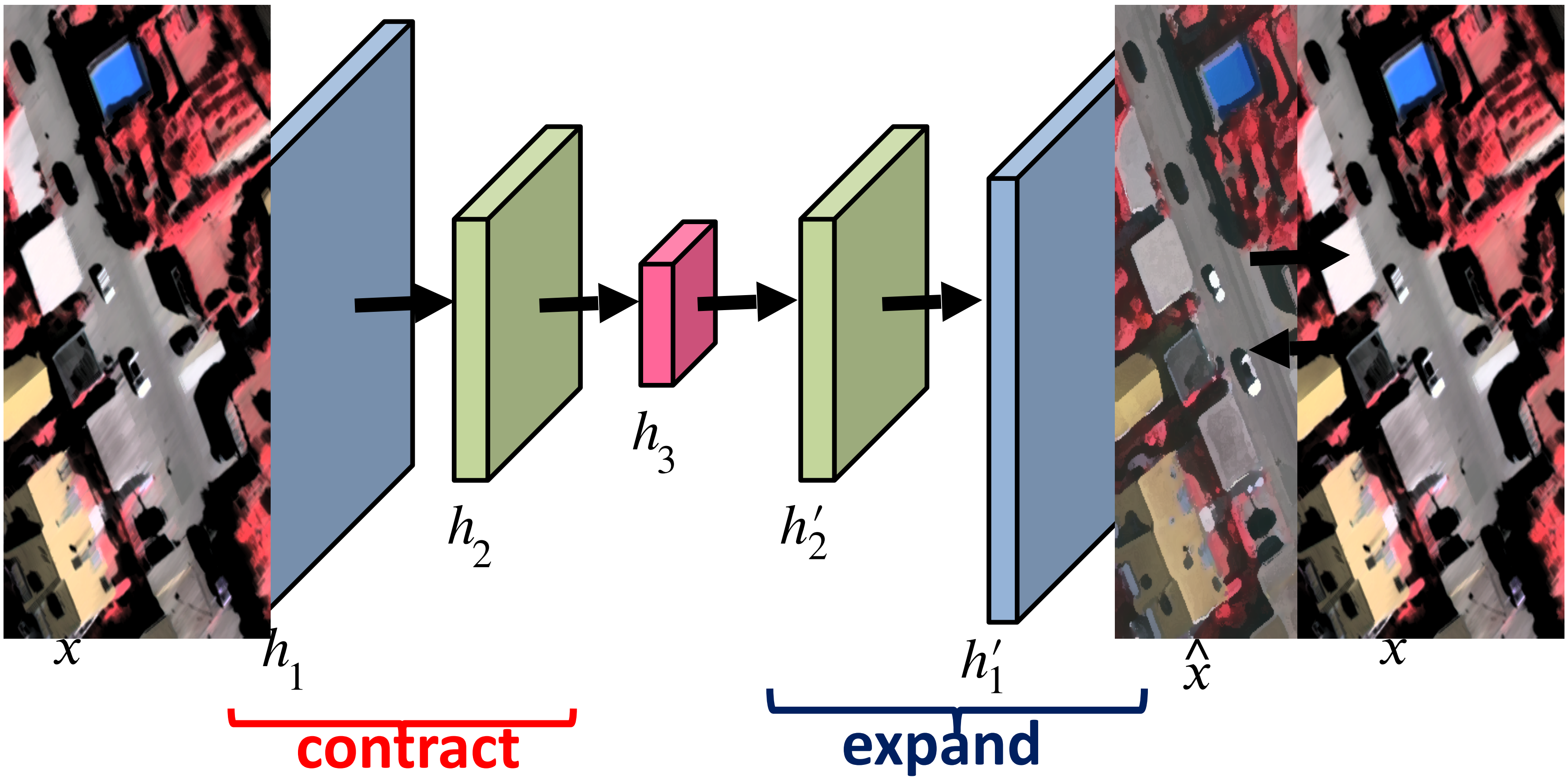
Convolutional autoencoders

Denoising autoencoders

Variational autoencoders

Convolutional Autoenconders

Aim at reproducing the input at the output



Overview

Introduction

Autoencoders

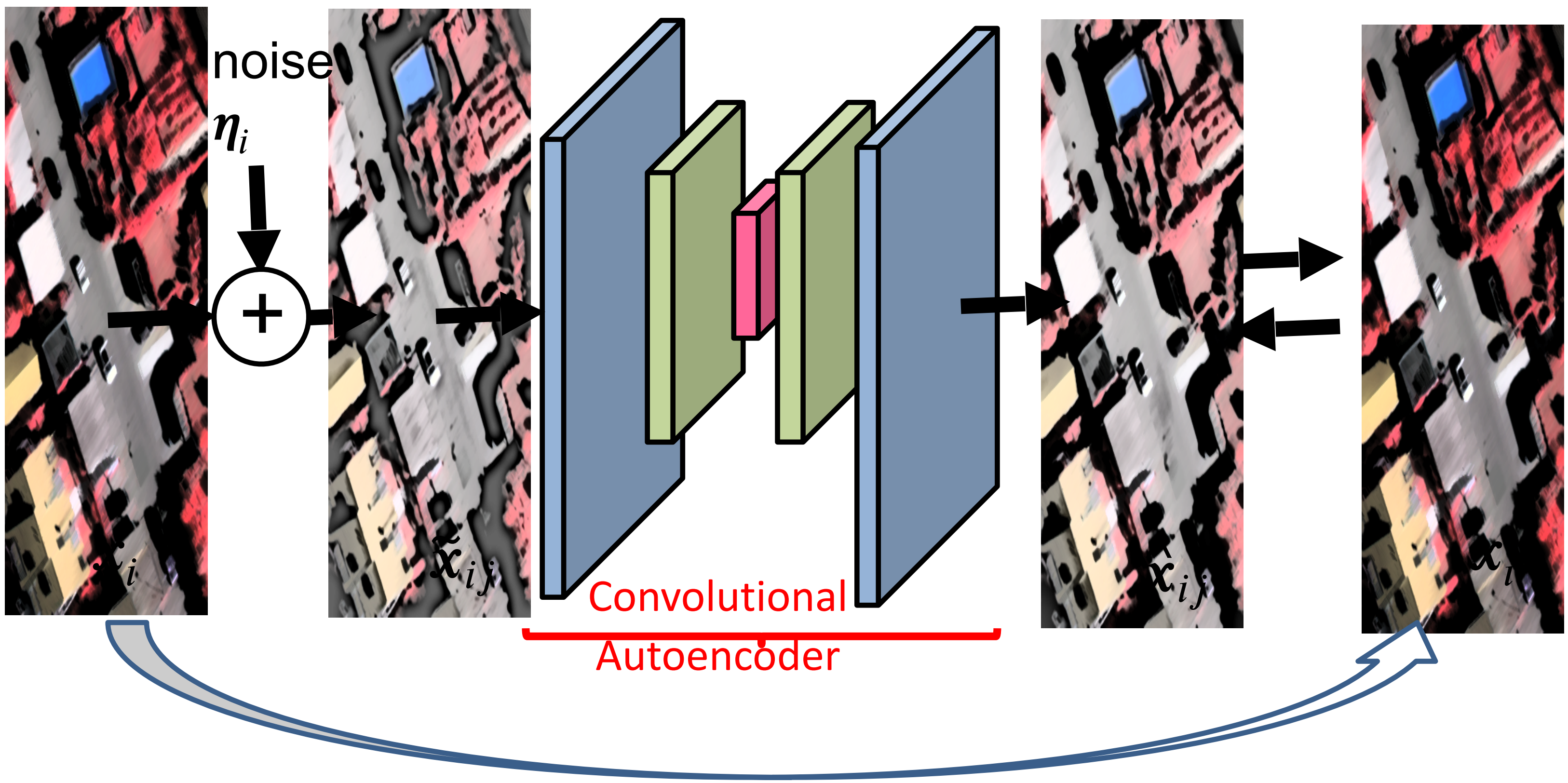
Convolutional autoencoders

Denoising autoencoders

Variational autoencoders

Denoising Autoencoders

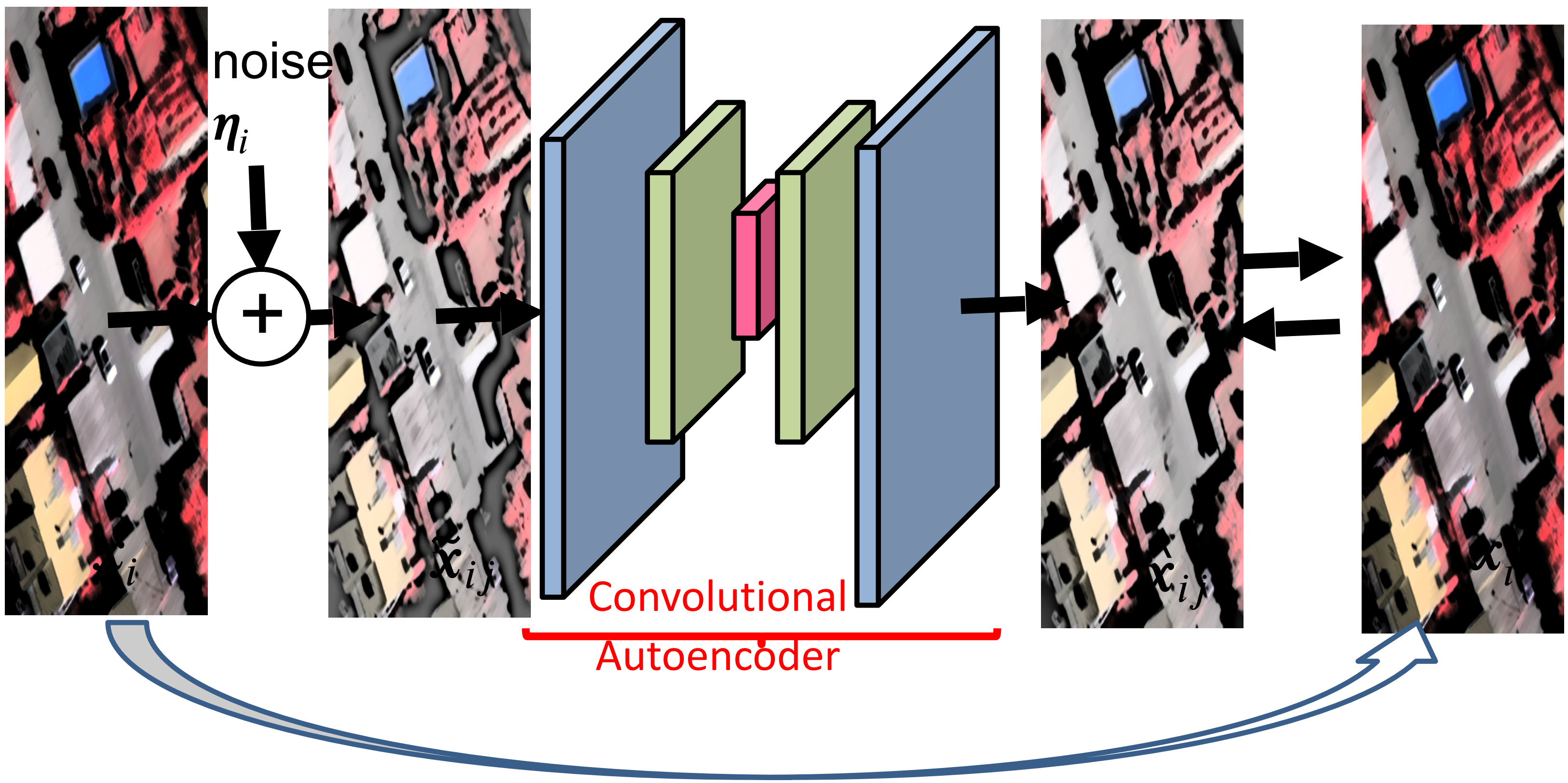
Add noise to a clean image $\boldsymbol{x}_i$ to create noisy versions ($\tilde{\boldsymbol{x}}_{ij}$) of it.



Alain, Hinton, 2013). What regularized auto-encoders learn from the data generating distribute. ICLR~2013

Denoising Autoencoders

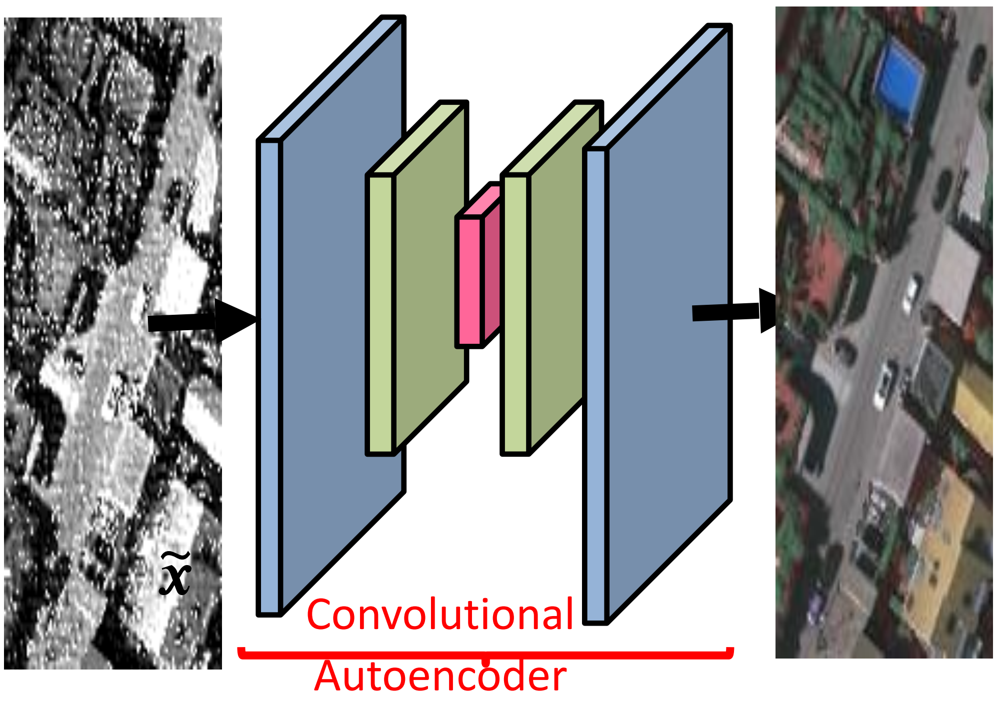
Train the autoencoder with the clean/noisy pairs ($\boldsymbol{x}_i/\tilde{\boldsymbol{x}}_{ij}$) to reconstruct the clean image from a noisy input.



Alain, Hinton, 2013). What regularized auto-encoders learn from the data generating distribute. ICLR~2013

Denoising Autoencoders

Apply the trained autoenconder to denoise any image.



