Face Generation with VAEs using Pytorch

Veysel Oğulcan Kaya -17804
Barış Temel - 19233
Tunahan Parlayıcı -29913

Motivation

Main Objectives

- Generate new artificial face images from Celebrity Faces
- Face expression generation for group member selfies

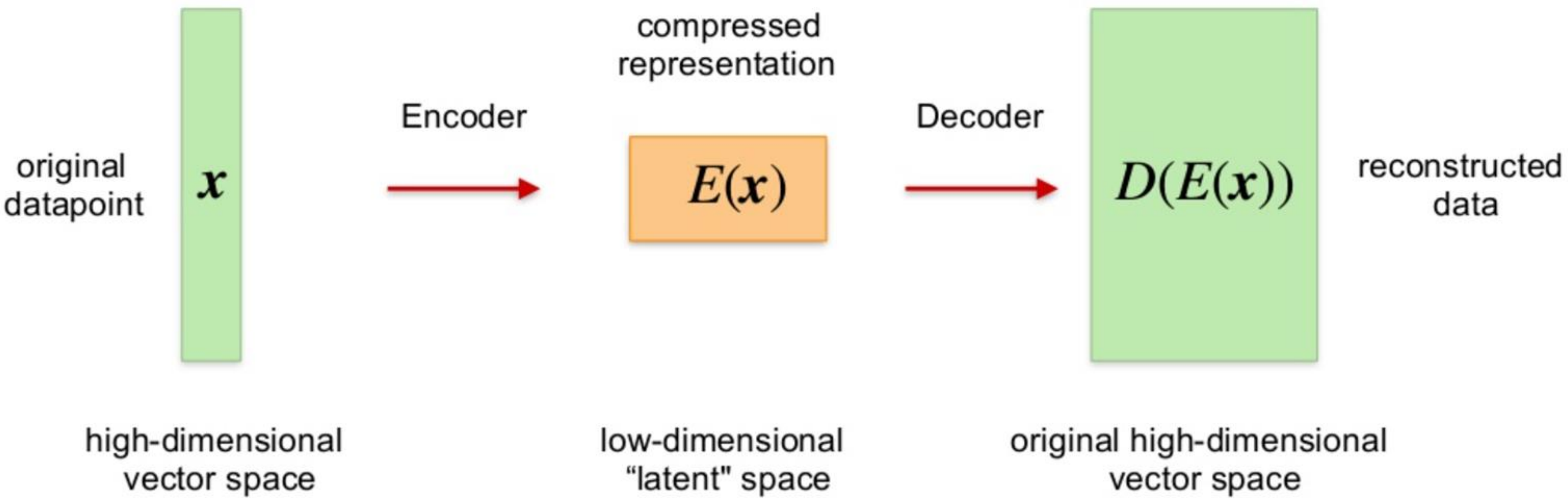
Learning Objectives

- Compete with standard AutoEncoders using VAE
- Learning encoder & decoder behaviours in Pytorch
- Expression synthesis with multiple Decoder architecture

$$L = \frac{1}{|T|} \sum_{x \in T} d(\boldsymbol{x}, D(E(\boldsymbol{x})))$$

Loss Function of AE

Standard Autoencoders(AE)



