UE D Deep Learning
Supervised by Farrugia Nicolas

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Autoregressive Image Modeling

OUESLATI Mohamed Amine
JEBARI Khaled

2025-2026

Summary

01 Introduction

For an image represented as a vector

$$x = (x_1, x_2, \ldots, x_D)$$

An autoregressive model writes the joint distribution as

$$p(x) = \prod_{i=1}^{D} p(x_i \mid x_1, \ldots, x_{i-1})$$

01 Introduction

These models are usually trained via **maximum likelihood estimation**

$$\mathcal{L}(\theta) = -\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}\left[\log p_\theta(\mathbf{x})\right] = -\mathbb{E}_{\mathbf{x}}\left[\sum_{i=1}^{D} \log p_\theta(x_i \mid x_{<i})\right]$$

Sampling and generation:

1. Sample $x_1 \sim p(x_1)$,
2. Sample $x_2 \sim p(x_2 \mid x_1)$,
3. ... up to $x_D \sim p(x_D \mid x_{<D})$.

01 Introduction

We used **FID (Fréchet Inception Distance)** as an evaluation metric to compare images generated pixel-by-pixel by the model with real images, focusing on **distributional similarity** rather than likelihood

$$\mathrm{FID} = \|\mu_r - \mu_g\|_2^2 + \mathrm{Tr}\left(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}\right)$$

02

MADE

02 MADE

MADE (Masked Autoencoder for Distribution Estimation)

Turn an autoencoder into a valid **autoregressive density model** by enforcing the factorization using **binary masks** on network connections.

Hidden layer

$$\mathbf{h}(\mathbf{x}) = g\big(\mathbf{b} + (W \odot M_W)\,\mathbf{x}\big) \qquad (M_W)_{k,d} = \mathbf{1}\,[\,m(k) \geq d\,]$$

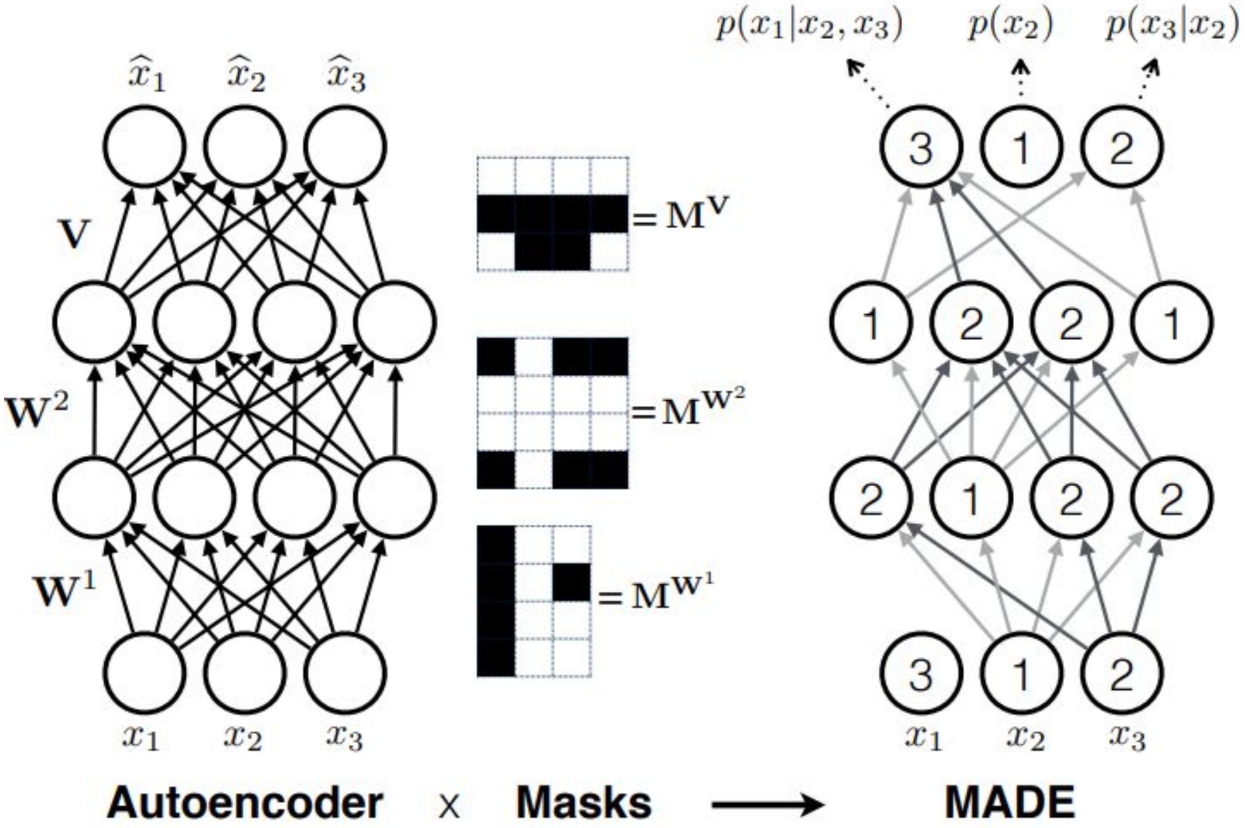
The mask removes illegal dependencies on future inputs

