Studying data variability in variational autoencoders using a chain model

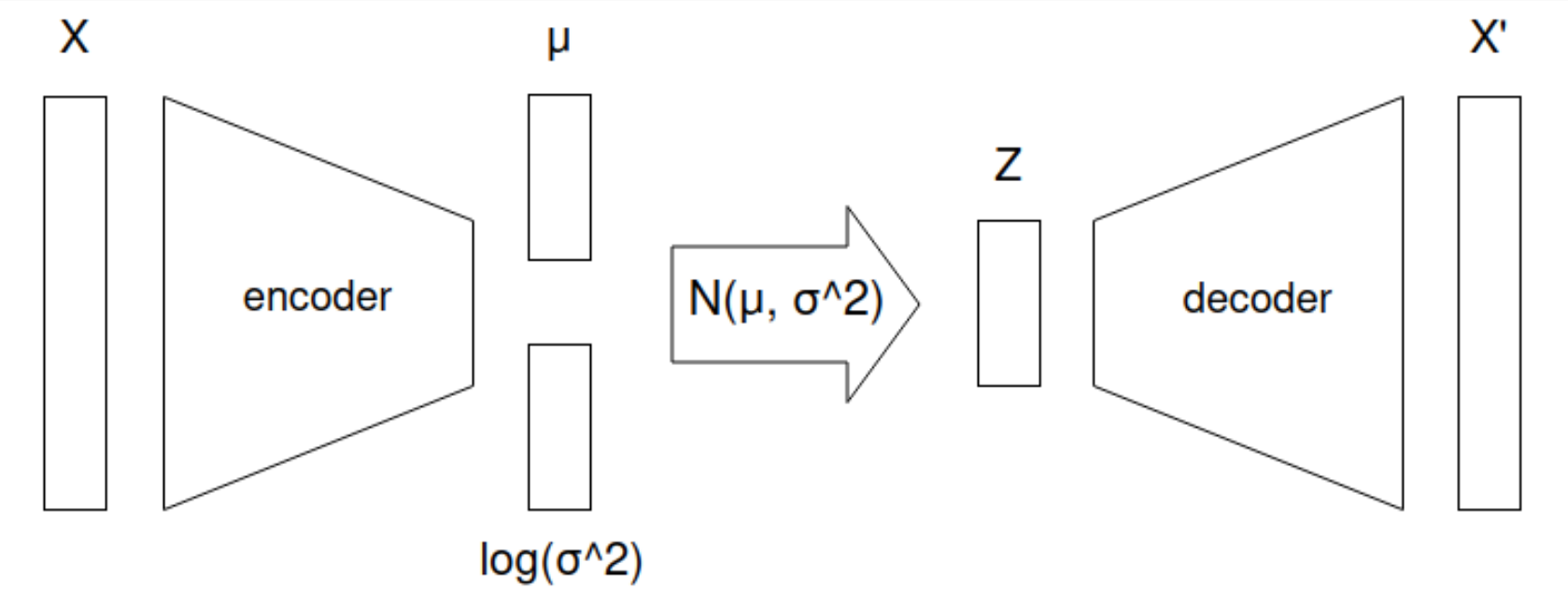
Tommaso Tarchi

January 15, 2024

University of Trieste

Introduction

General architecture of a VAE

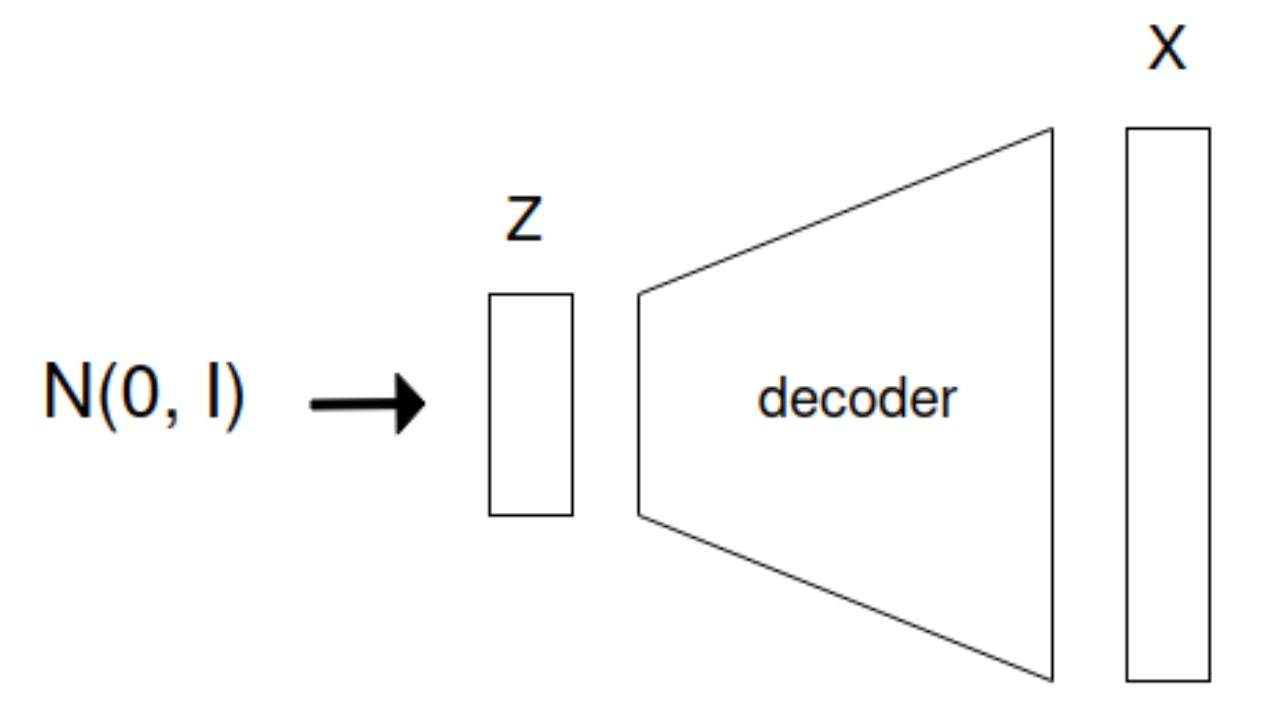


Loss function

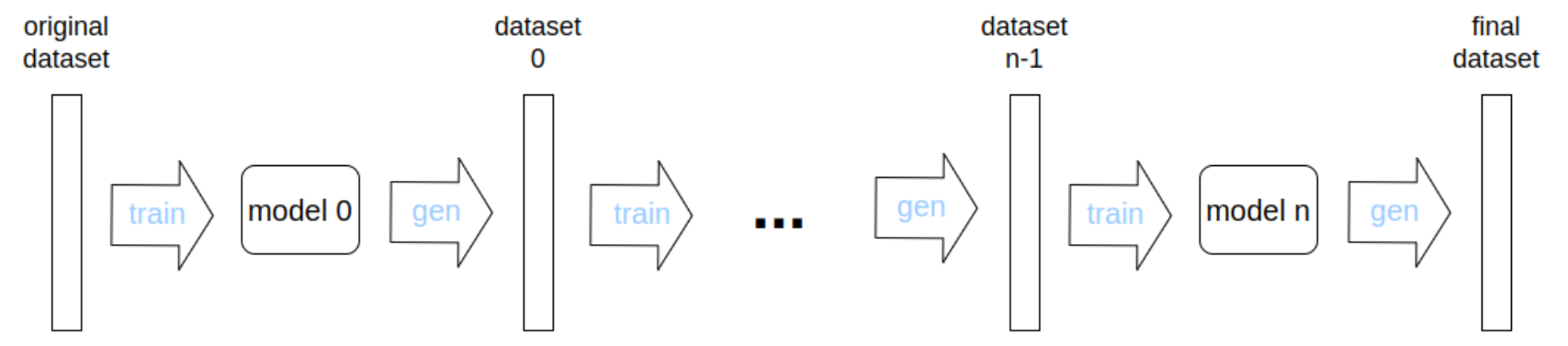
$$\text{LOSS}(\theta) = MSE_\theta\left(x, x_{recon}\right) + KL\left[q_\theta\left(z|x\right)||p\left(z\right)\right],$$