Alain, Hinton, 2013). What regularized auto-encoders learn from the data generating distribute. ICLR~2013

Learning Geometric Features for Improving the Automatic Detection of Citrus Plantation Rows in UAV Images

Laura Elena Cué La Rosa, *Graduate Student Member, IEEE*, Dário A. B. Oliveira, *Associate Member, IEEE*, Maciel Zortea , *Member, IEEE*, Bruno Holtz Gemignani, and Raul Queiroz Feitosa, *Senior Member, IEEE*

***Abstract*— Unmanned aerial vehicles (UAVs) allow on-demand imaging of orchards at an unprecedented level of detail. The automated detection of plantation rows in the images helps in the successive analysis steps, such as the detection of individual fruit trees and planting gaps, aiding producers with inventory and planting operations. Citrus trees can be planted in curved rows that form intricate geometric patterns in aerial images, requiring robust detection approaches. While deep learning methods rank among state-of-the-art methods for segmenting images with particular geometrical patterns, they struggle to hold their performance when testing data differs much from training data (e.g., image intensity differences, image artifacts, vegetation characteristics, and landscape conditions). In this letter, we propose a method to learn geometric features of orchards in UAV images and use them to improve the detection of plantation rows. First, we train a detection encoder–decoder network (DetED) to segment planting rows in RGB images. Then, with labeled data, we train an encoder–decoder correction network (CorrED) that learns to map binary masks with spurious row segmentation geometries into corrected ones. Finally, we use the CorrED network to fix geometric inconsistencies in DetED outcome. Our experiments with commercial plantations of orange trees show that the proposed CorrED postprocessing can restore missing segments of plantation rows and improve detection accuracy in testing data.**

***Index Terms*— Encoder–decoder networks, geometric patterns, plantation rows detection, postprocessing.**

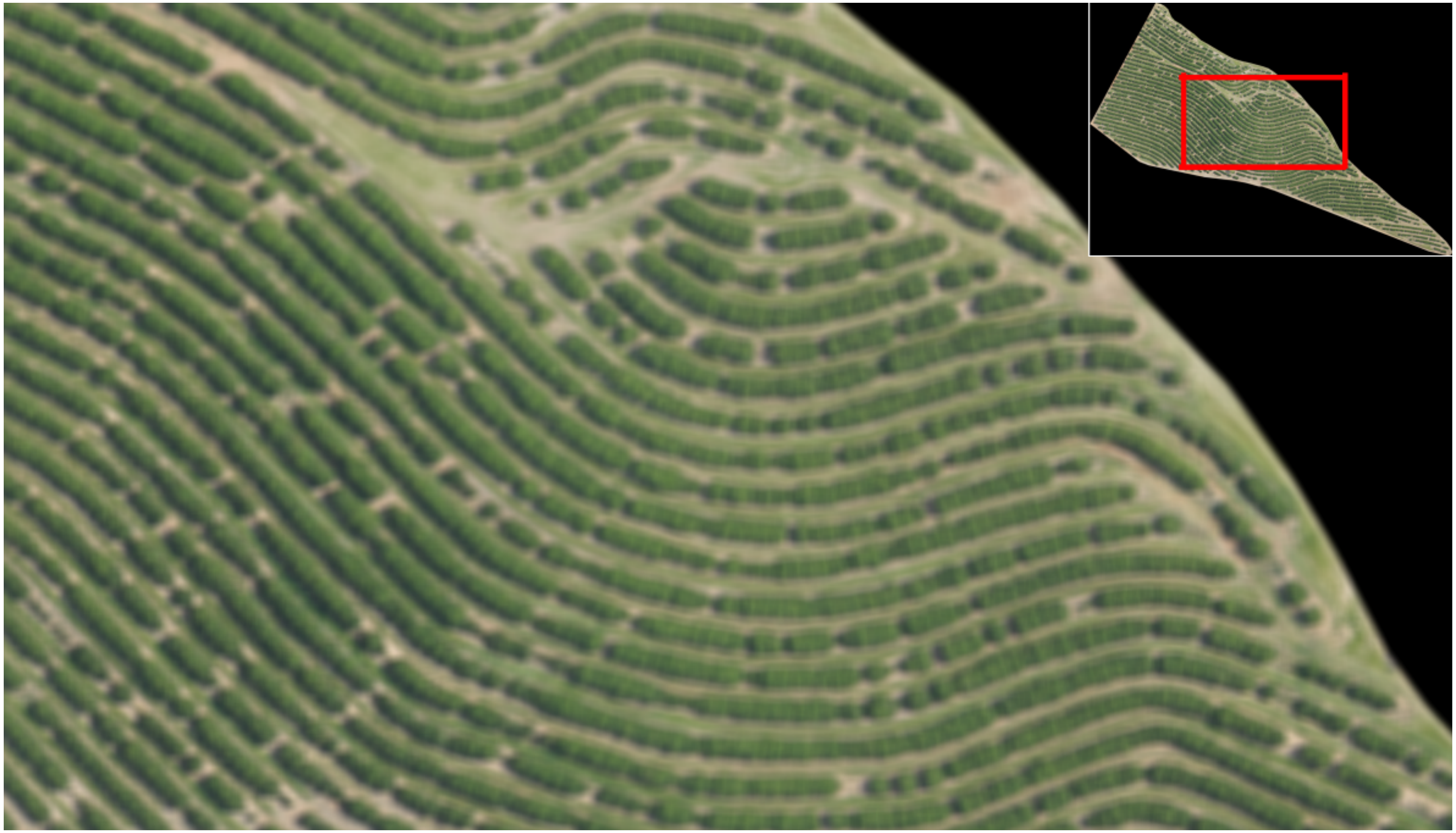
Accurately locating trees is a critical step in the automatic monitoring of orchards and farms, supporting related activities such as counting plants, estimating the yield, providing automatic vehicle guidance, and upholding site-specific tasks. Several methods aims at addressing this problem, with applications in the detection of planting failures in crops such as coffee [1], tomato [2], and location of vine and coffee rows [3], [4]. However, most of these approaches rely on traditional digital image processing techniques such as morphological operators, Hough transform (HT), crop row orientation, and region growing segmentation, that not rarely fail to cope with different crop complexity, shadows, background information, sensor characteristics, planting method, and region conditions.

More recently, deep learning (DL) approaches became very popular in remote sensing, frequently ranking among state-of-the-art techniques in many applications [5]. Regarding UAV image analysis, several studies highlight the potential of convolutional neural networks (CNNs), particularly in the detection of planting lines [6]–[8]. Notwithstanding the reasonable results reported in the literature for detecting the plantation rows, most of the proposed DL methods struggle to embed the somewhat complex geometry observed in orchards correctly. Citrus trees are planted in regular rows, with a predefined

La Rosa, et al., 2020: Learning Geometric Features for Improving the Automatic Detection of Citrus Plantation Rows in UAV Images, 2020, IEEE-GSRL

Denoising Autoencoders: Example

Detection of Citrus Plantation Rows in UAV Images



input image

[La Rosa, et al., 2020: Learning Geometric Features for Improving the Automatic Detection of Citrus Plantation Rows in UAV Images, 2020, IEEE-GSRL]

Denoising Autoencoders: Example

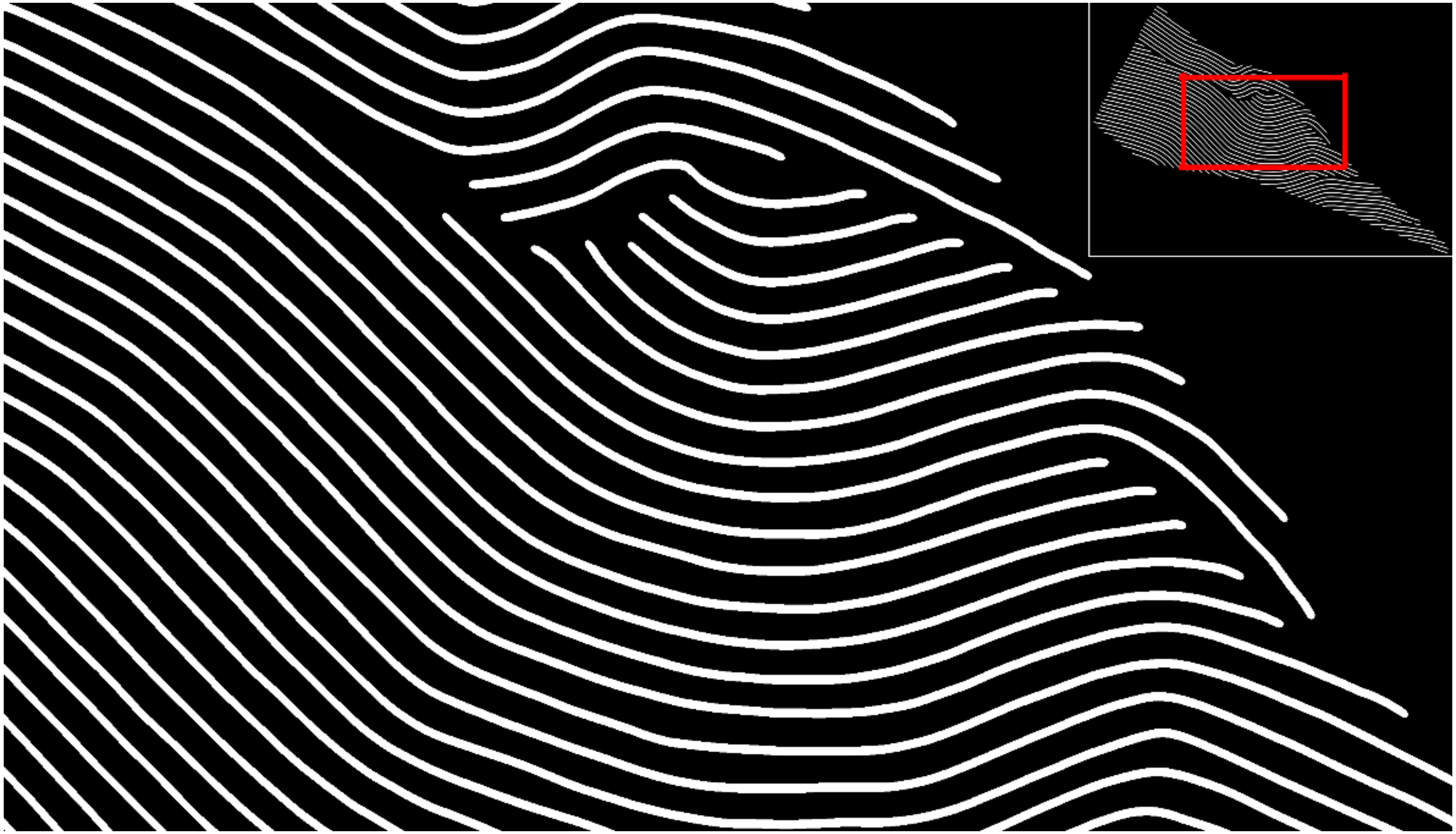
Detection of Citrus Plantation Rows in UAV Images

result o row detection algorithm **before** denoising

La Rosa, et al., 2020: Learning Geometric Features for Improving the Automatic Detection of Citrus Plantation Rows in UAV Images, 2020, IEEE-GSRL

Denoising Autoencoders: Example

Detection of Citrus Plantation Rows in UAV Images

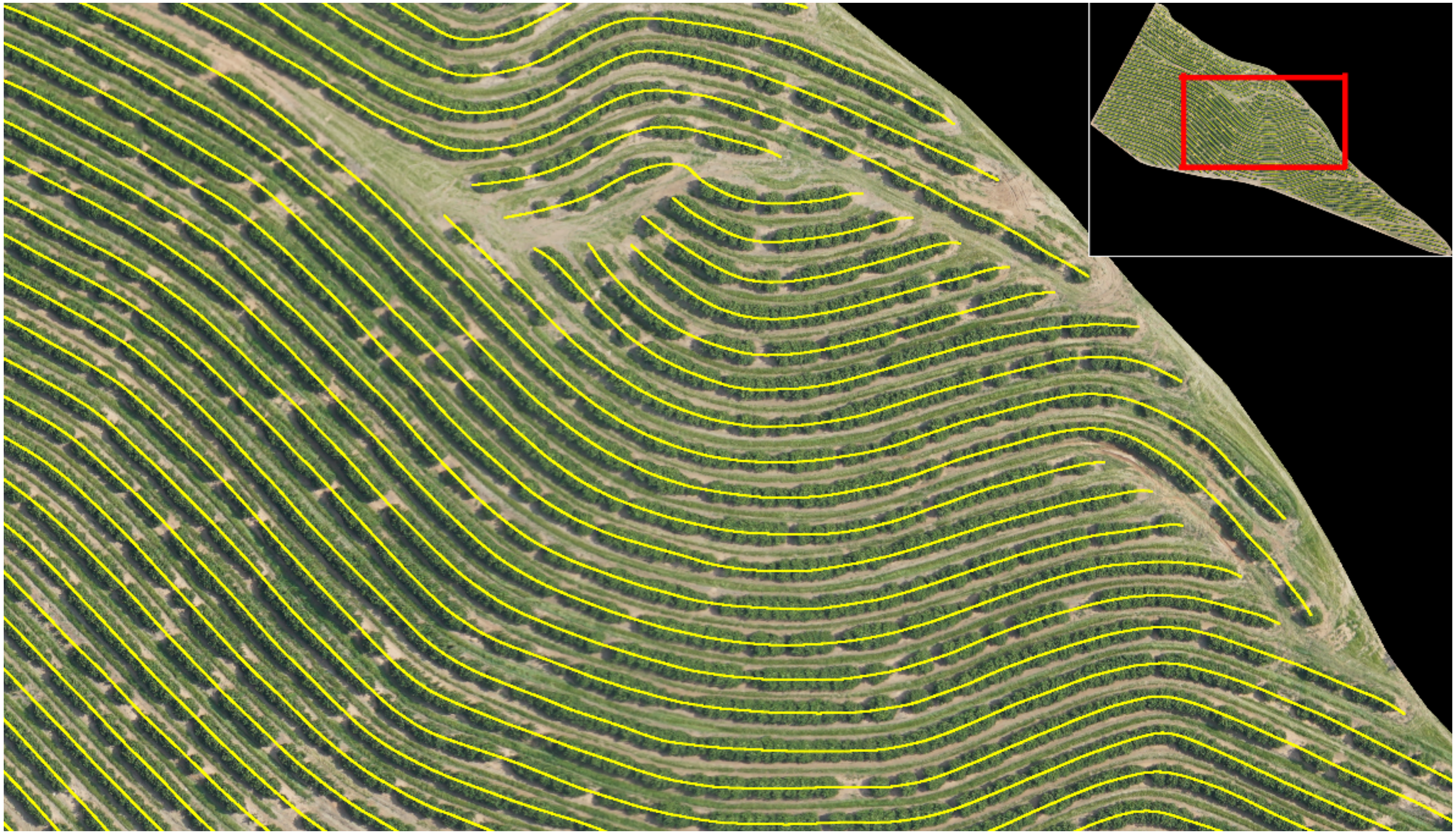


result o row detection algorithm **after** denoising

[La Rosa, et al., 2020: Learning Geometric Features for Improving the Automatic Detection of Citrus Plantation Rows in UAV Images, 2020, IEEE-GSRL]

Denoising Autoencoders: Example

Detection of Citrus Plantation Rows in UAV Images



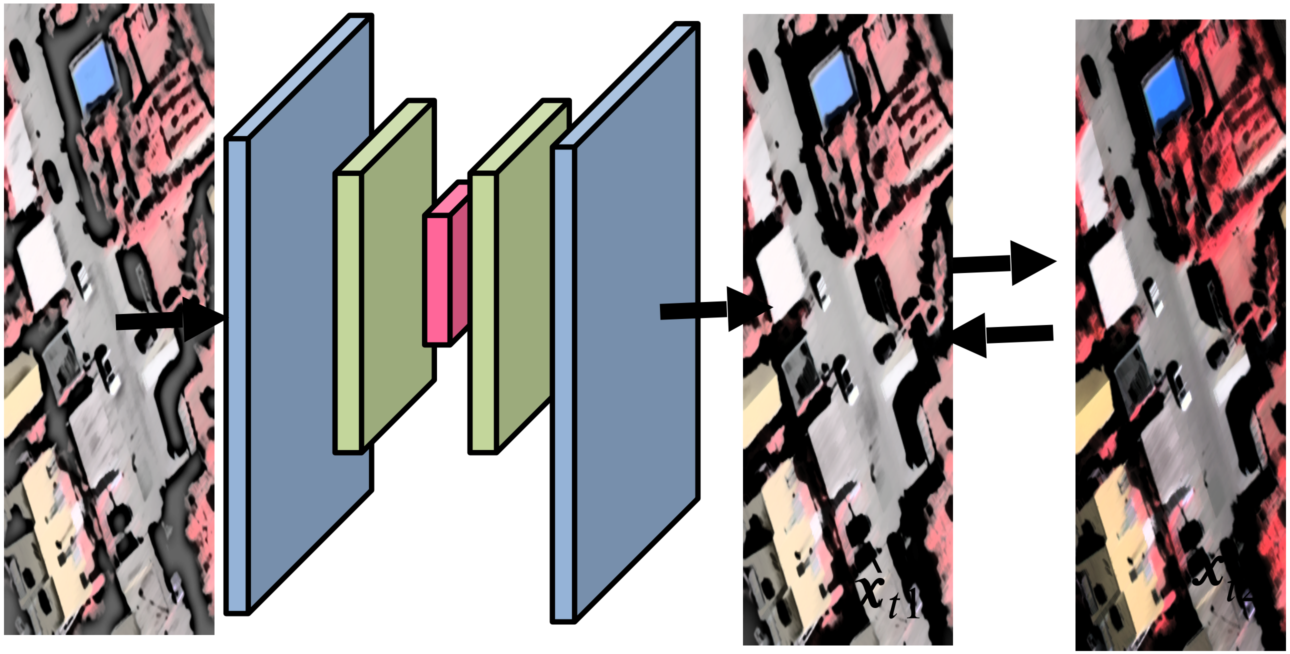
rows over the input image

[La Rosa, et al., 2020: Learning Geometric Features for Improving the Automatic Detection of Citrus Plantation Rows in UAV Images, 2020, IEEE-GSRL]

Speckel suppression in SAR Images

Problem: no clean image version for training the denoising AE

Hypothesis: no change between two consecutive dates.