Output layer

$$\hat{\mathbf{x}} = \sigma\big(\mathbf{c} + (V \odot M_V)\,\mathbf{h}(\mathbf{x})\big) \qquad (M_V)_{d,k} = \mathbf{1}\,[\,d > m(k)\,]$$

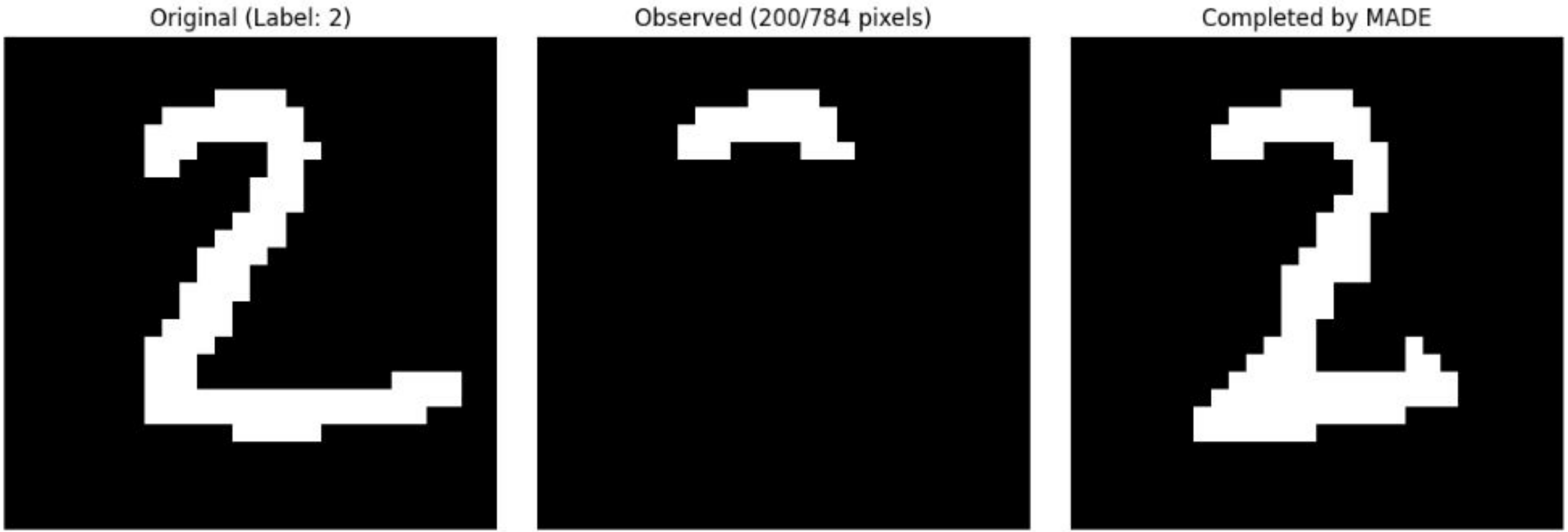
The mask ensures autoregressive structure