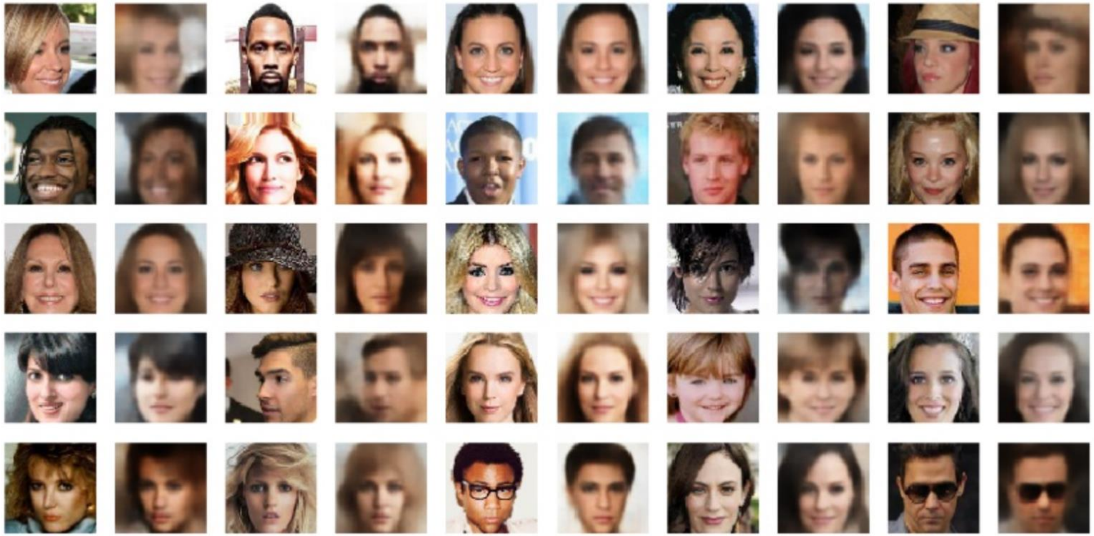
AE Face Reconstruction with celebA

- Dataset over 200k celebrity faces

- Limitation: Encoded representations

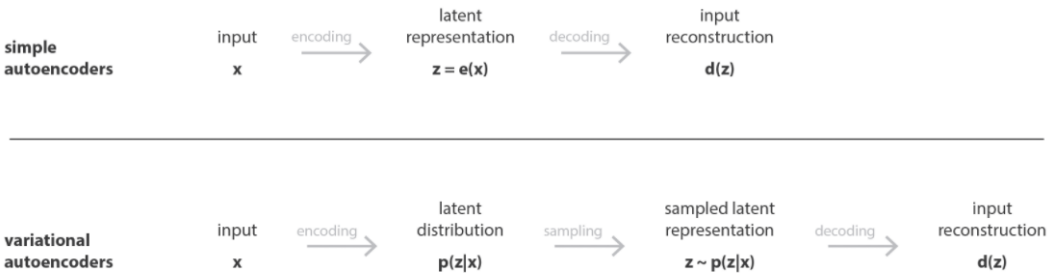
  Optimize for data reconstruction, not generation

- Using just AE was not giving good results

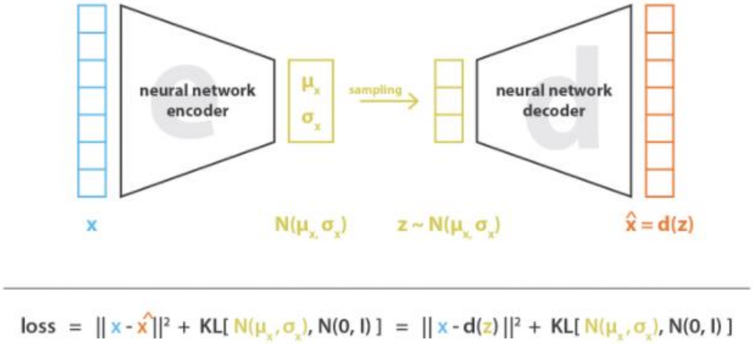


source literature

Variational Autoencoders



Difference between autoencoder (deterministic) and variational autoencoder (probabilistic).