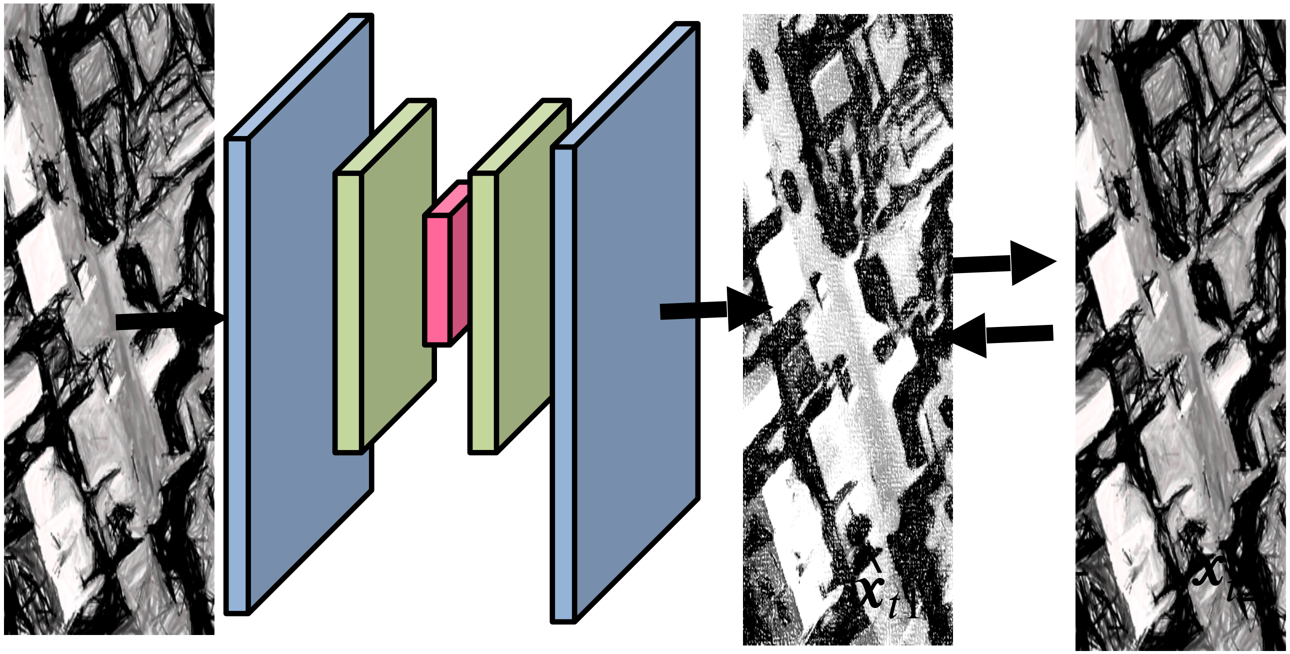


From : Boulch, et al., 2018. [Learning Speckle Suppression In SAR Images Without Ground Truth: Application To Sentinel-1 Time-series. IGARSS, 2018]

Speckel suppression in SAR Images

Solution:

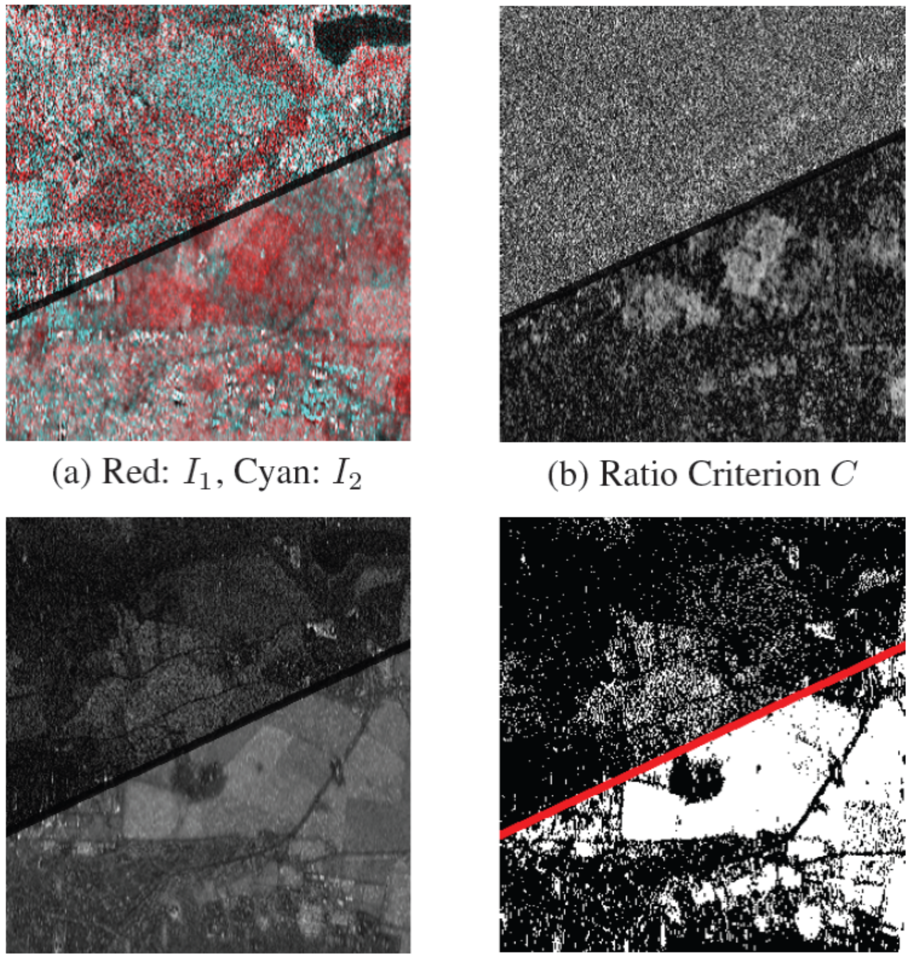
- Train a denoising AE with pairs of registered image patches of close dates, e.g., $\boldsymbol{x}_{t1}$ to produce $\boldsymbol{x}_{t2}$, and vice versa.
- Apply the trained autoencoder to overlapping patches of any date and build a mosaic with the central part of reconstructed patches.



From : Boulch, et al., 2018. [Learning Speckle Suppression In SAR Images Without Ground Truth: Application To Sentinel-1 Time-series. IGARSS, 2018]()

Speckel suppression in SAR Images

Sample Result:



(a) Red: I_1, Cyan: I_2

(b) Ratio Criterion C

(c) γ

(d) Change map.

From : Boulch, et al., 2018. [Learning Speckle Suppression In SAR Images Without Ground Truth: Application To Sentinel-1 Time-series. IGARSS, 2018]

Overview

Introduction

Autoencoders

Convolutional autoencoders

Denoising autoencoders

Variational autoencoders

Recall GAN

- The GAN generator G maps samples $(\boldsymbol{z})$ drawn from a known distribution $q(\boldsymbol{z})$ to samples of an arbitrarilly complex target distribution $p(\boldsymbol{x})$.

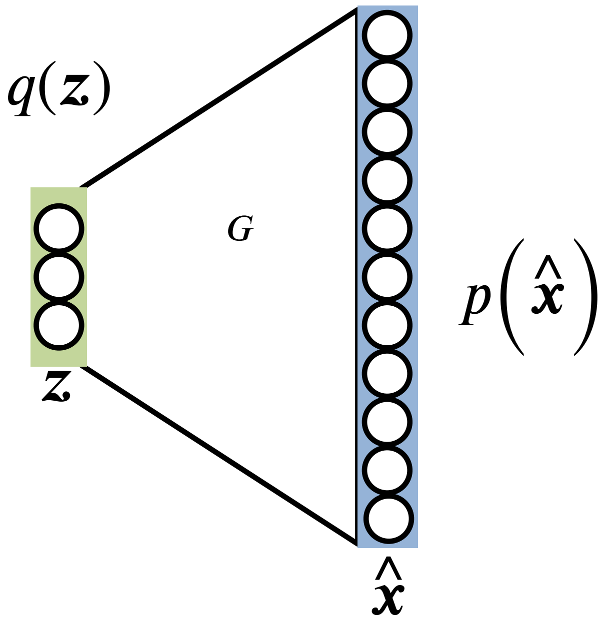
known

$$G: Z \rightarrow X$$

- The GAN doesn't tell which $\boldsymbol{z}$ value will generate a *desired* output (image) $\boldsymbol{x}$.

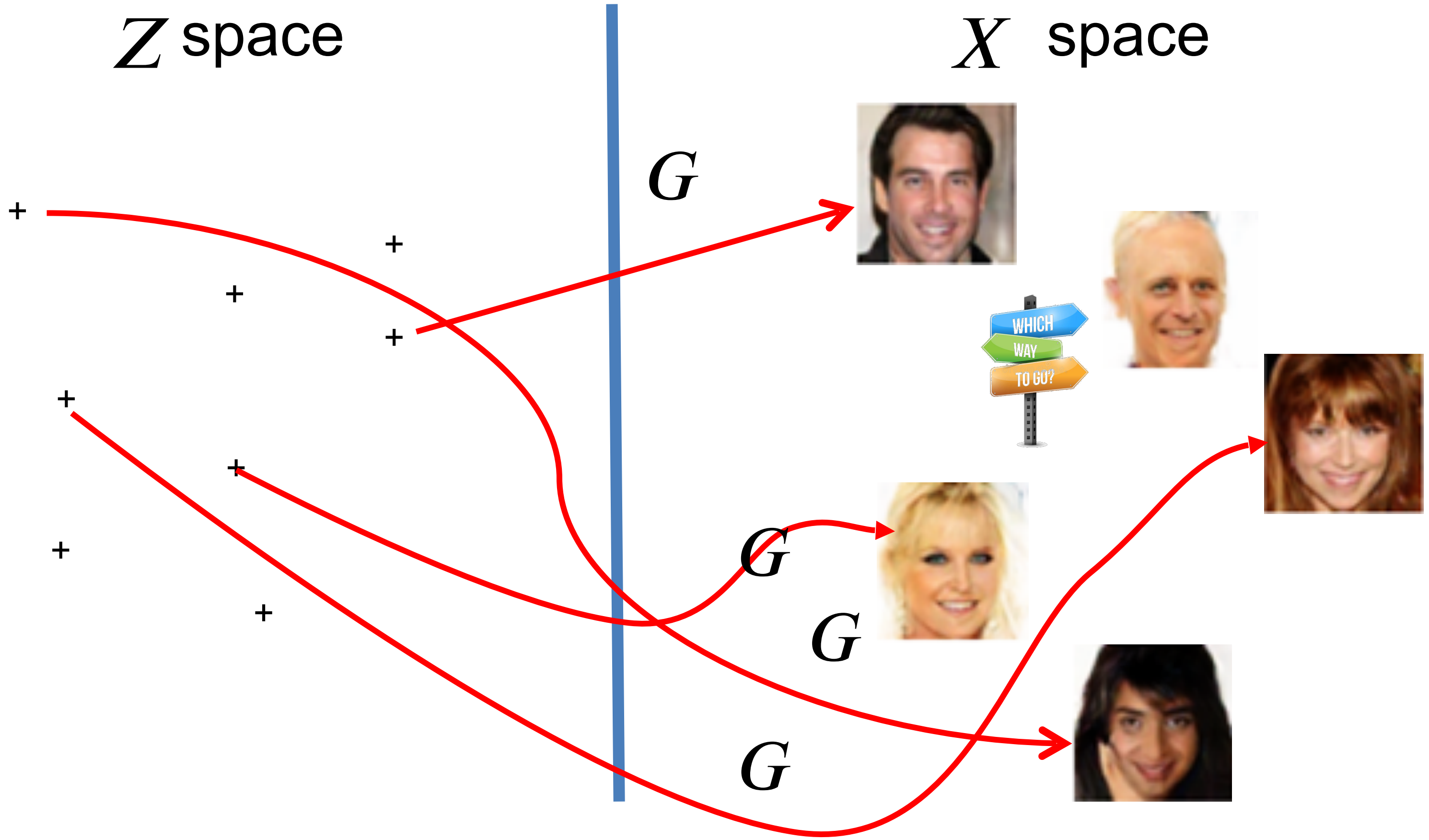
unknown

$$G^{-1}: X \rightarrow Z$$



Since we don't have $G^{-1}: X \to Z$

... we can't predict what $\boldsymbol{z}$ correspond to a given $\boldsymbol{x}$



If we had $G^{-1}: X \to Z$

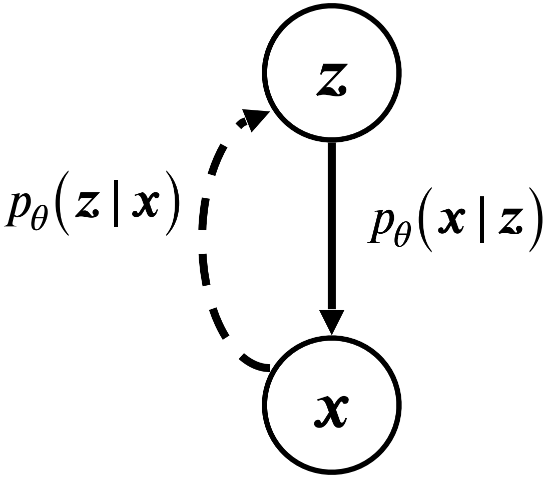
we could tune z to change characteristic of the output x

Source: Larsen et al. 2016 [Autoencoding beyond pixels using a learned similarity metric]

Graphical models

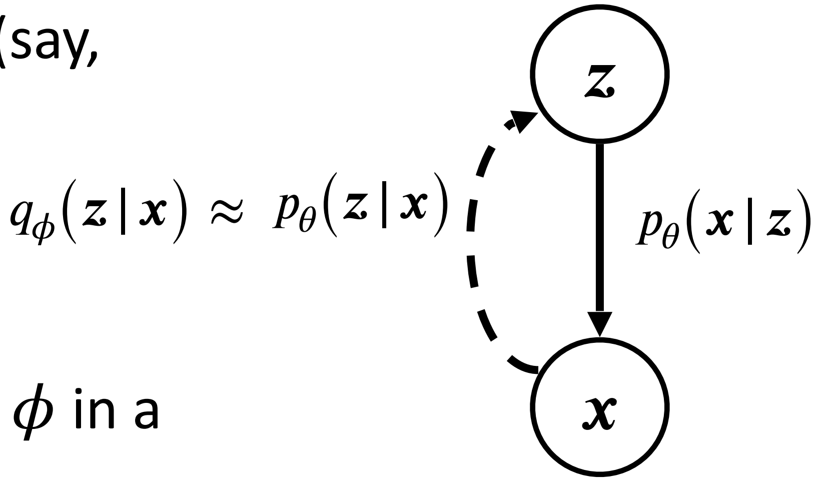
- Let's assume that the distribution of an image $\boldsymbol{x}$ is determined by a latent (non observable) variables $\boldsymbol{z}$.
- Example: for face image, latent variables may determine :

  pose, expression, hair stile, gender, etc.
- Some methods allow estimating $p_\theta(\boldsymbol{x} \mid \boldsymbol{z})$.
- Estimating $p_\theta(\boldsymbol{z} \mid \boldsymbol{x})$ is, however intractable →unknown.