Each output $\hat{x}_d = p(x_d \mid x_{<d})$

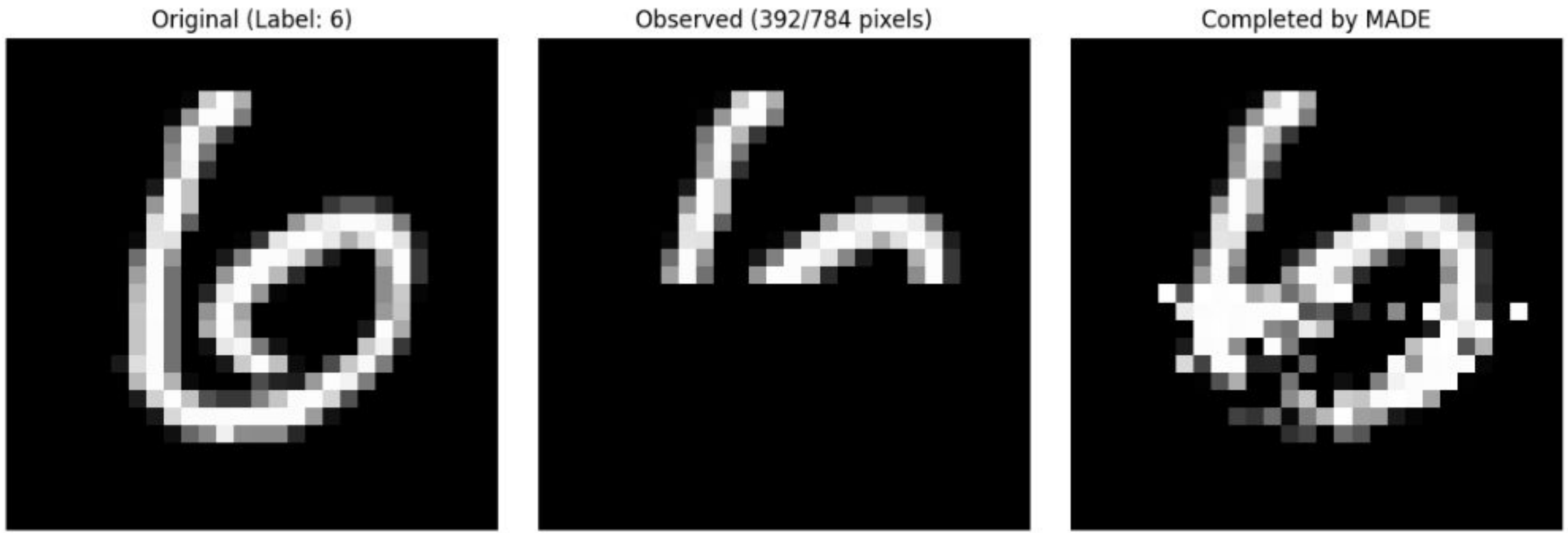


$p(x_1|x_2, x_3)$
$p(x_2)$
$p(x_3|x_2)$
$\widehat{x}_1$ $\widehat{x}_2$ $\widehat{x}_3$
$\mathbf{V}$
$\mathbf{W}^2$
$\mathbf{W}^1$
$= \mathbf{M}^{\mathbf{V}}$
$= \mathbf{M}^{\mathbf{W}^2}$
$= \mathbf{M}^{\mathbf{W}^1}$
3 1 2
1 2 2 1
2 1 2 2
3 1 2
x_1 x_2 x_3
x_1 x_2 x_3
Autoencoder × Masks → MADE