Limitations

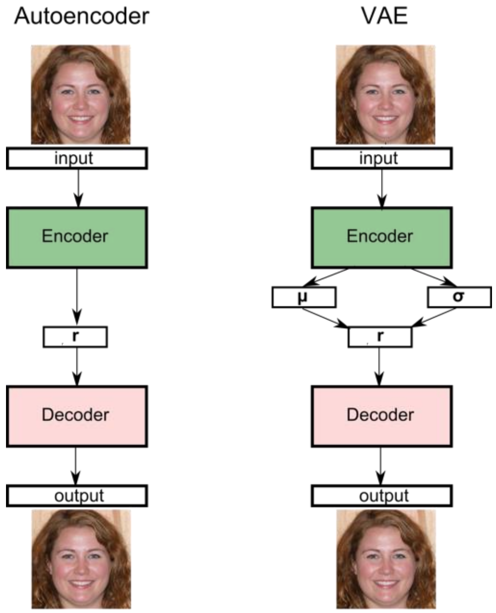
- If Image data is not just faces, images are often blurry
- Not better than GAN’s of today (but straightforward to implement)

Dataset

Examples of celebrity images in LFW & CelebA and FER13 dataset

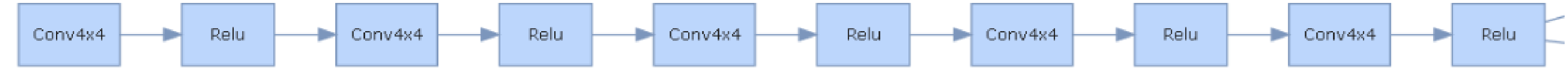
Model Architecture

- Encoder: 5 conv2d layers with kernel size 4, stride 2x2, padding 2
- Linear layers for reparameterization(backprop mean and variance)
- Decoder: 5 convTranspose2d layers with kernel size:3,stride:2x2

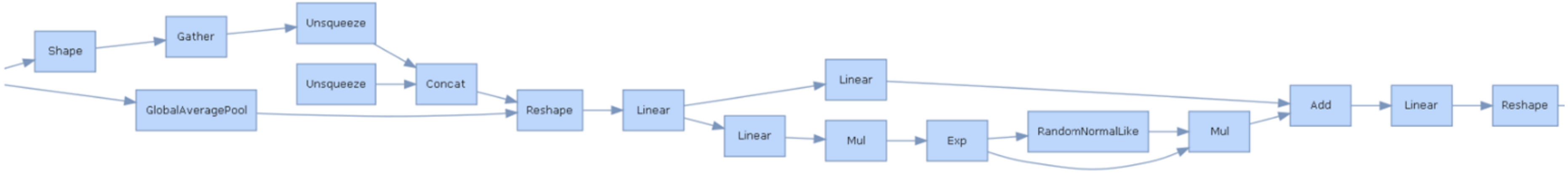


Model Architecture

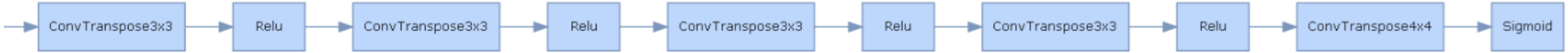
Encoder Part:



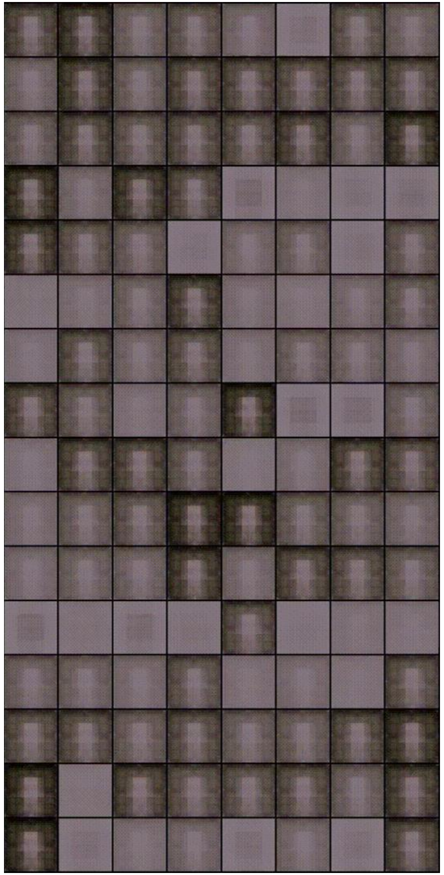
Reparameterization Part:



Decoder Part:



Results

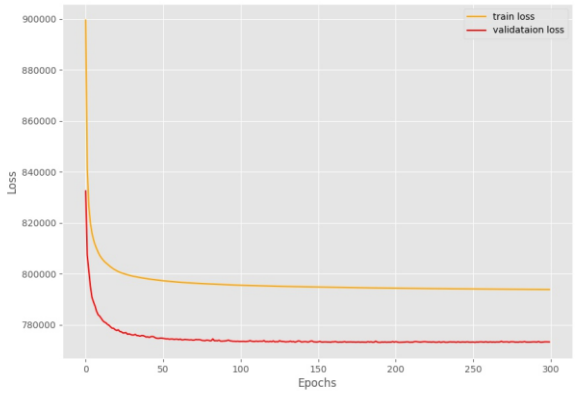


LFW generation

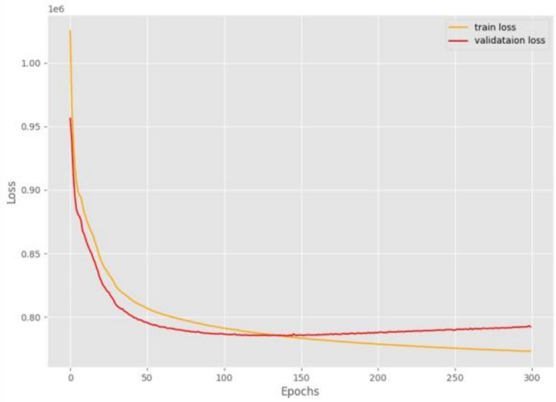
CelebA generation

FER13

Performance Measuring using Loss Function



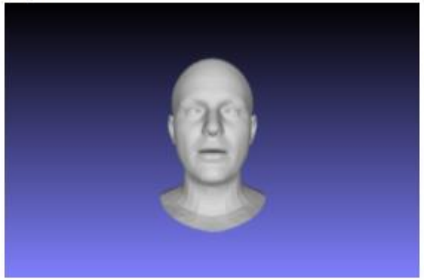
LFW



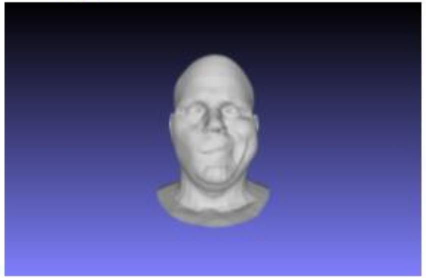
CelebA

Original COMA implementation (pyTorch version) (10 epochs only)

Input

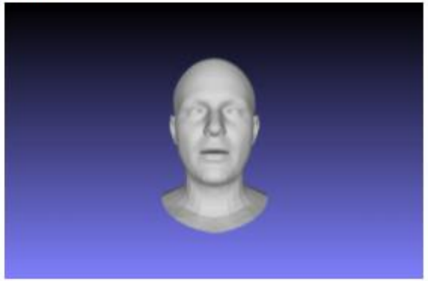


Output

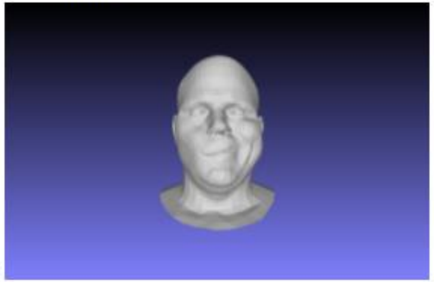


Vs Our implementation of GraphSAGE (pyTorch) (10 epochs only)

Input



Output



Layer	Avg. Runtime
ChebNet	30 mins per epoch
GraphSAGE	22 mins per epoch