VAE approach

- $p_\theta(\boldsymbol{z} \mid \boldsymbol{x})$ is usually complex and thus difficult to sample from
- approximate $p_\theta(\boldsymbol{z} \mid \boldsymbol{x})$ by another tractable distribution $q_\phi(\boldsymbol{z} \mid \boldsymbol{x})$ (say, Gaussian), and

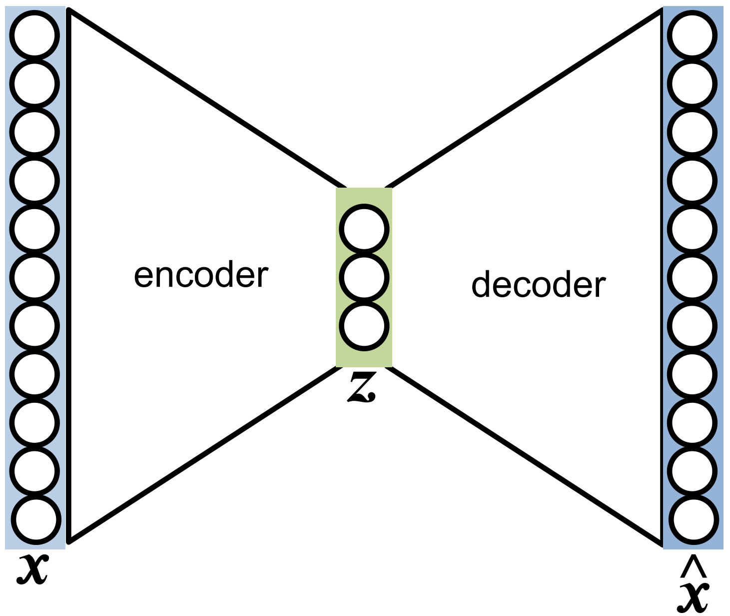


- play around with its parameters ϕ in a way that $q_\phi(\boldsymbol{z} \mid \boldsymbol{x})$ gets close enough to a collection of samples from $p_\theta(\boldsymbol{z} \mid \boldsymbol{x})$.

Recall the traditional AE

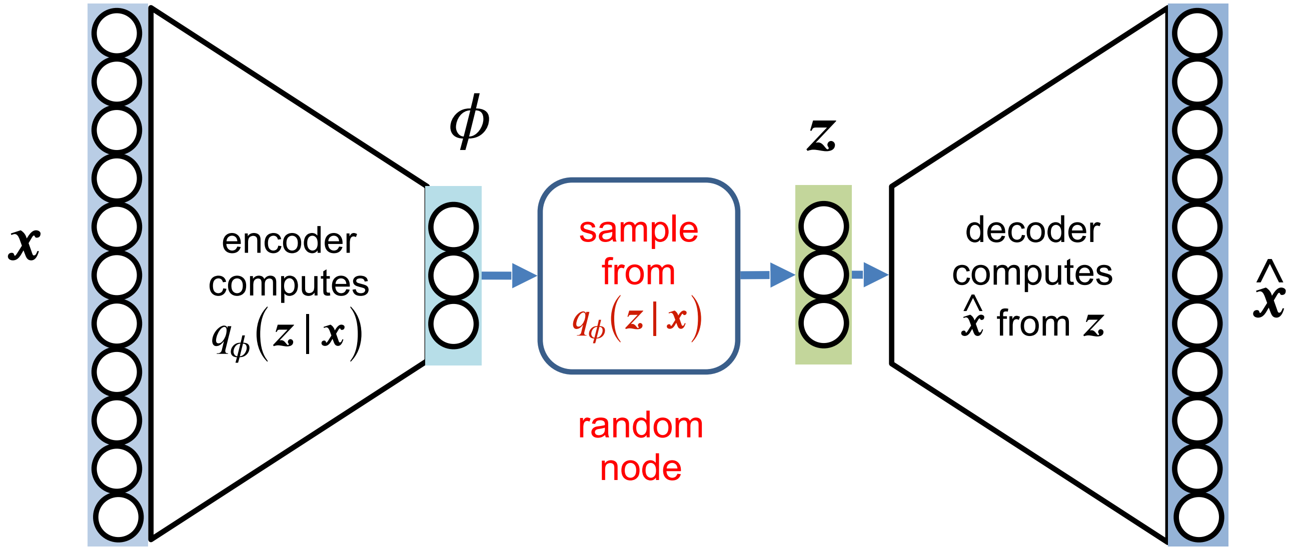
Is a deterministic mapping.

given an input $\boldsymbol{x}$ it will always produce the same output $\hat{\boldsymbol{x}}$



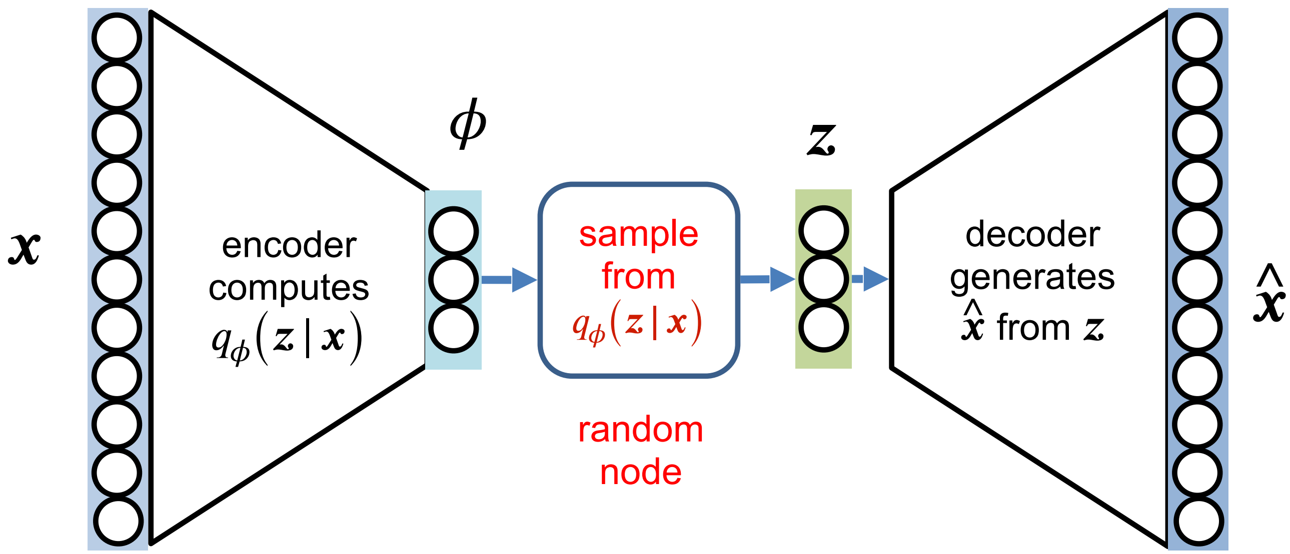
VAEs: key difference with AE

1) Instead of producing single $\boldsymbol{z}$, from an input $\boldsymbol{x}$, the encoder generates a distribution $q_\phi(\boldsymbol{z} \mid \boldsymbol{x})$ of the $\boldsymbol{z}$ values that correspond to the input $\boldsymbol{x}$.



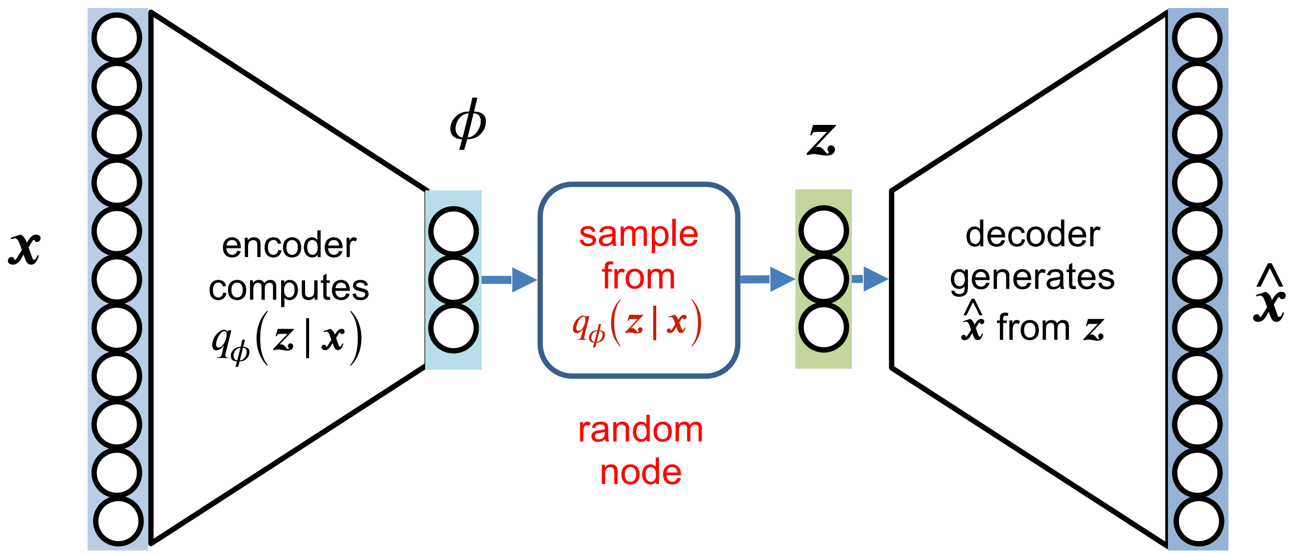
VAEs: key difference with AE

2) then the VAE samples a $\boldsymbol{z}$ from $q_\phi(\boldsymbol{z}\,|\,\boldsymbol{x})$



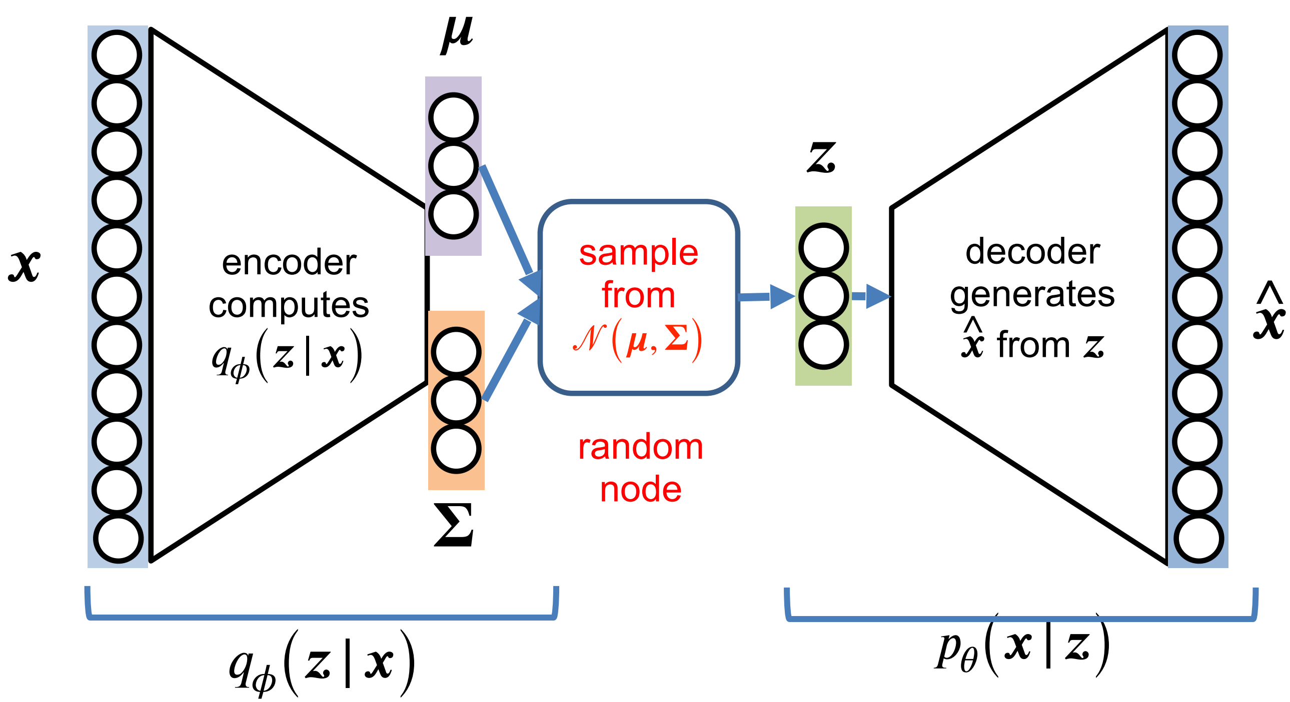
VAEs: key difference with AE

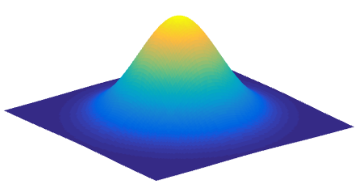
3) finally, the decoder generates the output $\hat{\boldsymbol{x}}$ from a sample $\boldsymbol{z}$.



Defining the latent variable distribution

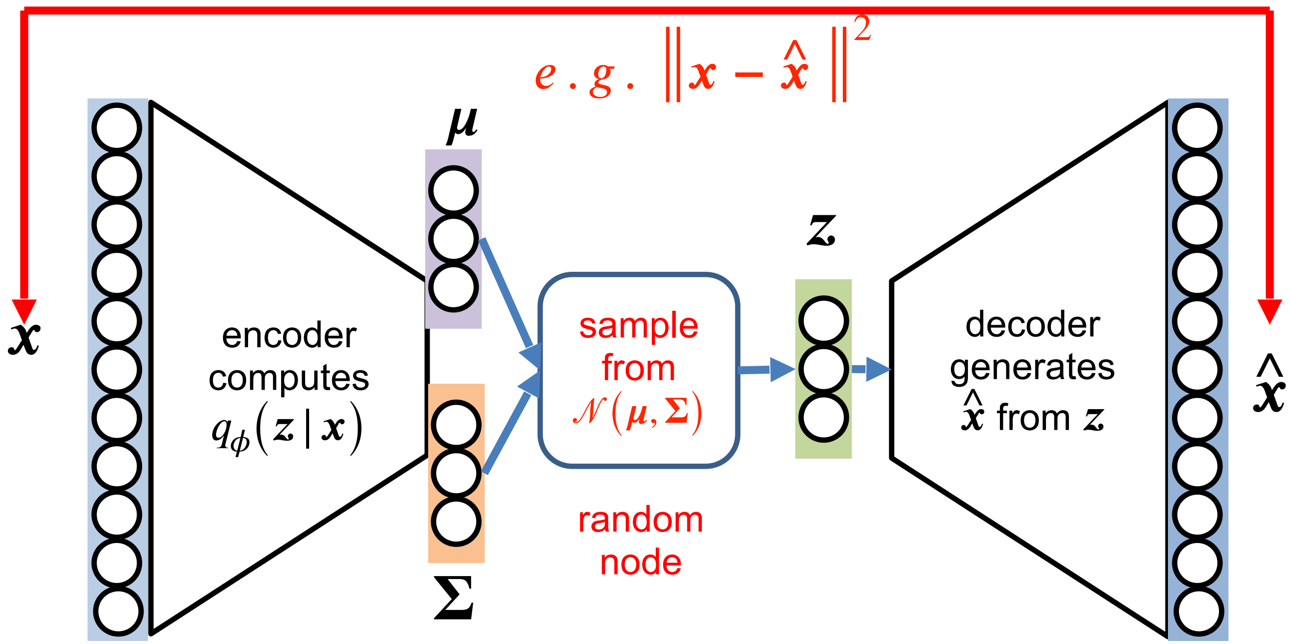
Recall that $q_\phi(\boldsymbol{z}\,|\,\boldsymbol{x})$ must be tractable to enable sampling from it. So, we choose q_ϕ to be a Gaussian with mean vector $\boldsymbol{\mu}$ and the standard deviation vector $\boldsymbol{\Sigma}$