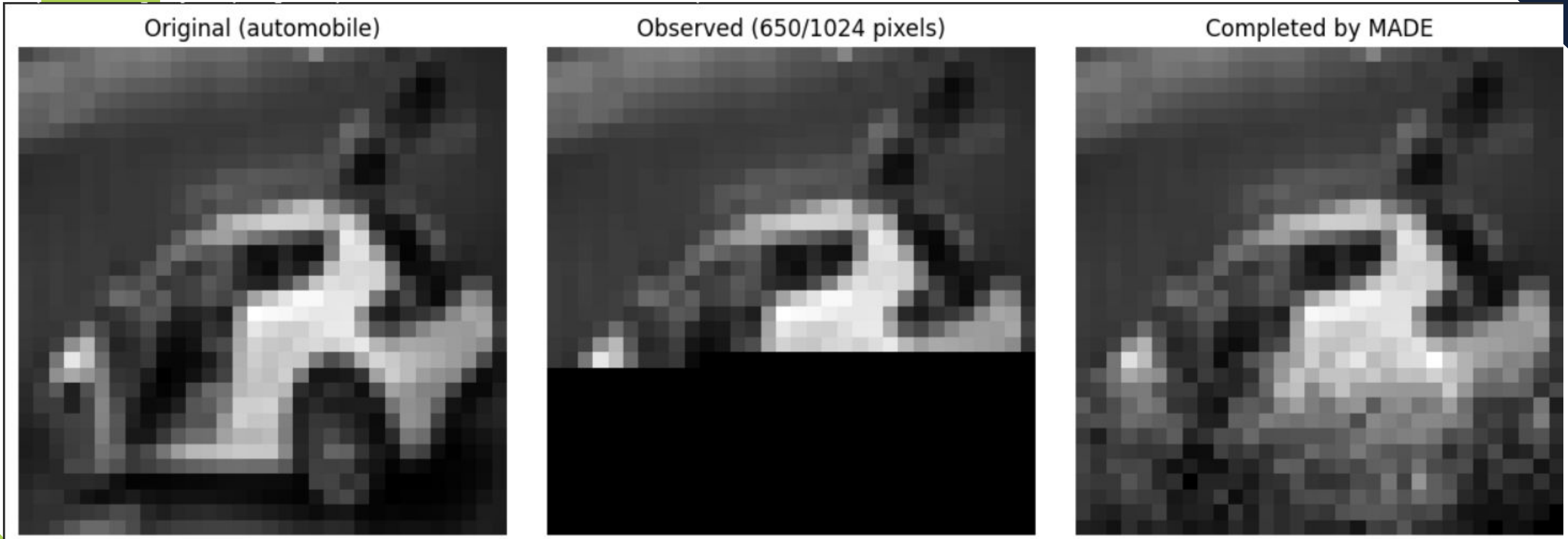
Binarized Mnist



MNIST Grayscale



CIFAR10 Grayscale



03

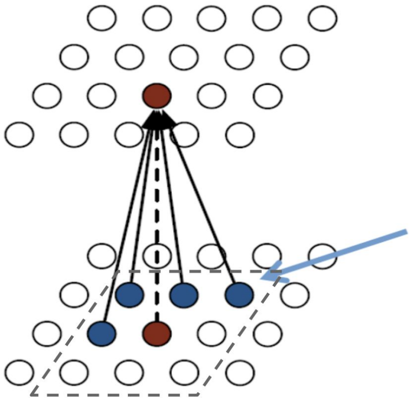
PixelCNN

03 PixelCNN

PixelCNN uses **convolutional neural networks** with masked convolutions to ensure that the prediction of each pixel depends only on previous pixels

It defines a **binary mask** over the convolution kernel so that the receptive field of the convolution only includes **“past” pixels** (above or to the left)

Mask A prevents a pixel from seeing itself in the first layer, while Mask B allows self-conditioning through previous-layer activations in deeper layers