$$\text{with} \quad \text{p}(z) = N\left(z|0, I\right)$$

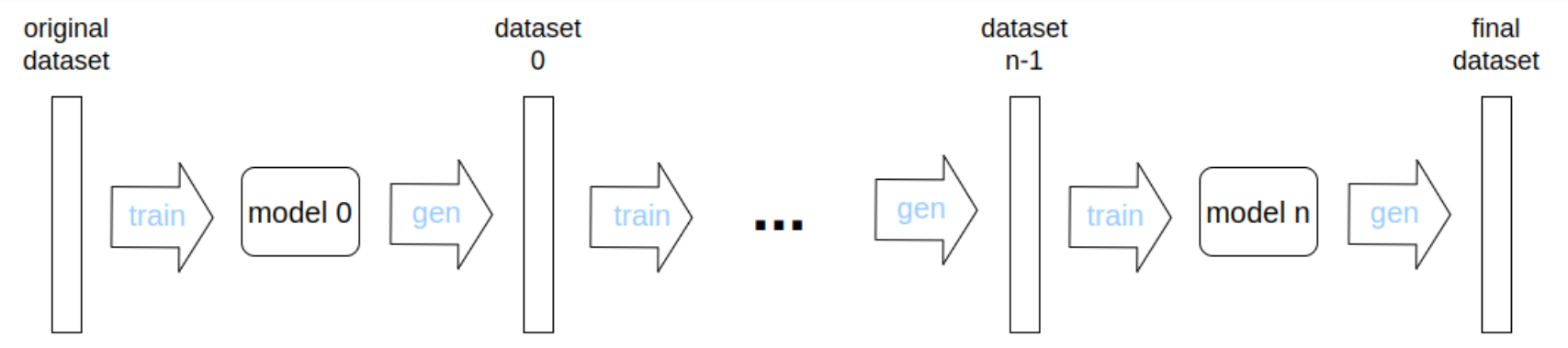
Generative mode



VAEs chain


original dataset
train
model 0
gen
dataset 0
train
...
gen
dataset n-1
train
model n
gen
final dataset

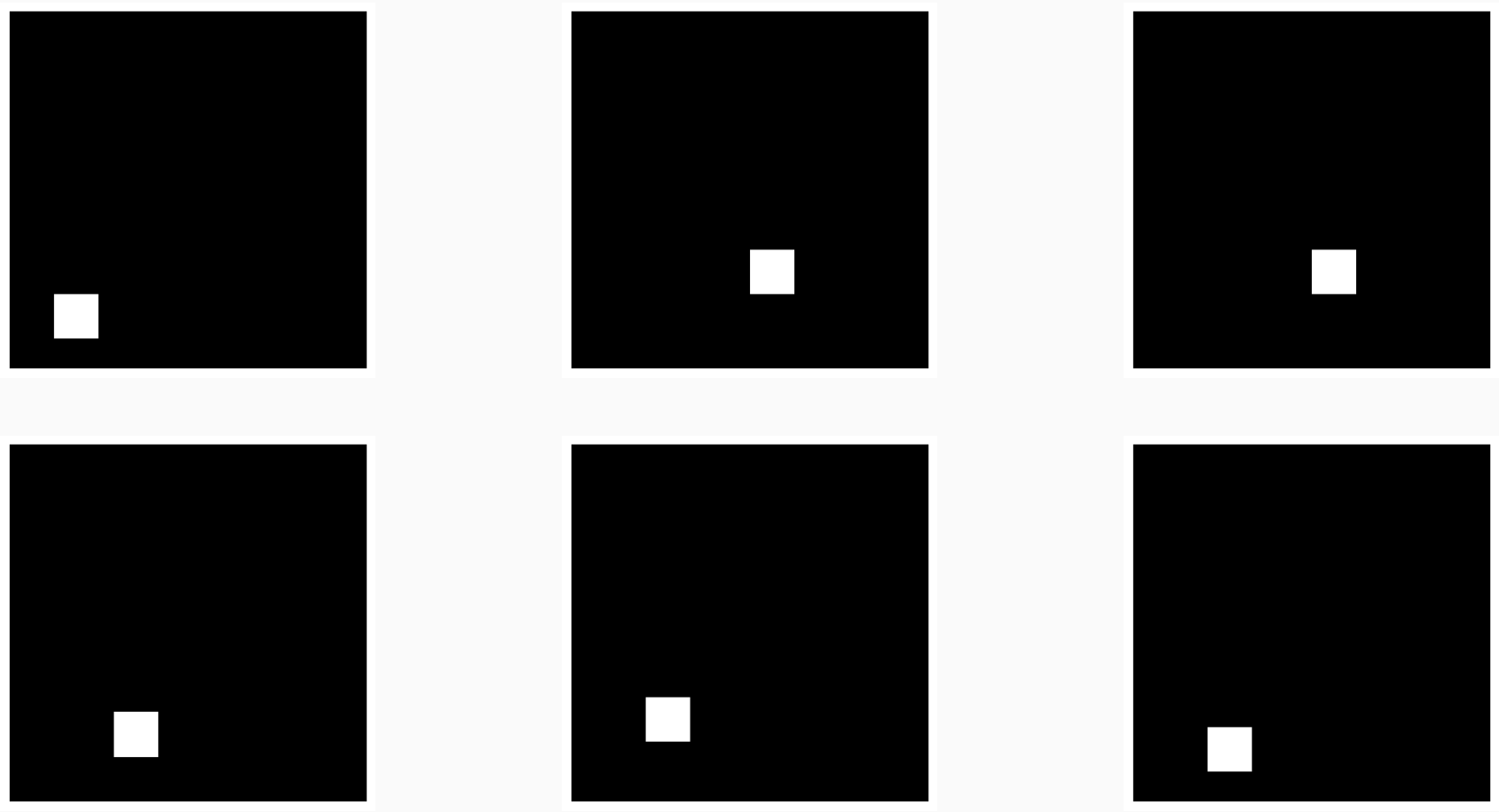
VAEs chain



Systematic effects introduced by VAEs will be amplified.