VAE Optimization

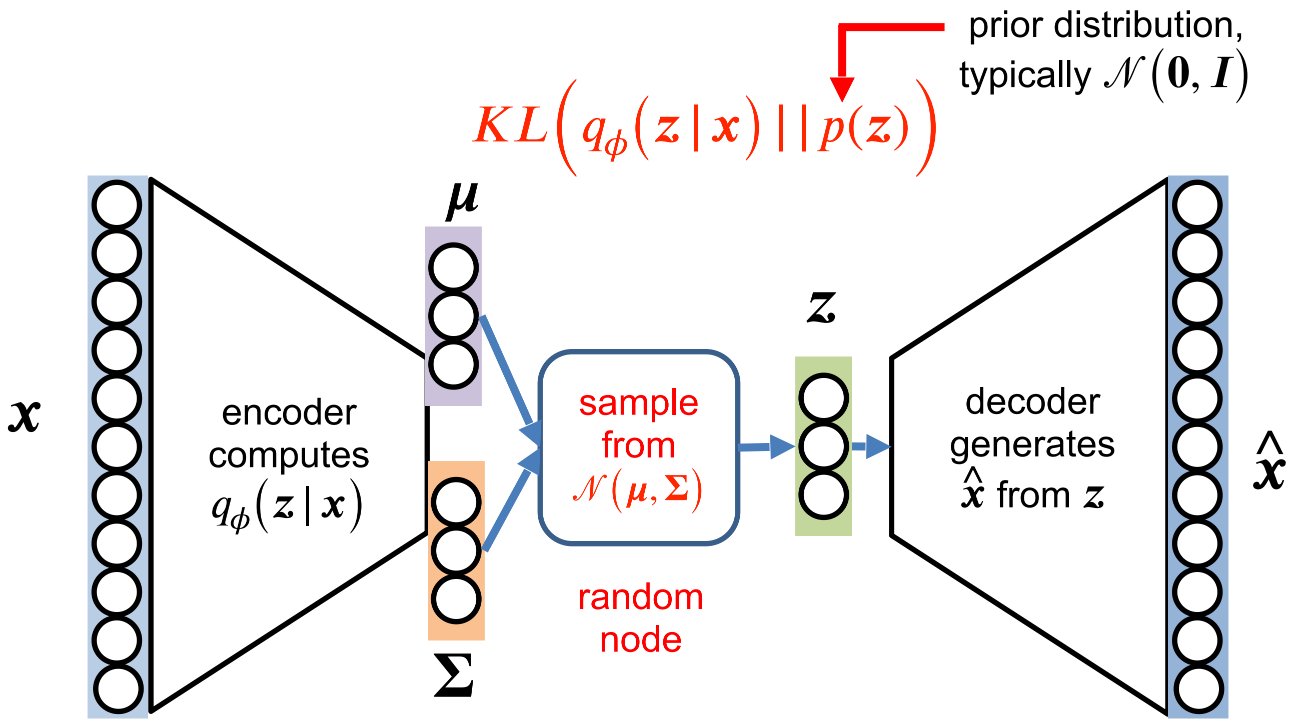
The VAE loss comprises two terms:



$$\mathscr{L}(\phi, \theta, x) = \boxed{(\text{reconstruction loss})} + (\text{regularization term})$$

VAE Optimization

The VAE loss comprises two terms:



$$\mathscr{L}(\phi, \theta, x) = (\text{reconstruction loss}) + \boxed{(\text{regularization term})}$$

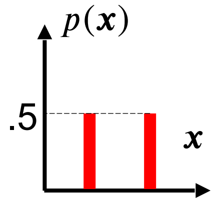
The Kullback Leibler divergence

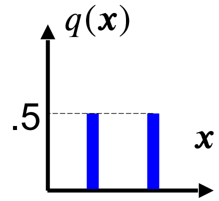
… measures how different are two probability distributions

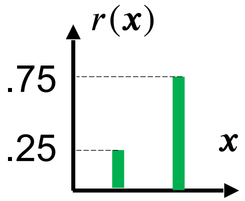
Definition:

$$KL\left(p(\boldsymbol{x})\,||\,q(\boldsymbol{x})\right) = -\sum_{\boldsymbol{x}} p(\boldsymbol{x})\log q(\boldsymbol{x}) + \sum_{\boldsymbol{x}} p(\boldsymbol{x})\log p(\boldsymbol{x}) = \sum_{\boldsymbol{x}} p(\boldsymbol{x})\log\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}$$

Examples:







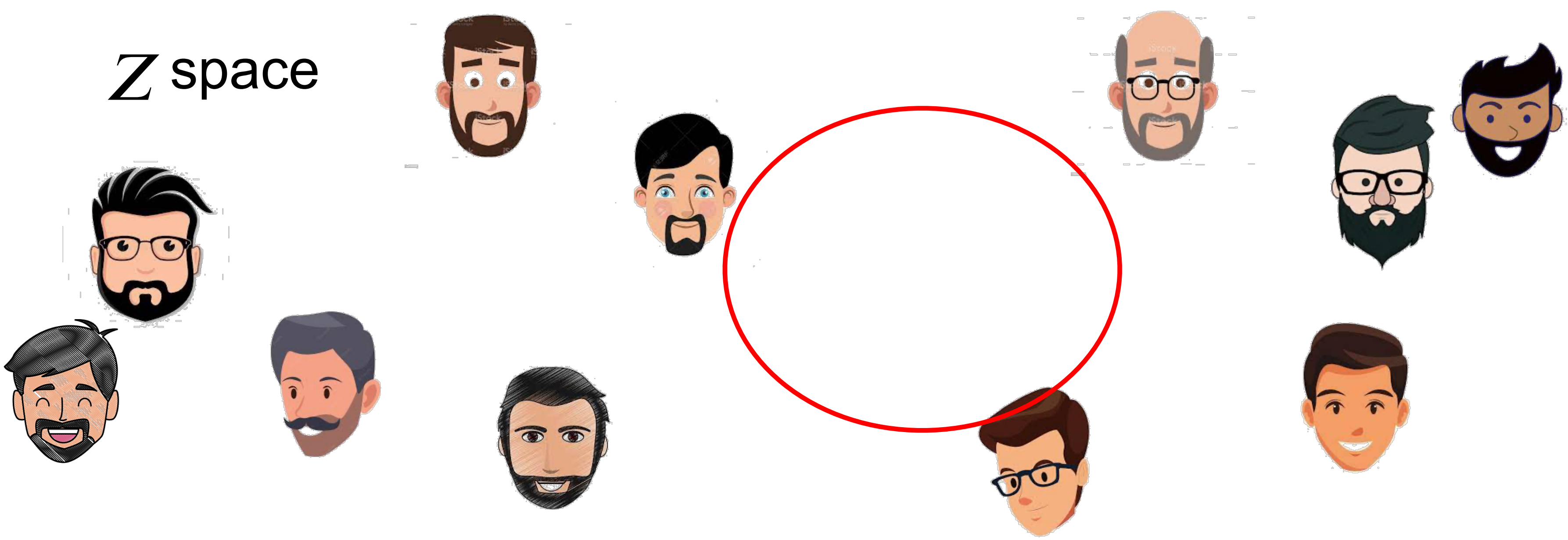
$$KL\left(p(\boldsymbol{x})\,||\,q(\boldsymbol{x})\right) = 0$$

$$KL\left(p(\boldsymbol{x})\,||\,r(\boldsymbol{x})\right) = .5\log\frac{.5}{.25} + .5\log\frac{.5}{.75} = 0.0625$$

Intuition about the regularization

For the generative process work well the latent space should meet two properties:

- Continuity: two close points in the latent space should give two different but similar contents once decoded
- Completeness: a point sample from the latent space should give “meaningful” content once decoded



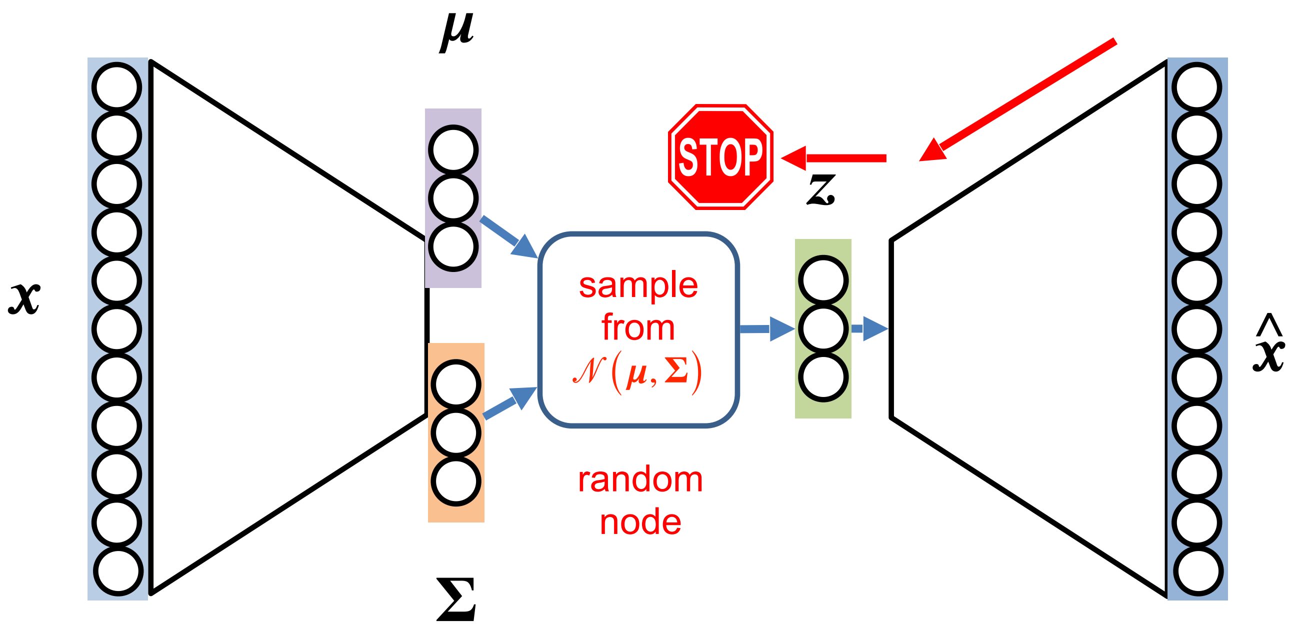
Intuition about the regularization

$$KL\left(q_\phi(\boldsymbol{z}\,|\,\boldsymbol{x})\,||\,\mathcal{N}(\boldsymbol{0},\boldsymbol{I})\right) = -\frac{1}{2}\sum_{j=0}^{k-1}\left(\sigma_j + \mu_j^2 - 1 - \log\sigma_j\right)$$

- Encourages the encoder to distribute encodings evenly around the center of the latent space
- Penalizes the network when it tries to cluster points in specific regions

How to train a VAE?

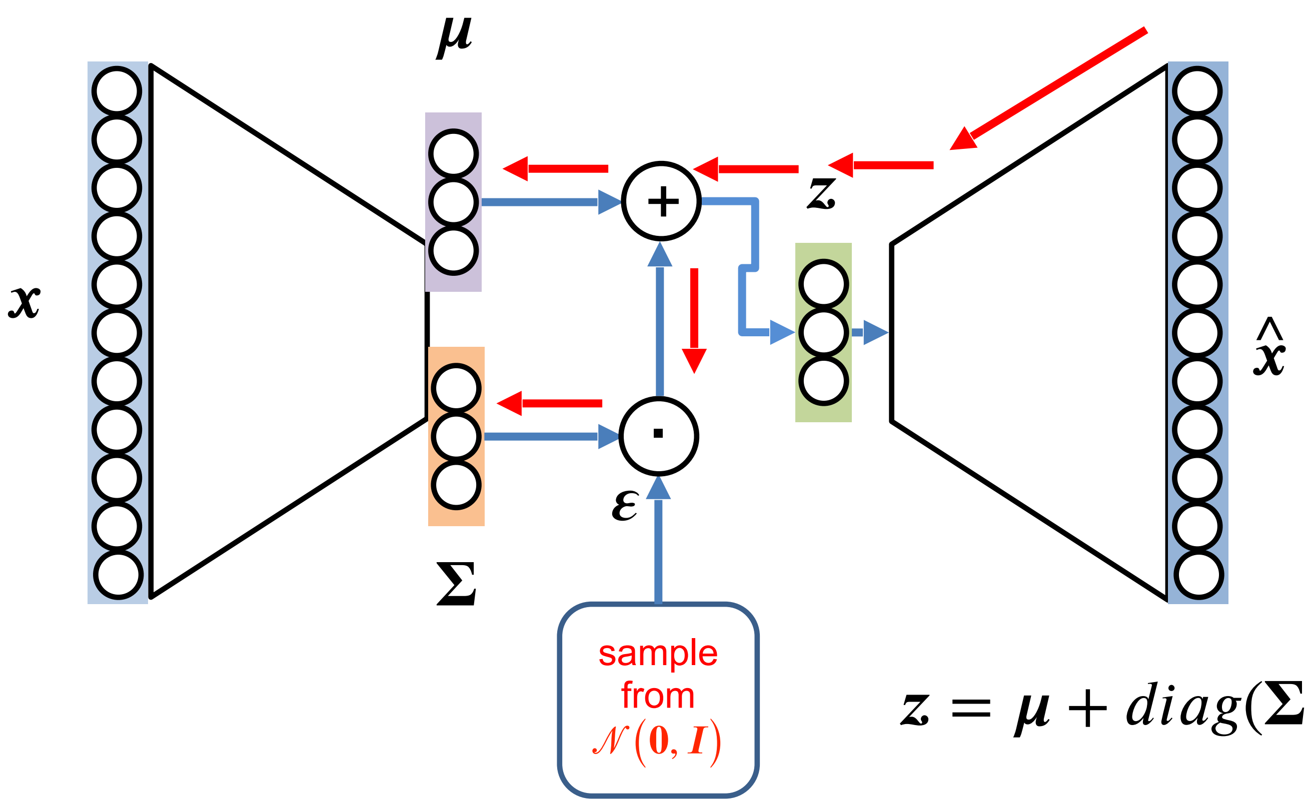
Instead of producing $\boldsymbol{z}$, the encoder generates $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ of the target Gaussian distribution and samples from it.



Problem: Backprop cannot flow through a random node and sample from it

Reparametrization Trick

Introducing a variable $\boldsymbol{\varepsilon}$ allows us to reparameterize $\boldsymbol{z}$ in a way that backprop flows through deterministic nodes.



$$\boldsymbol{z} = \boldsymbol{\mu} + diag(\boldsymbol{\Sigma}) \odot \boldsymbol{\varepsilon}$$

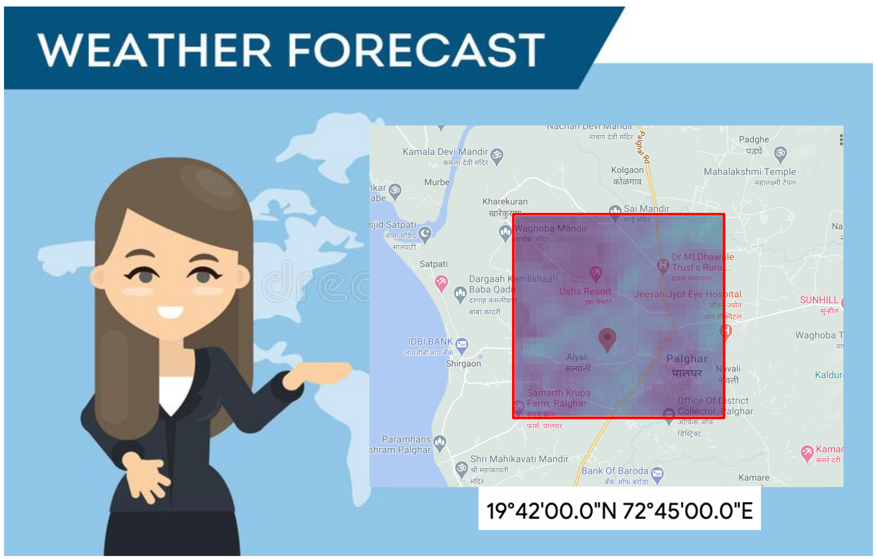
input image

See Tutorial at https://arxiv.org/pdf/1606.05908.pdf

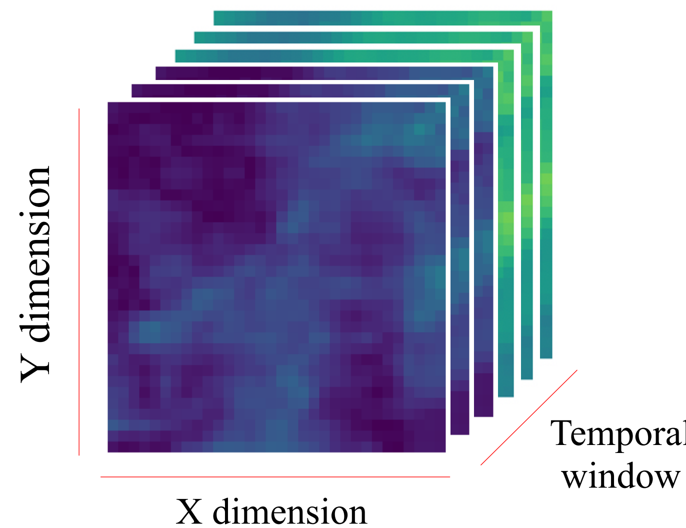
Variational Autoencoders

Controllable Weather Generators using Distribution Priors

Weather fields are the multidimensional representations of spatially distributed weather variables, like temperature, wind, precipitation, etc.