Masked convolution

1	1	1
1	0	0
0	0	0

1	1	1	1	1
1	1	1	1	1
1	1	0	0	0
0	0	0	0	0
0	0	0	0	0

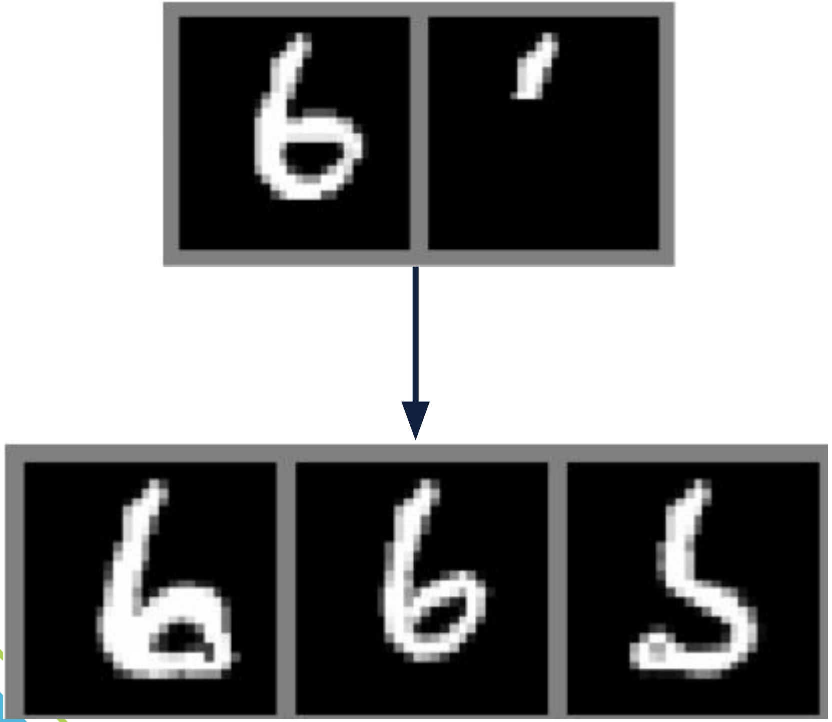
(a) mask A

1	1	1	1	1
1	1	1	1	1
1	1	1	0	0
0	0	0	0	0
0	0	0	0	0

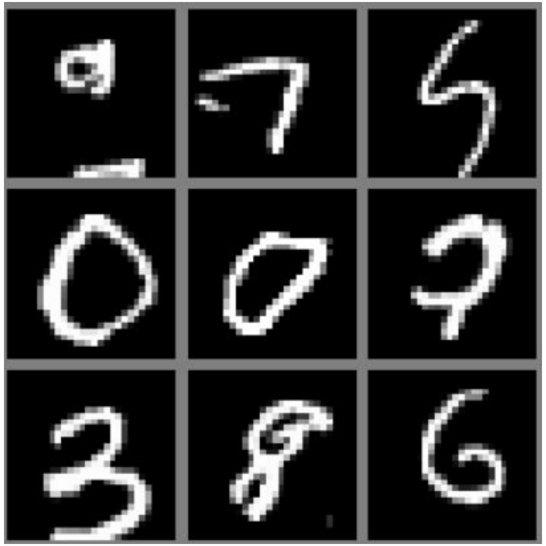
(b) mask B

MNIST

Autocompletion

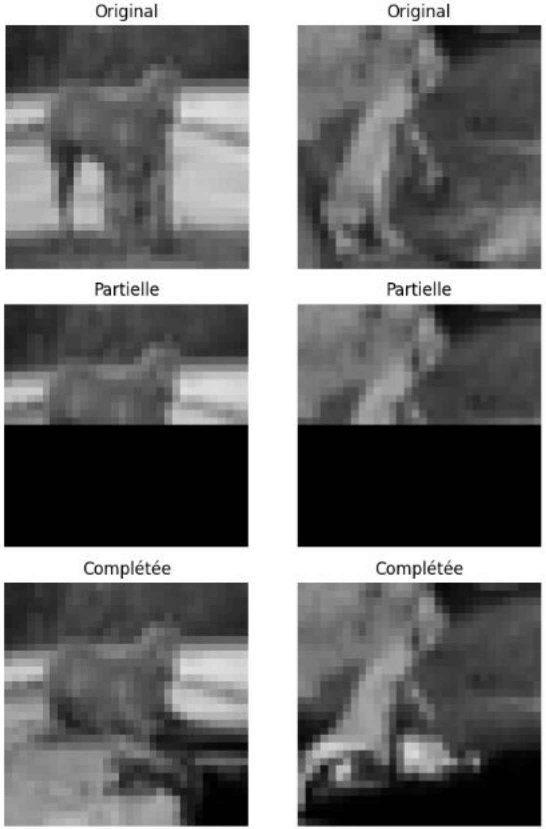


Sampling

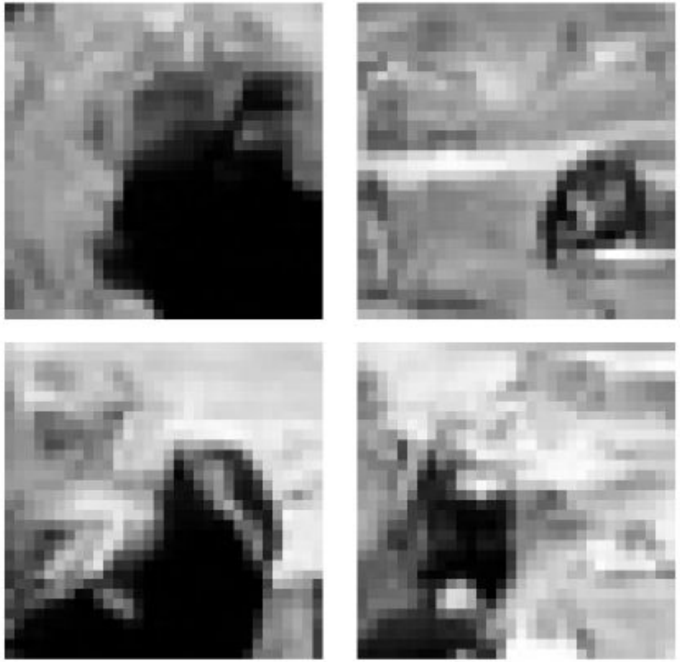


CIFAR10 Grayscale

Autocompletion



Sampling



04

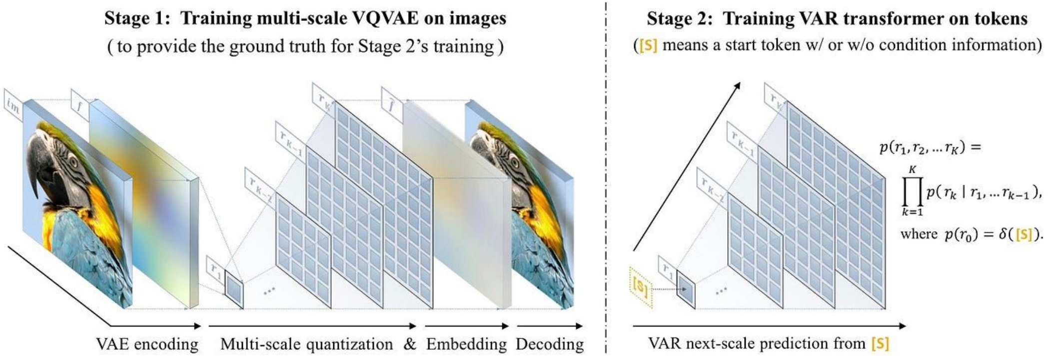
VAR

04 VAR