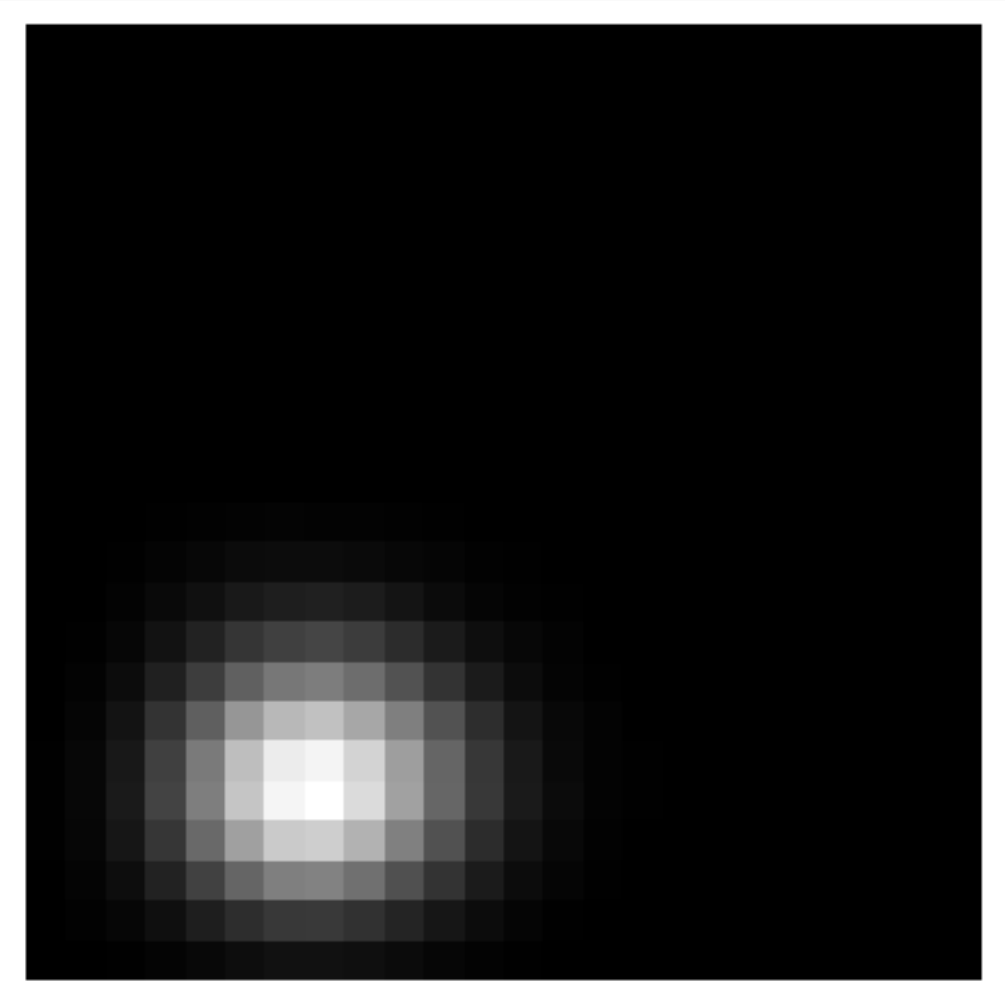
Our implementation

Initial dataset



24x24 grids, x-distribution: Binom(24,0.3), y-distribution: Binom(24,0.8)

Dataset's distribution



24x24 grids, x-distribution: Binom(24,0.3), y-distribution: Binom(24,0.8)

NN architectures

- One-layer fully connected (*FC1*)
- Two-layers fully connected (*FC2*)
- Convolutional with max-pooling (*Conv*)

How to evaluate results

- **Distribution difference**: difference between train dataset's and generated dataset's distributions (checks for biases)

How to evaluate results

- **Distribution difference**: difference between train dataset's and generated dataset's distributions (checks for biases)
- **Variability**: diversity of datasets (checks for variability reductions)

How to evaluate results

- **Distribution difference**: difference between train dataset's and generated dataset's distributions (checks for biases)
- **Variability**: diversity of datasets (checks for variability reductions)
- **Visual inspection**: checking random images visually (checks images to be of the right kind)

How to evaluate results

- **Distribution difference**: difference between train dataset's and generated dataset's distributions (checks for biases)
- **Variability**: diversity of datasets (checks for variability reductions)
- **Visual inspection**: checking random images visually (checks images to be of the right kind)

This method is NOT perfect!

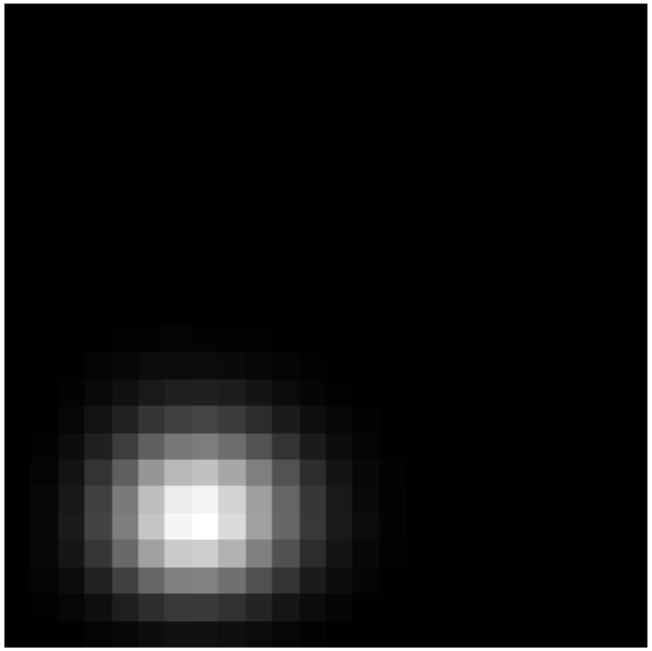
Results

Single model

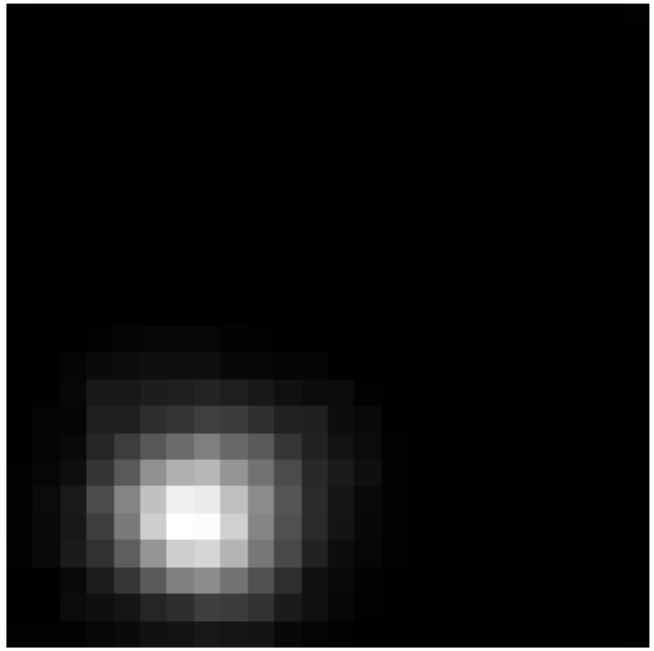
(a) Batch from original synthetic dataset