Weather Field Sample

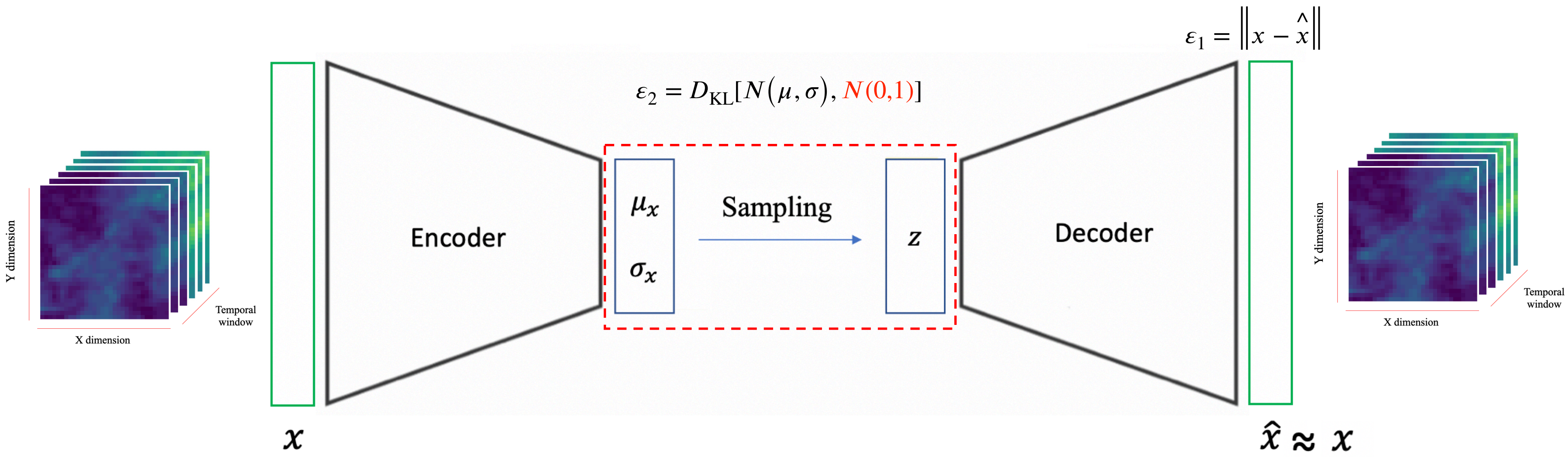


"Controlling Weather Field Synthesis Using Variational Autoencoders", Oliveira et al, ICML'21 Climate AI Workshop

Variational Autoencoders

Controllable Weather Generators using Distribution Priors

Variational Autoencoders

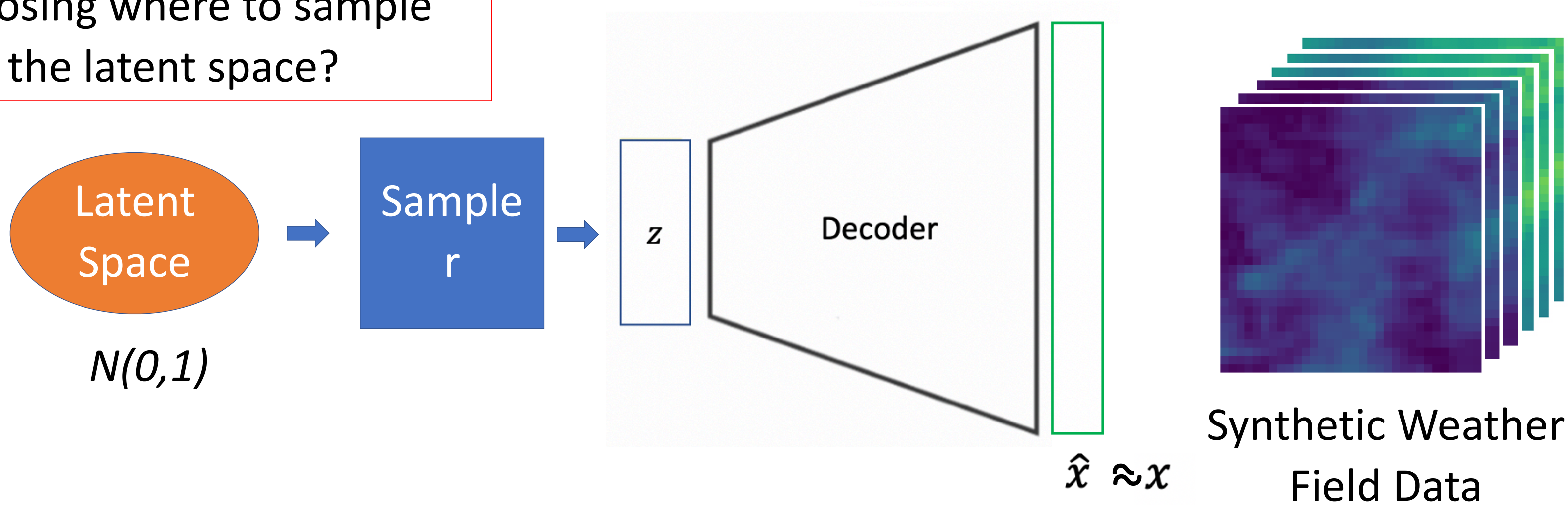


"Controlling Weather Field Synthesis Using Variational Autoencoders", Oliveira et al, ICML'21 Climate AI Workshop

Variational Autoencoders

Controllable Weather Generators using Distribution Priors

Can we control how extreme the synthesized climate data would be by choosing where to sample in the latent space?

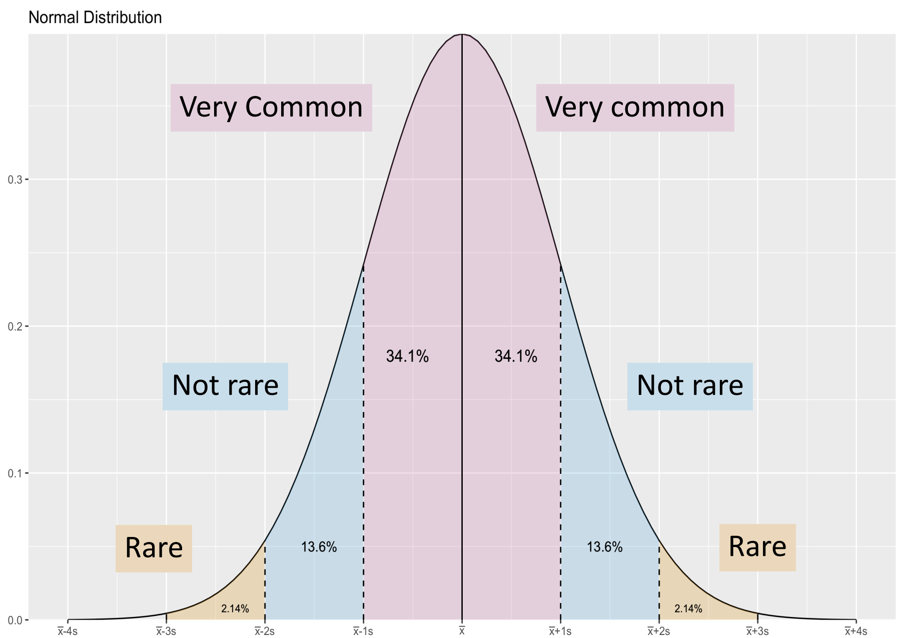


"Controlling Weather Field Synthesis Using Variational Autoencoders", Oliveira et al, ICML'21 Climate AI Workshop

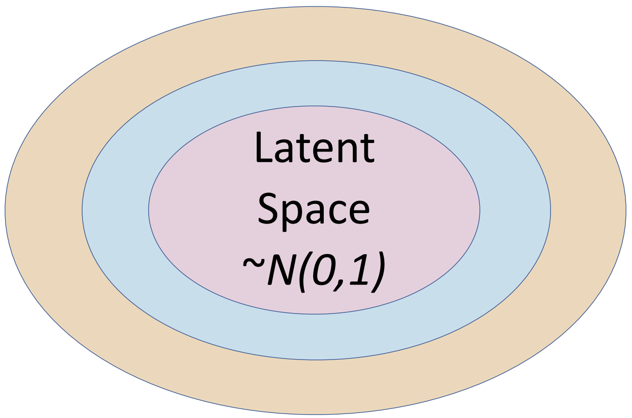
Variational Autoencoders

Controllable Weather Generators using Distribution Priors

How are climate events distributed in the latent space ~ *N(0,1)*?



Very common events would need to be located near the distribution mean and rare events will be located far from it

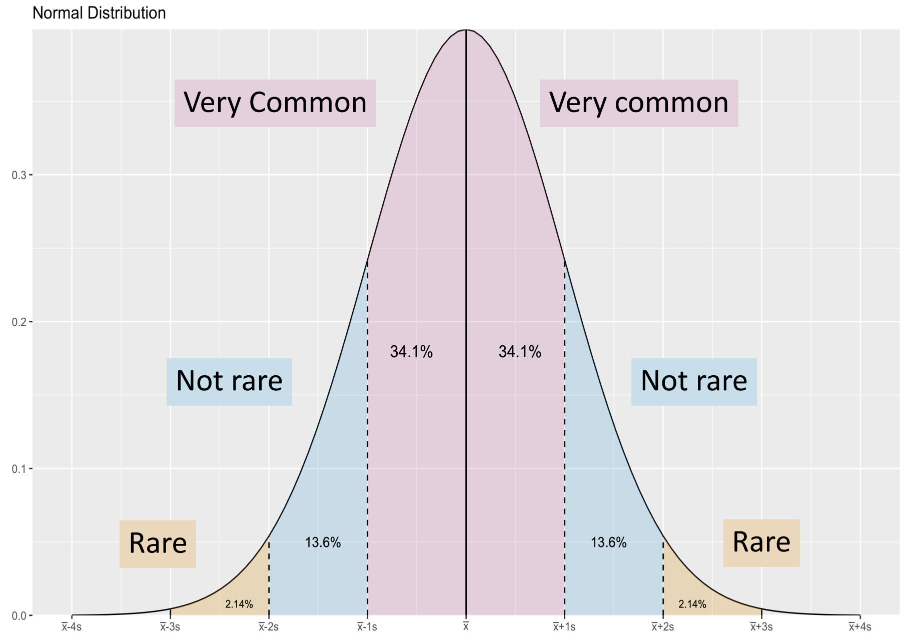


"Controlling Weather Field Synthesis Using Variational Autoencoders", Oliveira et al, ICML'21 Climate AI Workshop

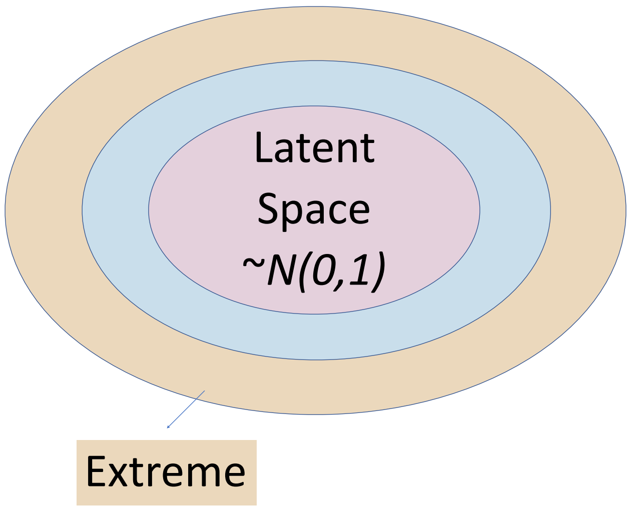
Variational Autoencoders

Controllable Weather Generators using Distribution Priors

How are climate events distributed in the latent space ~ *N(0,1)*?



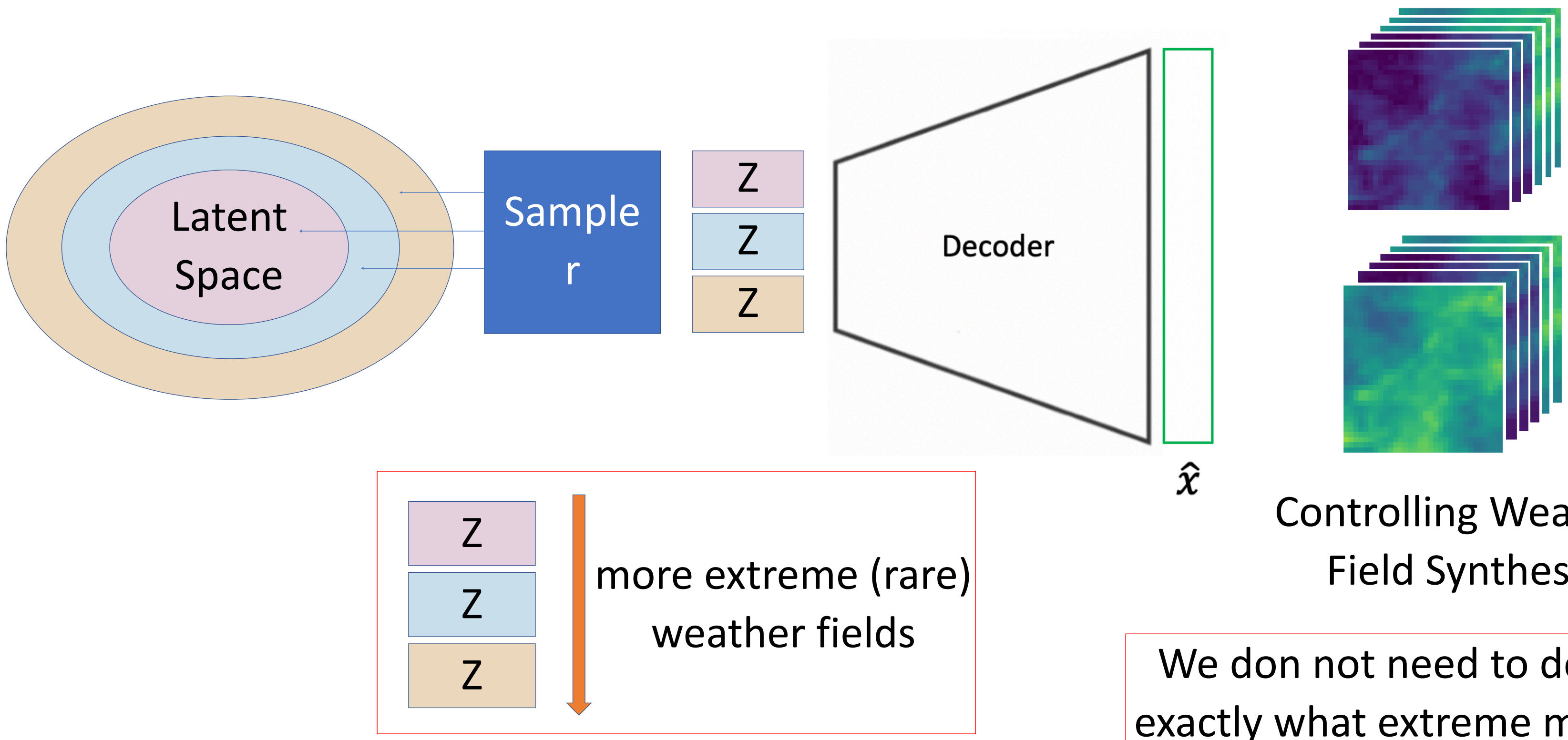
But extreme weather events are also usually rare!