VAR (**Visual Autoregressive Modeling**) redefines the “generation order” by using a **coarse-to-fine**, “**next-scale**” strategy: generate low-resolution (coarse) token maps first, then progressively generate higher-resolution (finer) token maps, until reaching full resolution

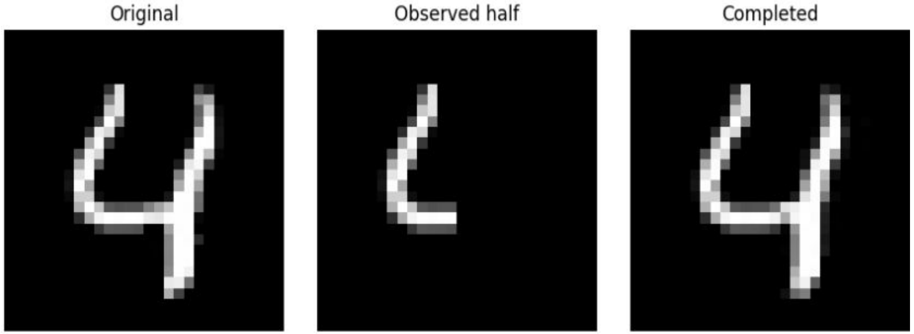
If we denote by T1,T2,…,TS the token maps at increasing scales (from coarse to fine), then VAR defines the joint probability of the full image tokens as:

$$p(T_1, T_2, \ldots, T_S) = p(T_1) \prod_{s=2}^{S} p(T_s \mid T_1, \ldots, T_{s-1})$$