(b) Batch from dataset generated by VAE

Single model



(a) Distribution of original synthetic dataset



(b) Distribution of dataset generated by VAE

Whole chain

(a) Original dataset of the chain

(b) Dataset 1 of the chain

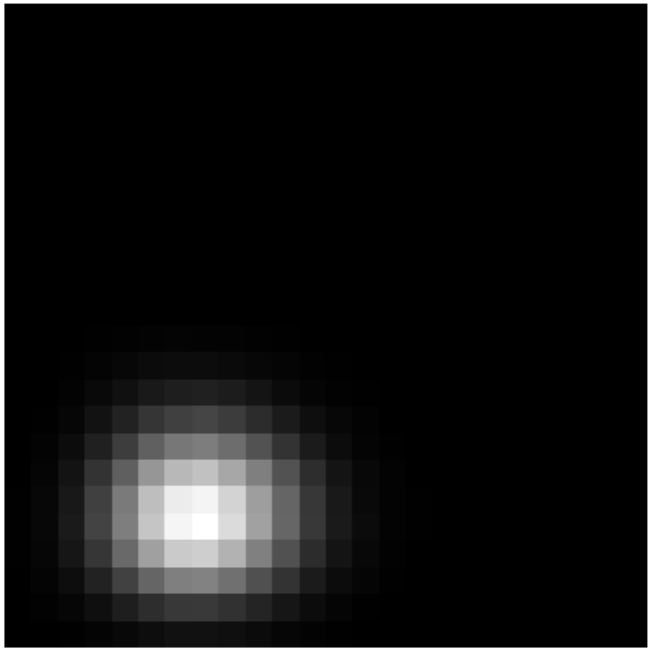
(c) Dataset 2 of the chain

(d) Dataset 3 of the chain

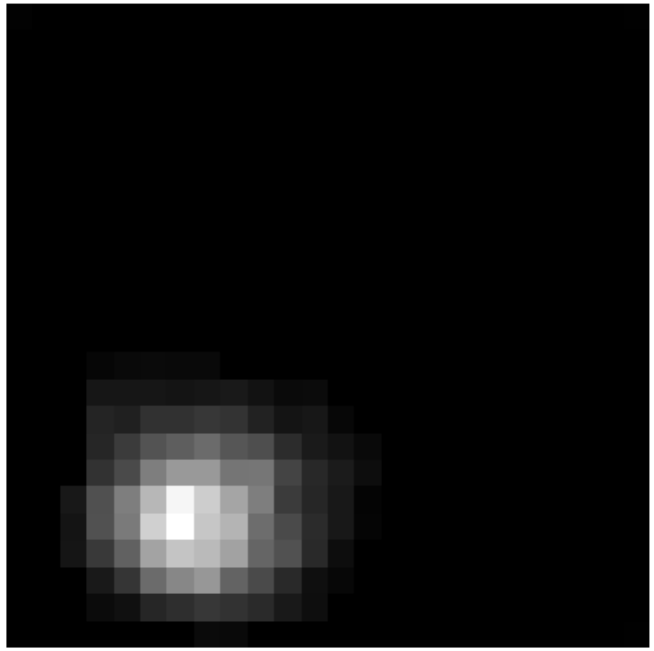
(e) Dataset 4 of the chain

(f) Final dataset of the chain

Whole chain

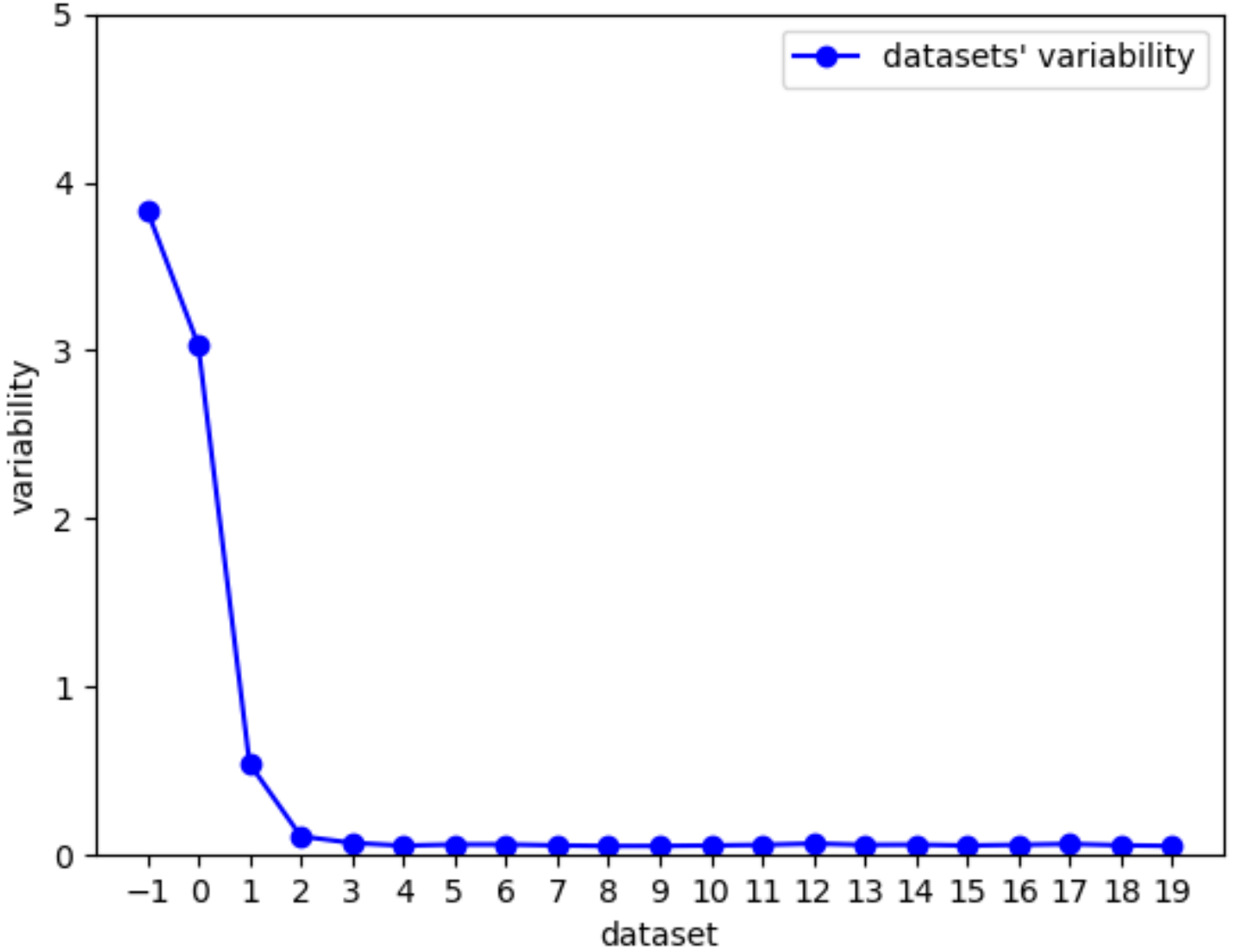


(a) Distribution of original synthetic dataset



(b) Distribution of dataset generated by last model in chain

Whole chain


datasets' variability
variability
dataset
0
1
2
3
4
5
−1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

Observations

- Images get lighter and more spread
- No evident bias is introduced in the data distribution (the data distribution is well "captured")
- **Variability is lost**

How can we fix this?

Possible causes

Possible causes

- **Regularization term**: the distribution in the latent space is forced to be close to a standard normal

Possible causes

- **Regularization term**: the distribution in the latent space is forced to be close to a standard normal
- **Dimensionality reduction**: less fundamental features are lost in latent space representation

Possible causes

- **Regularization term**: the distribution in the latent space is forced to be close to a standard normal
- **Dimensionality reduction**: less fundamental features are lost in latent space representation
- **Noisy reconstruction**: noise causes images to be more and more spread, may leading to more and more similar reconstructions

Possible causes

- **Regularization term**: the distribution in the latent space is forced to be close to a standard normal
- **Dimensionality reduction**: less fundamental features are lost in latent space representation
- **Noisy reconstruction**: noise causes images to be more and more spread, may leading to more and more similar reconstructions
- **Autoencoders' symmetry**: using mirror architectures for encoders and decoders may lead to unpredicted systematic patterns and behaviour