"Controlling Weather Field Synthesis Using Variational Autoencoders", Oliveira et al, ICML'21 Climate AI Workshop

Variational Autoencoders

Controllable Weather Generators using Distribution Priors



Controlling Weather Field Synthesis

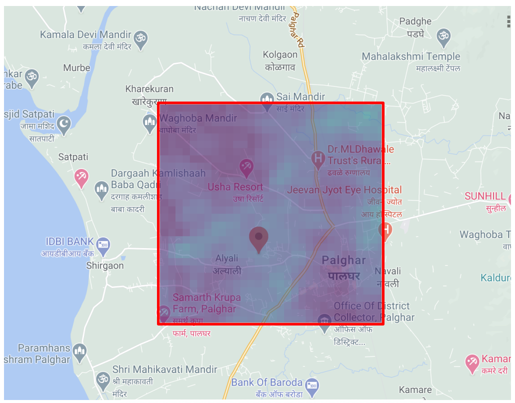
We don not need to define exactly what extreme means!
extreme=rare

"Controlling Weather Field Synthesis Using Variational Autoencoders", Oliveira et al, ICML'21 Climate AI Workshop

Variational Autoencoders

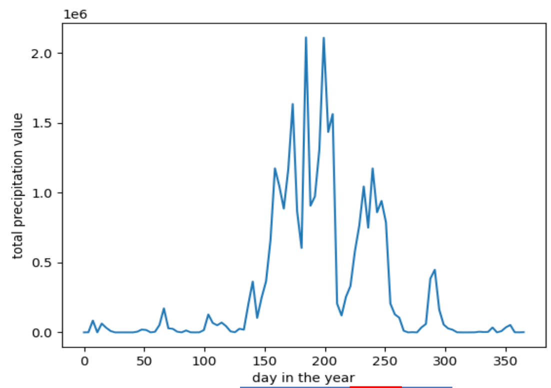
Controllable Weather Generators using Distribution Priors

Monsoons in Palghar,
Southwest India

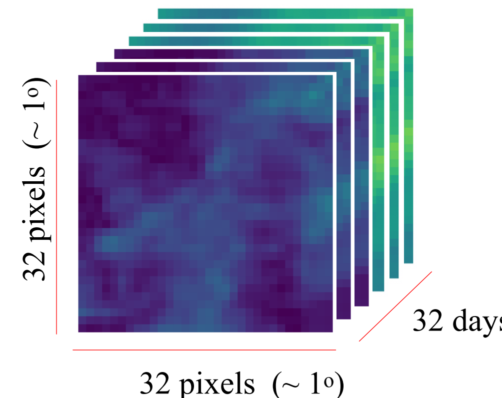


19°42'00.0"N 72°45'00.0"E

Precipitation in a random year



Weather Field Sample



sampling sequences of 32 days

CHIRPS dataset[1] from Palghar, India
39 years of data: 1981-01-01 to 2020-01-01
Training: 1981-2010
Testing: 2010-2020

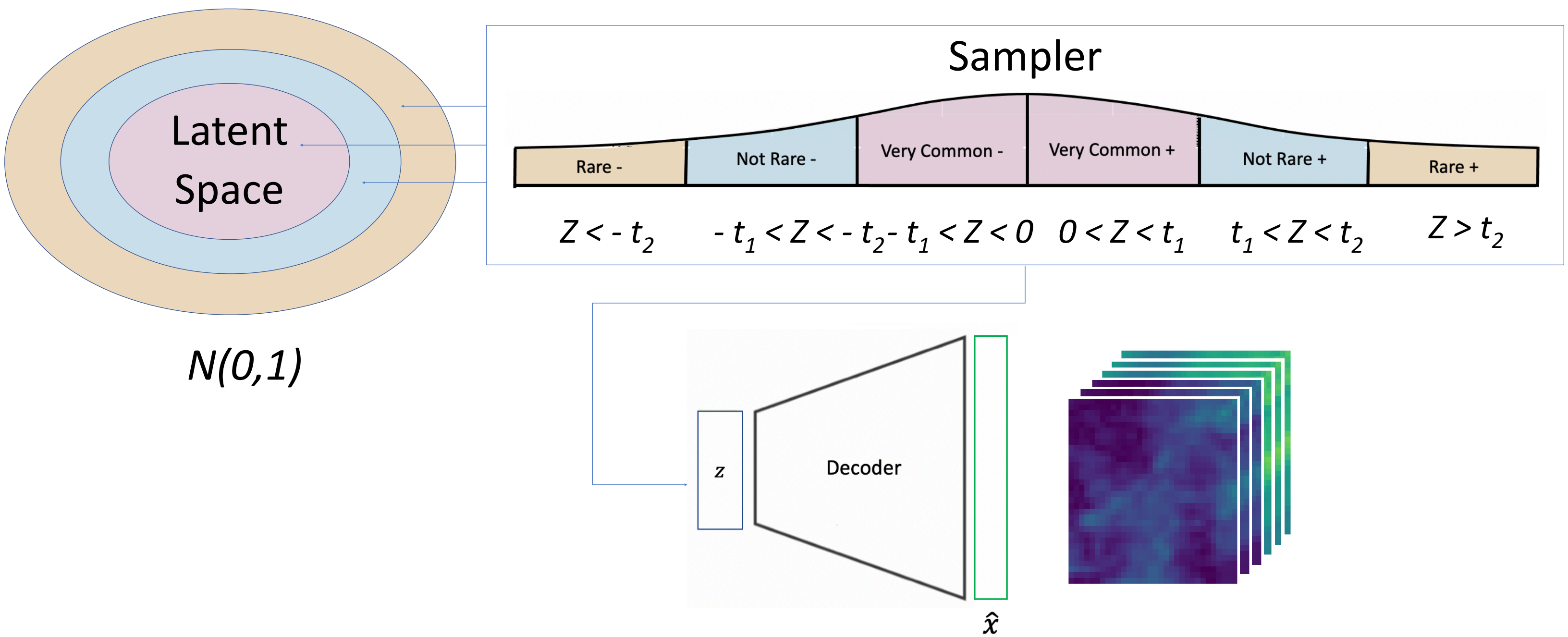
[1] Chris Funk, Pete Peterson, Martin Landsfeld, Diego Pedreros, James Verdin, Shraddhanand Shukla, Gregory Husak, James Rowland, Laura Harrison, Andrew Hoell, et al. The climate hazards infrared precipitation with stations—a new environmental record for monitoring extremes. *Scientific data*, 2(1):1–21, 2015.

"Controlling Weather Field Synthesis Using Variational Autoencoders", Oliveira et al, ICML'21 Climate AI Workshop

Variational Autoencoders

Controllable Weather Generators using Distribution Priors

Experiment: with a trained variational autoencoder, control the sampling of *Z* based on *N(0,1)* quantiles

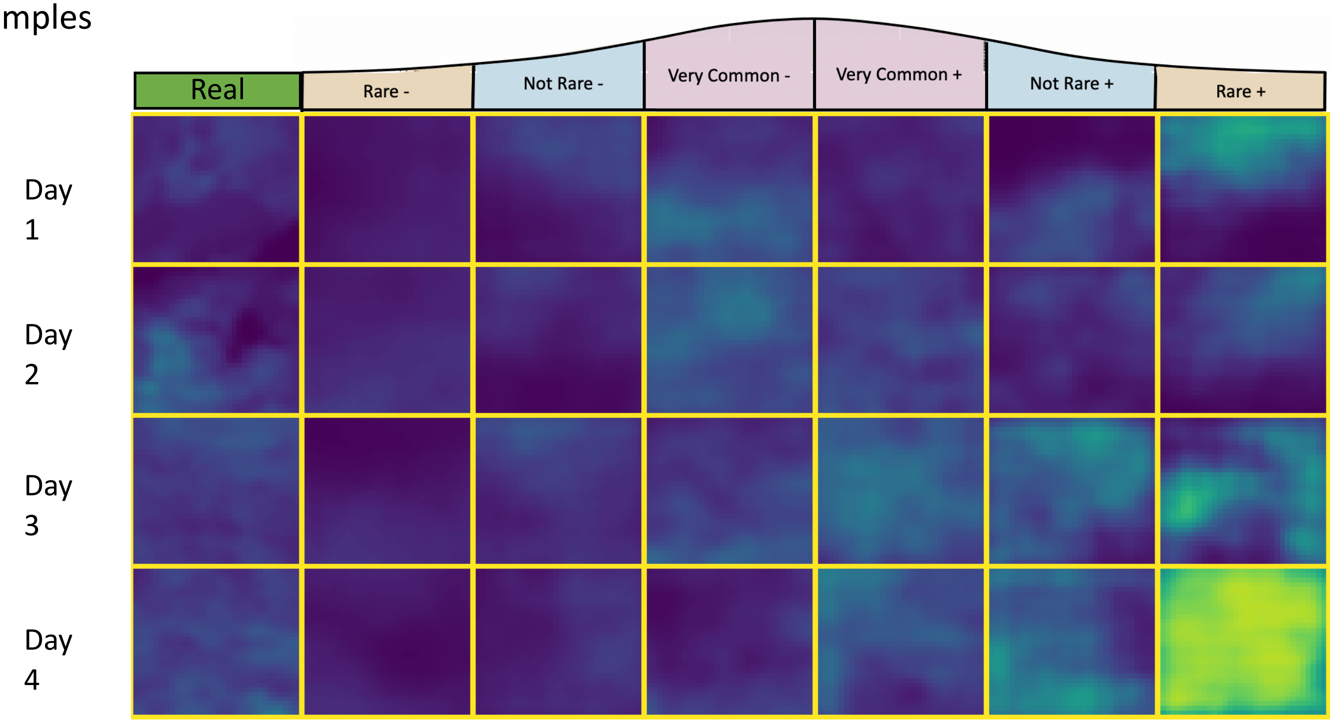


"Controlling Weather Field Synthesis Using Variational Autoencoders", Oliveira et al, ICML'21 Climate AI Workshop

Variational Autoencoders

Controllable Weather Generators using Distribution Priors

Random Daily Weather Field Samples

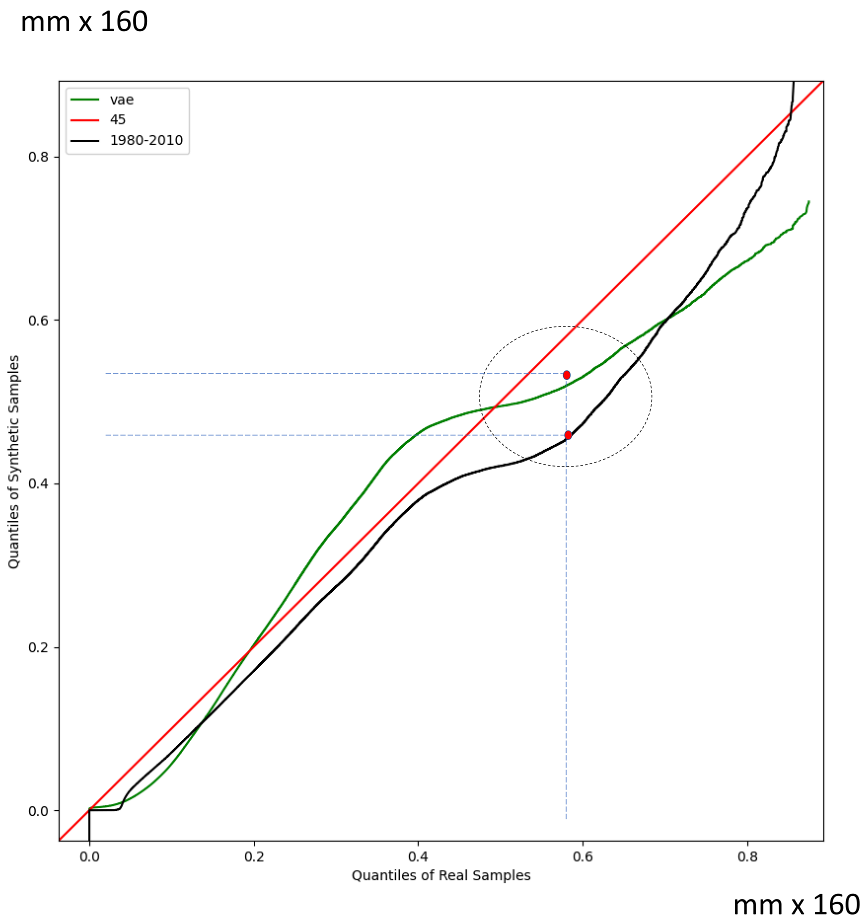


"Controlling Weather Field Synthesis Using Variational Autoencoders", Oliveira et al, ICML'21 Climate AI Workshop

Variational Autoencoders

Controllable Weather Generators using Distribution Priors

Quantile-Quantile Plot VAE vs Test



QQ-Plots compare distributions, where each point in the curves is the correspondence of their quantiles

- In the highlighted area, the amount of 80mm rain observed in test set was related to around 65mm in the historical data, and to around 80mm in the synthetic VAE data.

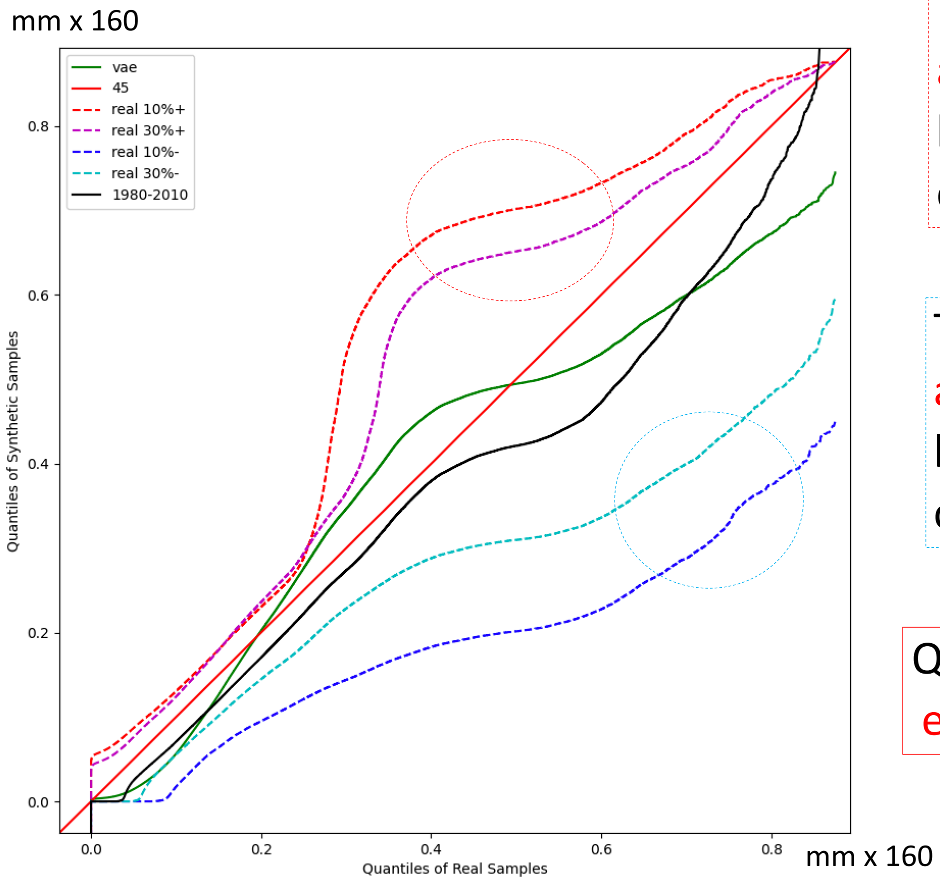
Historical data as a predictor underestimates precipitation for higher quantiles, which means that strong rain events increased comparing 1980-2010 to 2010-2020.

VAE synthetic data as a predictor is a bit better than historical data for above average values and a bit worse for the highest ones.