MNIST

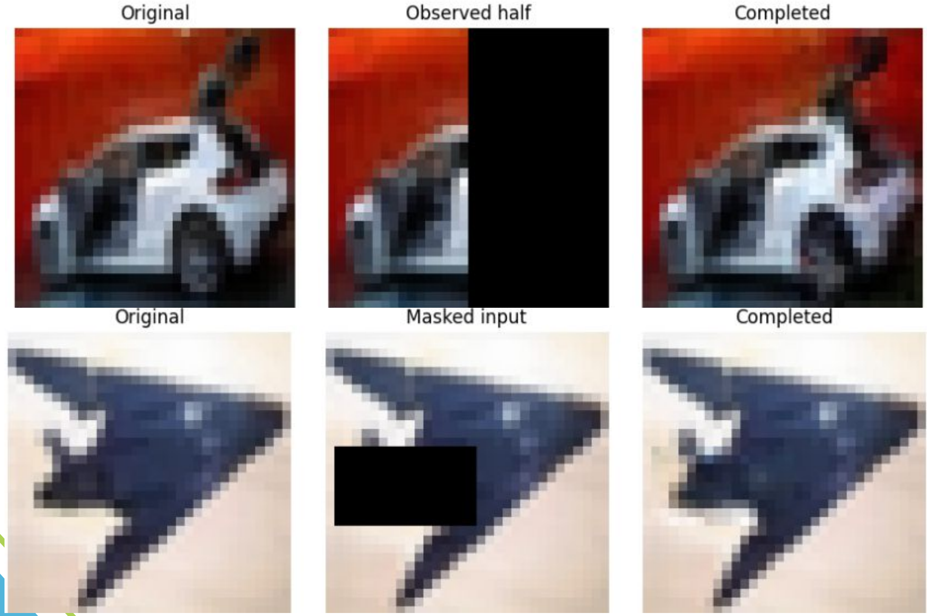
Autocompletion



Sampling

CIFAR10

Autocompletion



Sampling

05

Results

MADE

FID with respect to Number of Hidden Layers (here 500 neurons per layer)

Model / Hidden Layers	1	2	3
MNIST Binarized	53	58	64
MNIST Grayscale	270	274	313
Cifar10 Grayscale	258	262	271

Article loss results

MADE 1hl (1 mask)	88.40
MADE 2hl (1 mask)	89.59

Number of parameters ~ 0.8 M (CIFAR10)

MADE

FID with respect to Number of neurons (1 hidden layer here)

Model / nb neurons	500	1000	2000
MNIST Binarized	53	33	20
MNIST Grayscale	270	263	N/A
Cifar10 Grayscale	258	257	254

MADE

FID with respect to to distribution

Model / nb neurons	500	1000
MNIST Grayscale **Softmax**	172	165
MNIST Grayscale **Gaussian**	270	263

PixelCNN

CIFAR10 Grayscale	170
CIFAR10 Grayscale with added layers	166
MNIST	27

Number of parameters ~ 1.5 M (CIFAR 10)

VAR - VQVAE

CIFAR10 Grayscale	194
MNIST	102

Number of parameters ~ 4M (CIFAR 10)

06

Conclusion

Bibliography

[1] Mathieu Germain et al. (2015) “MADE: Masked Autoencoder for Distribution Estimation “ https://arxiv.org/abs/1502.03509

[2] Aaron van den Oord et al.(2016) “Conditional Image Generation with PixelCNN Decoders“ https://arxiv.org/abs/1606.05328

[3] Keyu Tian et al.(2024) “Visual Autoregressive Modeling: Scalable Image Generation via Next-Scale Prediction“ https://arxiv.org/abs/2404.02905

Generative AI Use

We used perplexity for bibliography search and to explain different parts in articles .
We used copilot with claude sonnet to implement Var model and to make some changes in other models (like fid calculation)