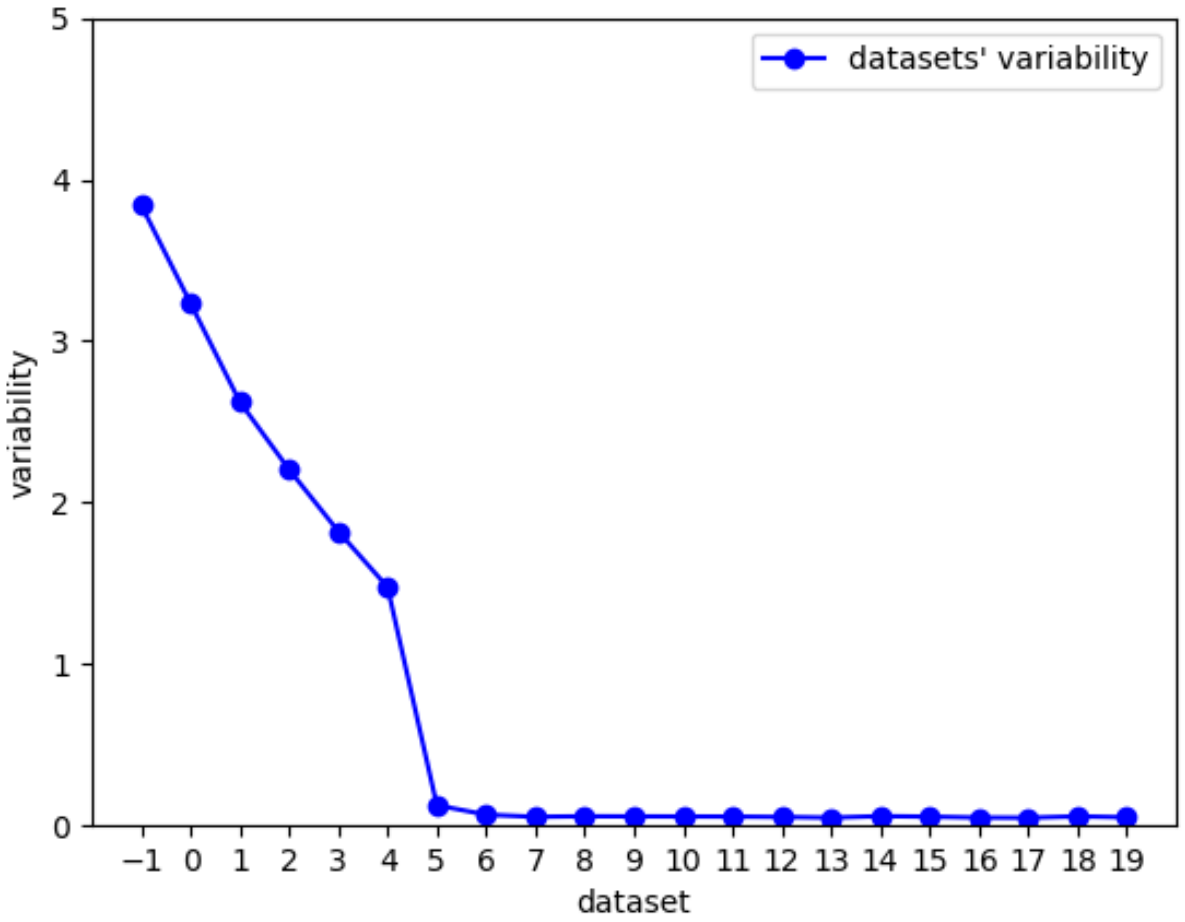
Possible causes

- **Regularization term**: the distribution in the latent space is forced to be close to a standard normal
- **Dimensionality reduction**: less fundamental features are lost in latent space representation
- **Noisy reconstruction**: noise causes images to be more and more spread, may leading to more and more similar reconstructions
- **Autoencoders' symmetry**: using mirror architectures for encoders and decoders may lead to unpredicted systematic patterns and behaviour
- **All (or some) of them combined**

Regularization constant tuning

$$\text{LOSS}(\theta) = MSE_\theta\left(x, x_{recon}\right) + \lambda KL\left[q_\theta\left(z|x\right)||p\left(z\right)\right]$$

Regularization constant tuning - results



Variability over models of the chain using $\lambda = 0.3$

Regularization constant tuning - results

(a) Batch from dataset generated by last model of the chain with $\lambda = 1$ (untuned model)

(b) Batch from dataset generated by last model of the chain with $\lambda = 0.3$ (tuned model)

Regularization constant tuning - conclusion

Pros:

- helps maintaining variability longer
- easy to implement
- easy to tune ("linear" behaviour)

Cons:

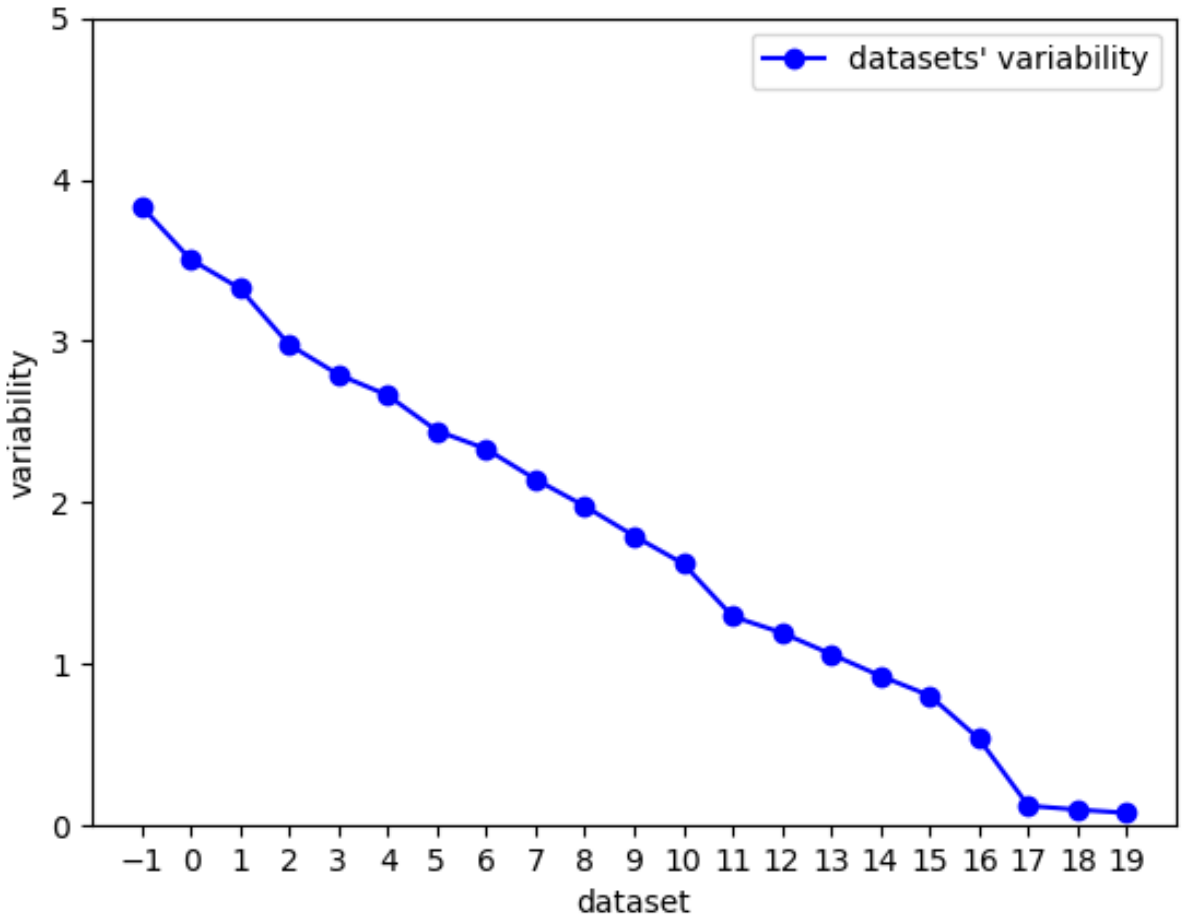
- less precise outputs (latent space distribution less close to a standard normal)
- eventually leads to zero variability

Variability loss term

$$\mathrm{LOSS}_{batch}(\theta) = \sum_i MSE_\theta\left(x_i, x_i^{recon}\right) + \sum_i KL\left[q_\theta\left(z_i|x_i\right)||p\left(z_i\right)\right]$$

$$+ \mathrm{K}\left|\sum_{i,j} MSE\left(x_i, x_j\right) - \sum_{i,j} MSE_\theta\left(x_i^{recon}, x_j^{recon}\right)\right|$$

Variability loss term - results

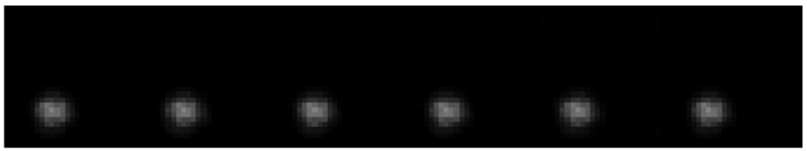


Dataset variability over models of the chain using $K = 10$

Variability loss term - results

(a) Batch from dataset generated by last model of the chain with $\lambda = 1$ and $K = 0$ (no tuning)

(b) Batch from dataset generated by last model of the chain with $\lambda = 0.3$ and $K = 0$ (regularization tuning)



(c) Batch from dataset generated by last model of the chain with $\lambda = 1$ and $K = 10$ (variability loss tuning)

Variability loss term - conclusion

Pros:

- helps maintaining variability longer
- easy to tune
- **better variability-precision trade-off than regularization term tuning**
- **less light images**

Cons:

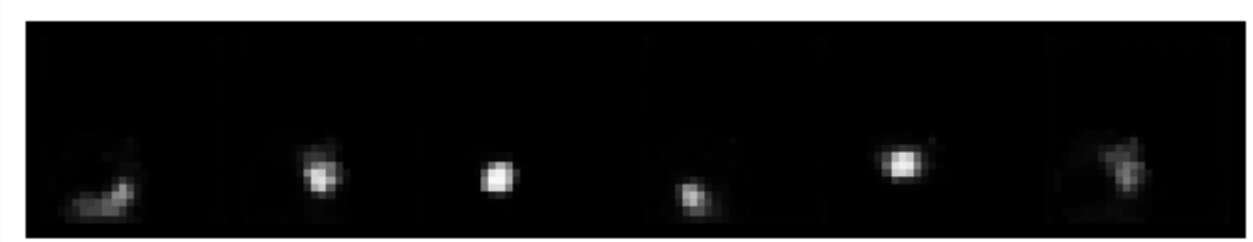
- still, eventually leads to zero variability

Denoising

(a) Undenoised data

(b) Denoised data ($thres = 50\%$)

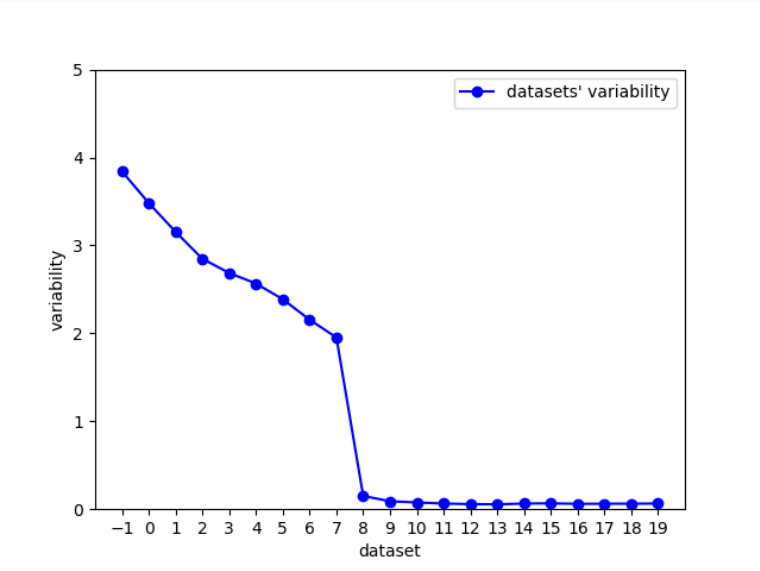
Improving non-optimal parameters with denoising



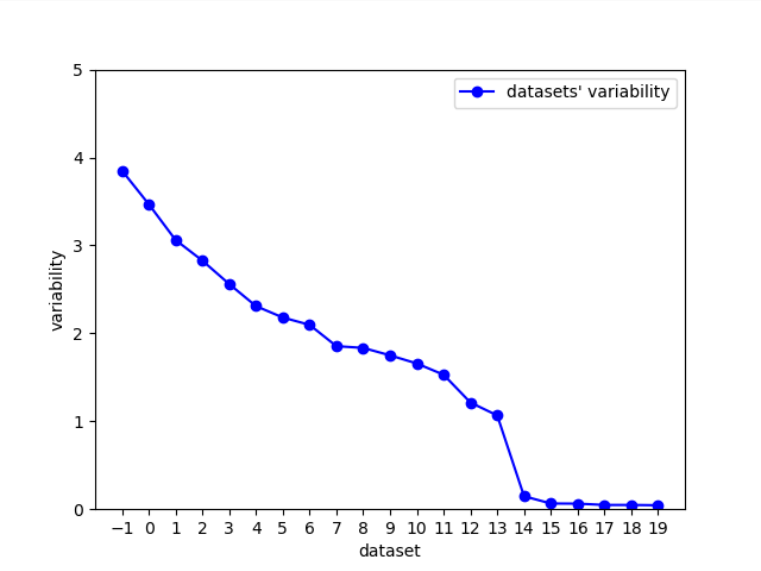
(a) Batch from fifth dataset without denoising ($\lambda = 0.7$ and $K = 10$)

(b) Batch from fifth dataset with denoising ($\lambda = 0.7$ and $K = 10$, thres=5%)

Improving non-optimal parameters with denoising

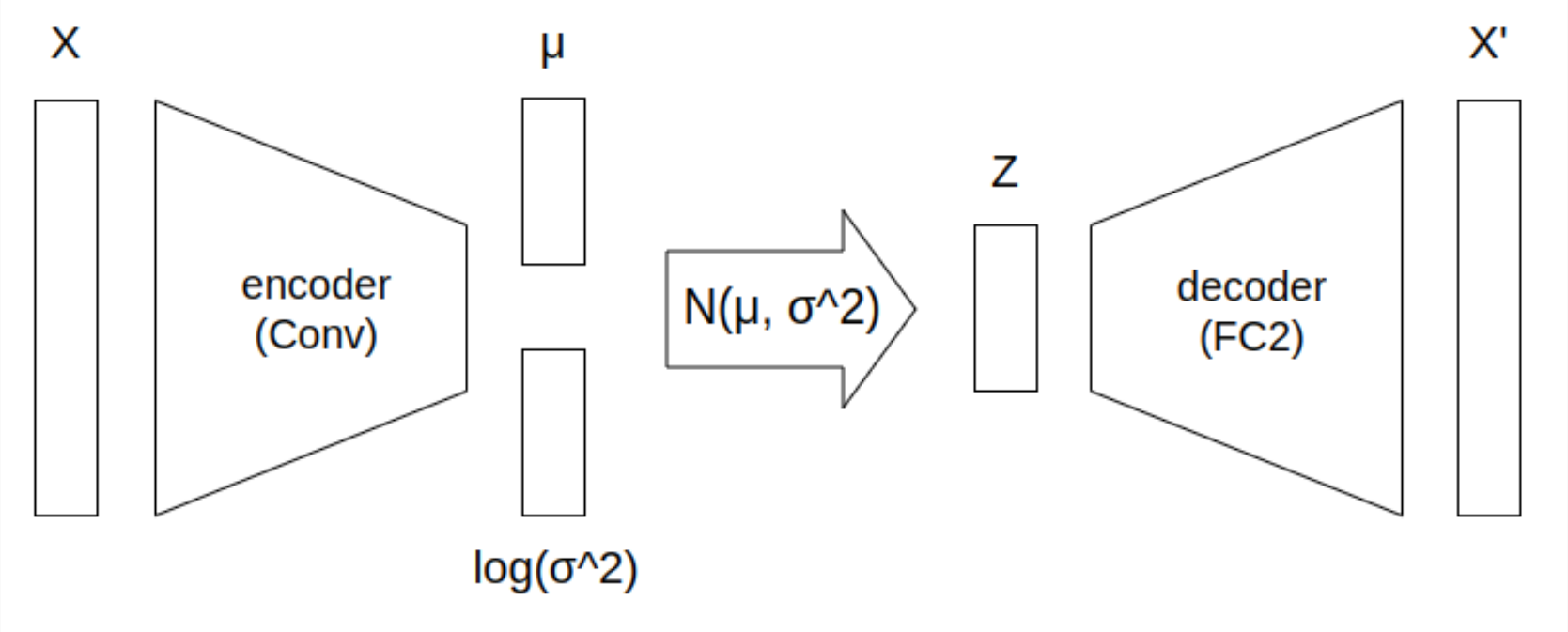


(a) Dataset variability without denoising ($\lambda = 0.7$, $K = 10$)