Variational Autoencoders

Controllable Weather Generators using Distribution Priors

Quantile-Quantile Plot for Extremes



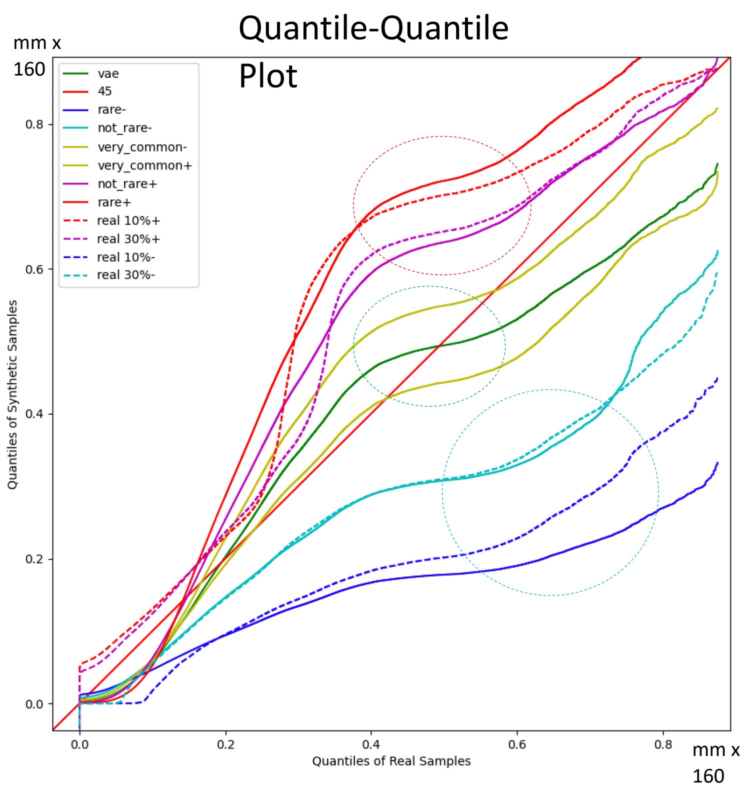
The 10% and 30% samples with more acumulated precipitation in the test set have considerably higher quantiles compared to overall test data (2010-2020)

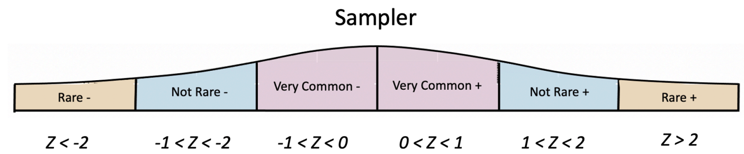
The 10% and 30% samples with less acumulated precipitation in the test set have considerably lower quantiles compared to overall test data (2010-2020)

QQ-plots can be used for evaluating extreme events compared to the regular distribution

Variational Autoencoders

Controllable Weather Generators using Distribution Priors





Synthetic samples for higher extremes are coherent with real test data for the 10% and 30% samples with the highest total monthly precipitation

Synthetic common scenarios are the ones closer to the overall test precipitation data distribution

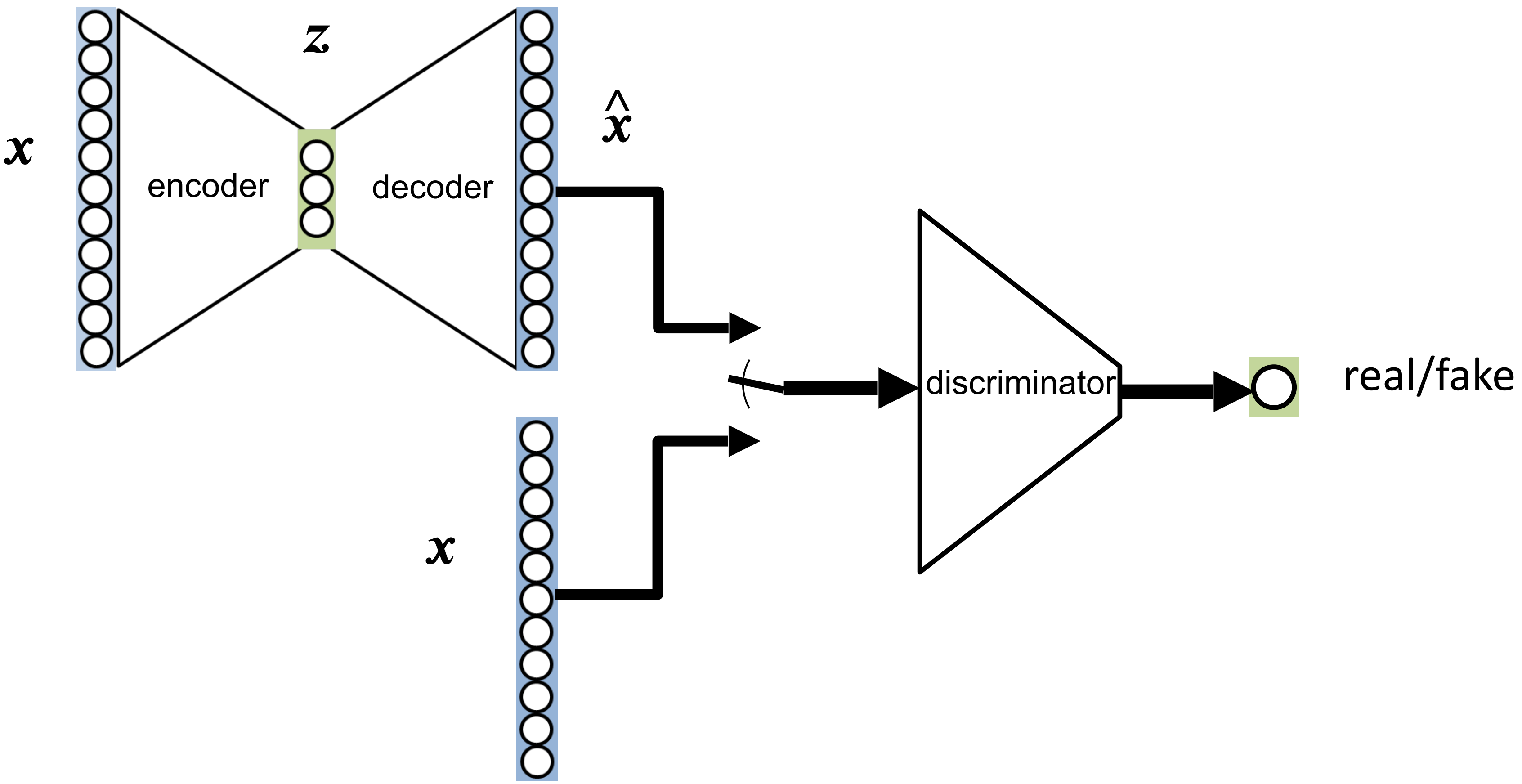
Synthetic samples for lower extremes are coherent with real test data for the 10% and 30% samples with the lowest total monthly precipitation

"Controlling Weather Field Synthesis Using Variational Autoencoders", Oliveira et al, ICML'21 Climate AI Workshop

Improving realism

VAE and GAN combined to produce more realistic images.

GAN's Discriminator loss term in the VAE loss.

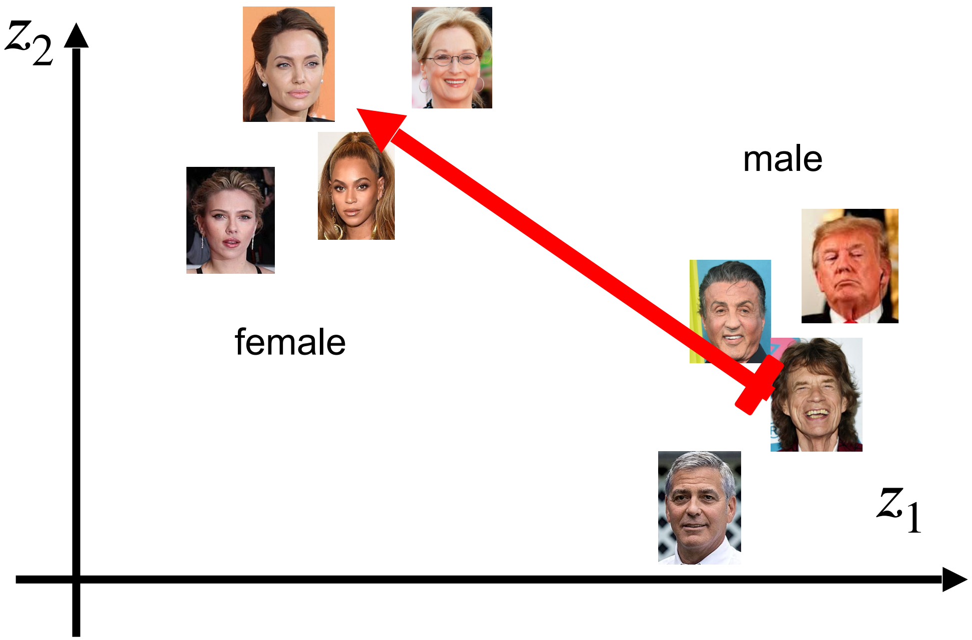


Source: Larsen et al. 2016 [Autoencoding beyond pixelsusing a learned similarity metric]

Disentangle latent variables

Check the region where images classes concentrate in the latent space .

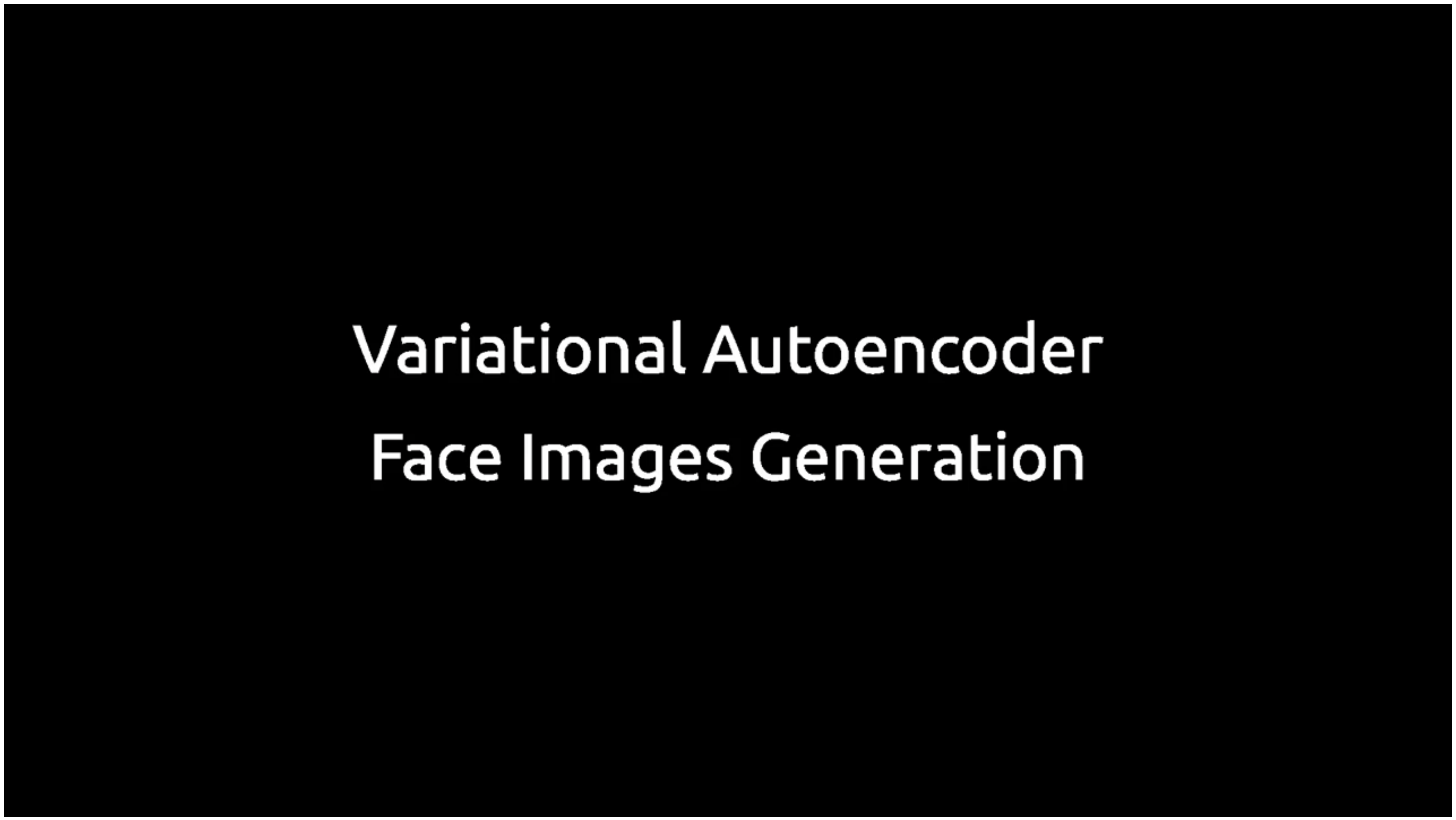
By moving along the vector connecting two regions will cause a morphing from one to the other class.



Disentangle latent variables

Source: Larsen et al. 2016 [Autoencoding beyond pixels using a learned similarity metric]

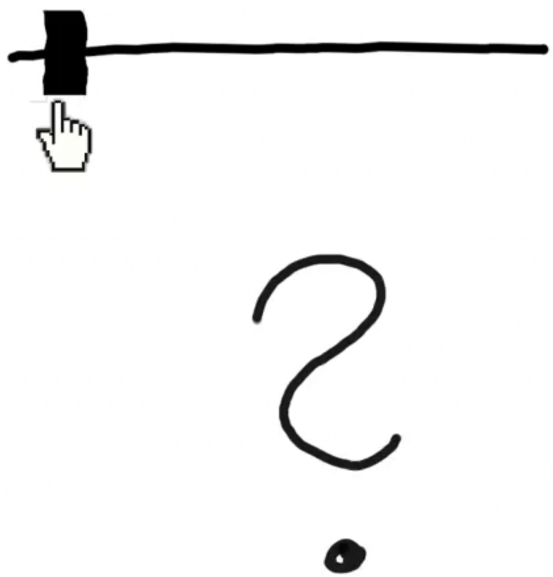
Generating Face Images with VAE

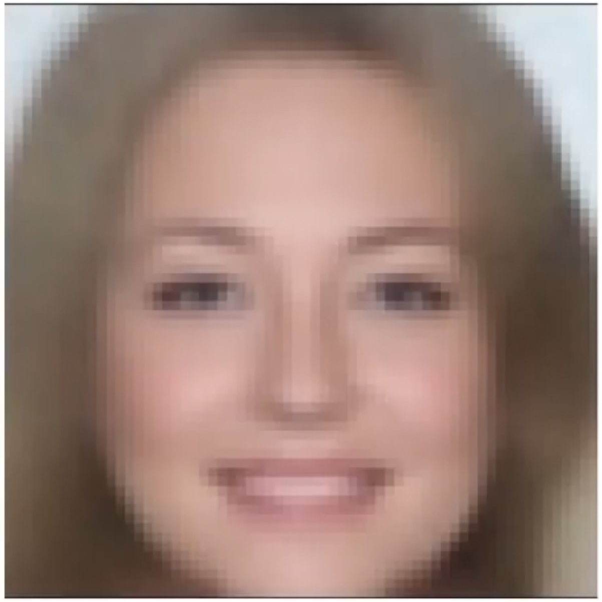


hit here to watch the demo

Higgins et al., 2017. ß-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework

Generating Face Images with VAE





Source: Steffen 2019 available at https://www.youtube.com/watch?v=uszj2MOLY08

VAE vs GAN

1. Both VAE and GAN are unsupervised learning
2. VAE
    - pro:
        - clear objective/cost function
        - Latent space more interpretable
    - con:
        - injected noise and imperfect reconstruction, result is blurred compared with GAN
3. GAN
    - pro:
        - result is better especially with noise? Nicer image.
    - con:
        - no clear object/cost function for comparison. Hard to train and converge

References on VAE

1. CARL DOERSCH, Tutorial on Variational Autoencoders,

2. Amini, et al., 2019 , Uncovering and Mitigating Algorithmic Bias through Learned Latent Structure, AIES '19: Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society, January 2019 Pages 289–295

3. Original VAE paper (2013): https://arxiv.org/abs/1312.6114

Next Lecture

Tuesday
Lab Autoencoders

See you next class ☺