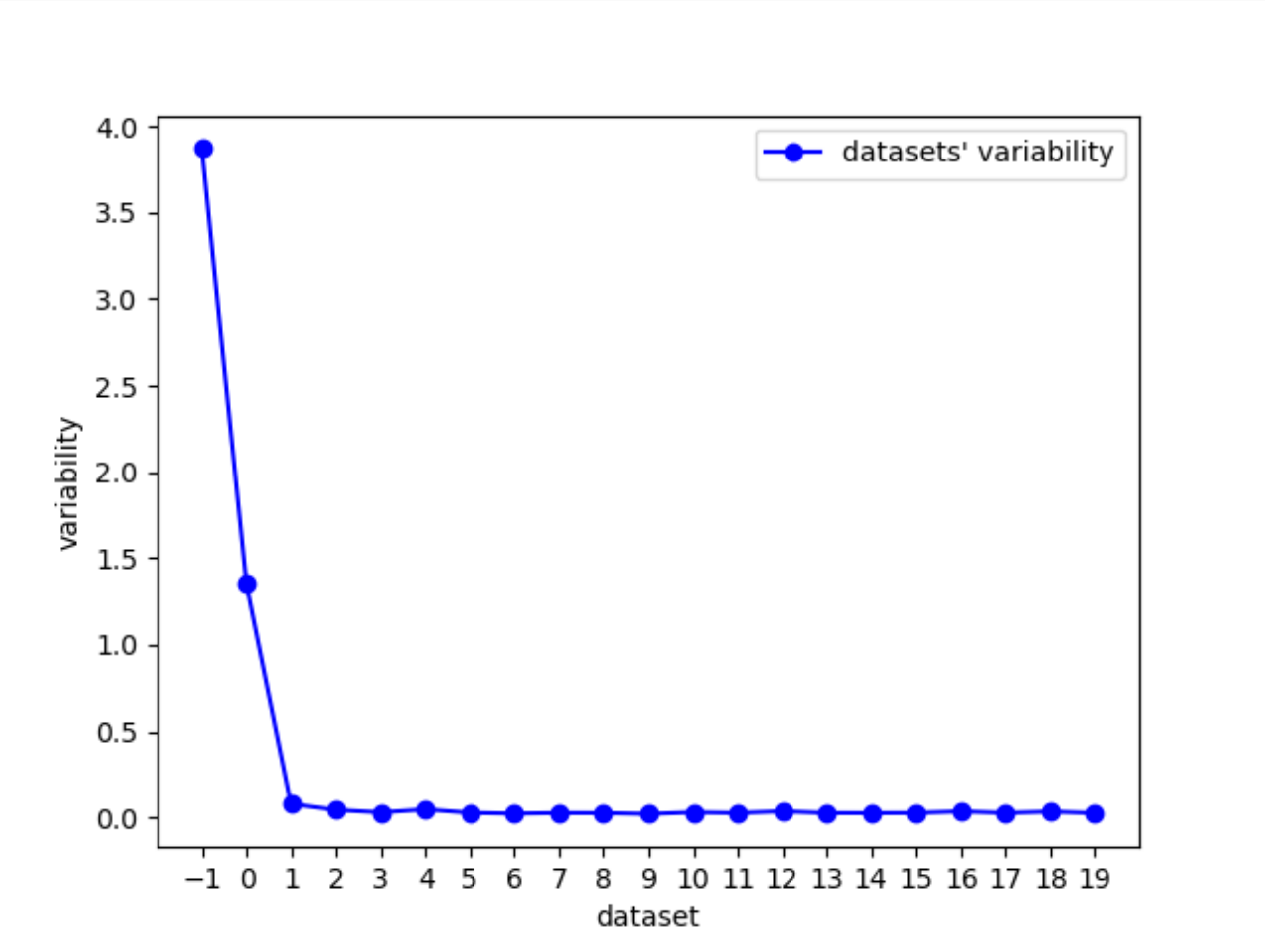


(b) Dataset variability with denoising ($\lambda = 0.7$, $K = 10$, thres=5%)

Asymmetric architecture

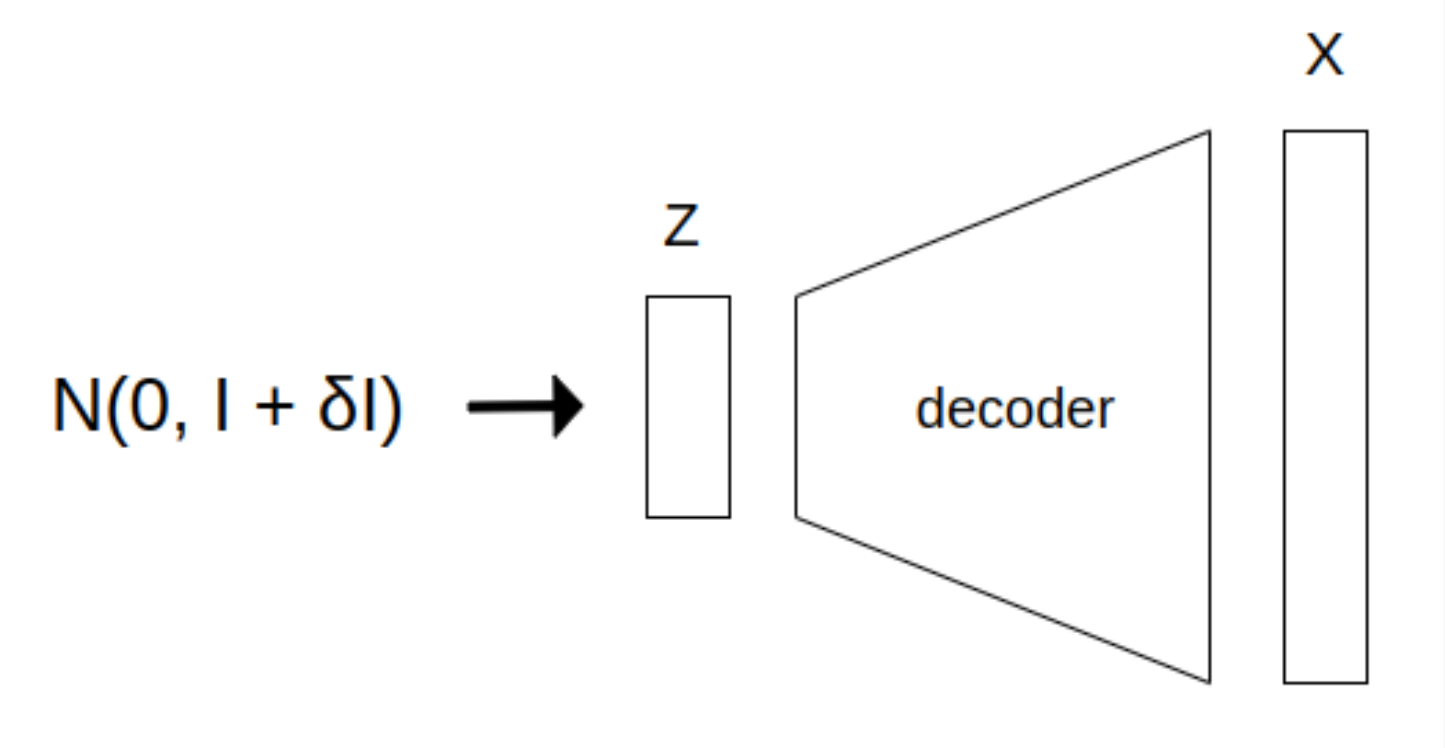


Asymmetric architecture - results

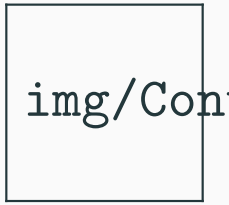


Dataset variability over models of the chain with asymmetric architecture

Wider distribution in generative mode

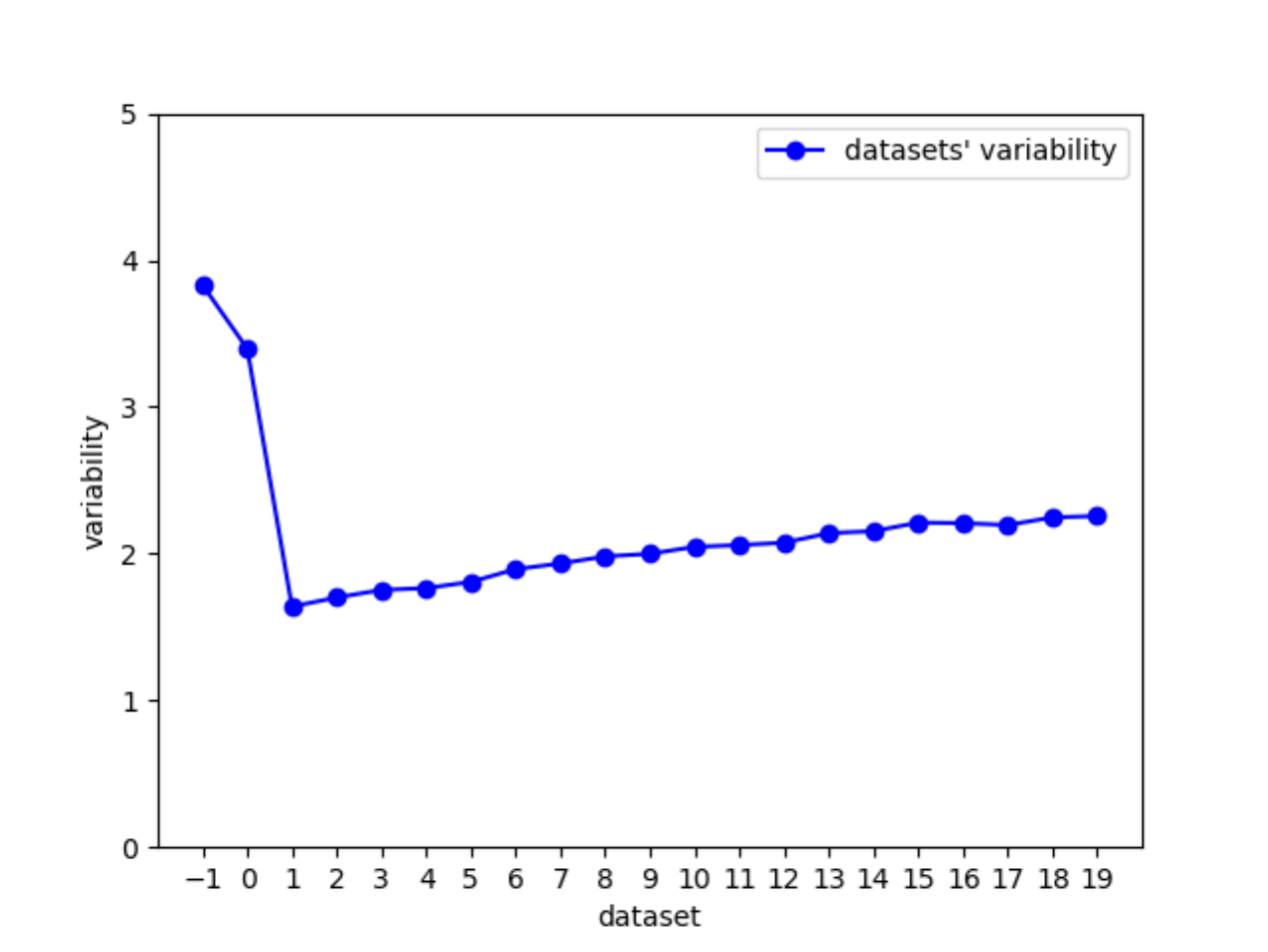


Wider distribution in generative mode - small increment



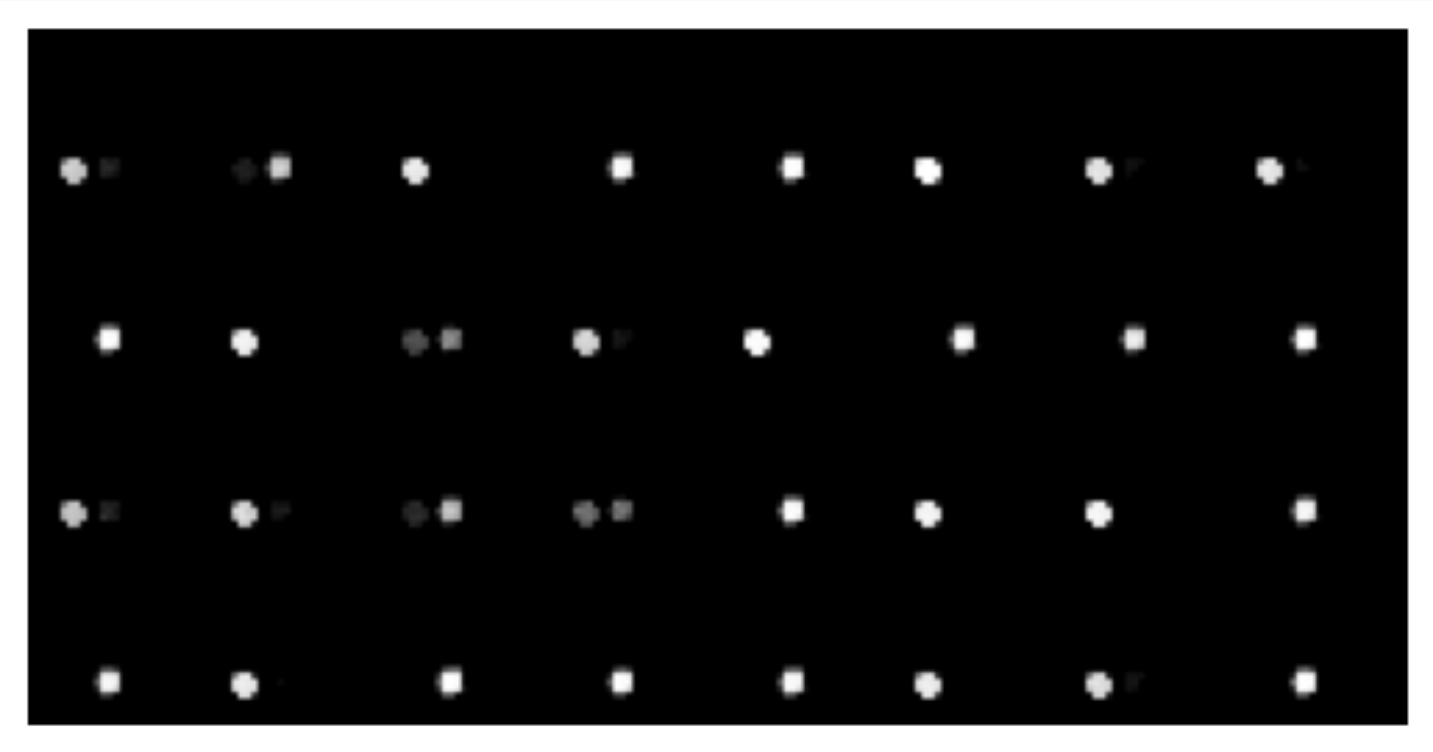
Dataset variability over models of the chain with $\delta = 0.7$, using denoising ($thres = 5\%$)

Wider distribution in generative mode - large increment



Dataset variability over models of the chain with $\delta = 0.7$, using denoising ($thres = 5\%$)

Wider distribution in generative mode - large increment



Batch from dataset generated by last model of the chain with $\delta = 0.7$, using denoising ($thres = 5\%$)

Wider distribution in generative mode - conclusion

Pros:

- helps maintaining variability longer
- **for some values of δ, keeps variability different from zero indefinitely**

Cons:

- may lead to "subclasses formation" (but they change at each iteration)
- may change distribution slightly
- very hard to tune (strongly non-linear behaviour)
- unpredictable outcome on more complex datasets
- may need denoising to work

Can we really fix this problem?

Can we really fix this problem?

Probably not! Due to information loss in latent space representation.

Can we really fix this problem?

Probably not! Due to information loss in latent space representation.

We can only either mitigate this effect or introduce "artificial" variability.

More general issue

This is just a very simple example of a much larger and fundamental problem:

AI models trained on AI-generated data.

Thank you!