



# Inverse-Flow: Parallel Backpropagation for Inverse of a Convolution with Application to Normalizing Flows

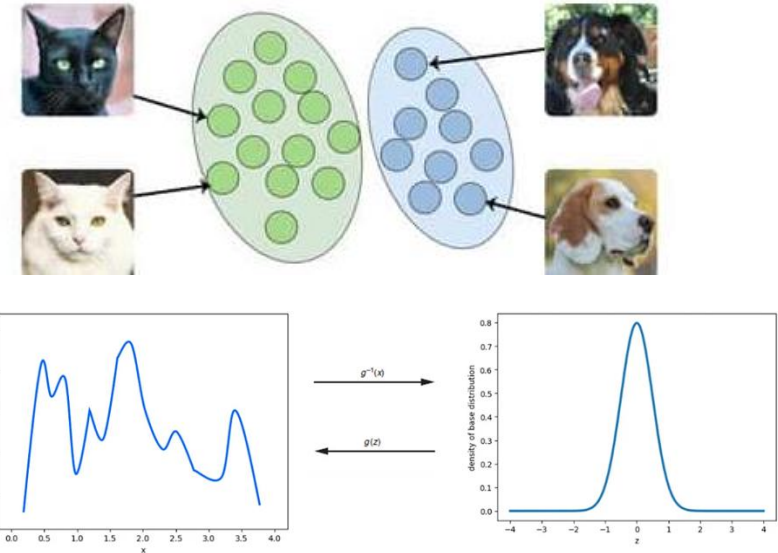
Sandeep Nagar, Girish Varma

MLL and C-STAR

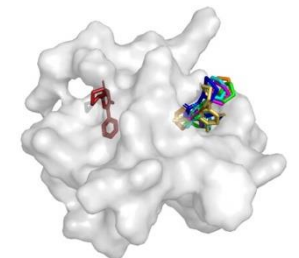
International Institute of Information Technology Hyderabad

# Generative Models

- Learn the underlying probability distribution of the dataset.
- Generate new, previously unseen samples that fit same distribution.
- Generates realistic samples.
- Applications:
  - Images,
  - Audio,
  - Text,
  - Drug design.

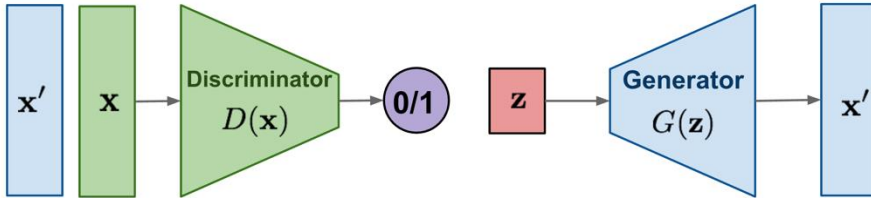


“a corgi  
playing a  
flame  
throwing  
trumpet”

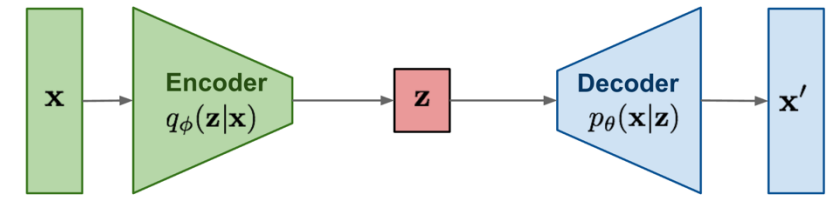


# Generative Models<sup>1</sup>

**GAN:** Adversarial training



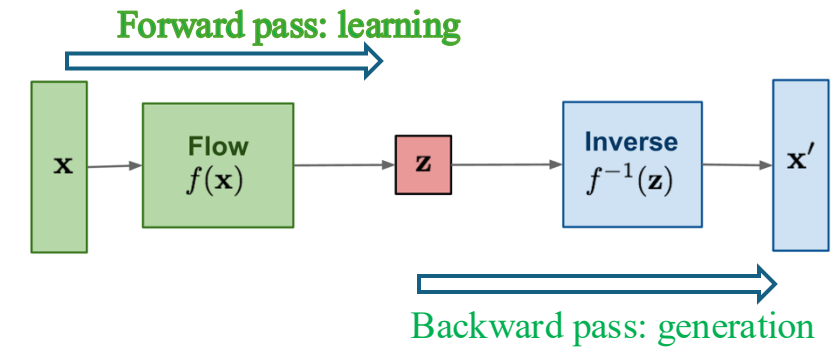
**VAE:** maximize variational lower bound



**Diffusion models:**  
Gradually add Gaussian noise and then reverse



**Flow-based models:**  
Invertible transform of distributions



<sup>1</sup>Bond-Taylor, Sam, Adam Leach, Yang Long, and Chris G. Willcocks. "Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models." *IEEE transactions on pattern analysis and machine intelligence* (2021).

# Designing Fast and Invertible Normalizing Flow Models

A quote from a famous statistician, **Dr. Goerge Box**:

*“All models are wrong, but some are useful.”*

## Why NF?

### Other Generative Models<sup>1</sup>:

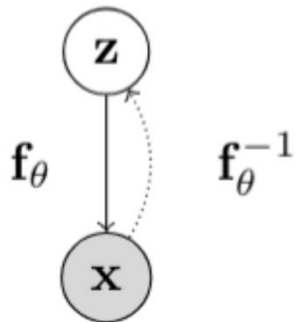
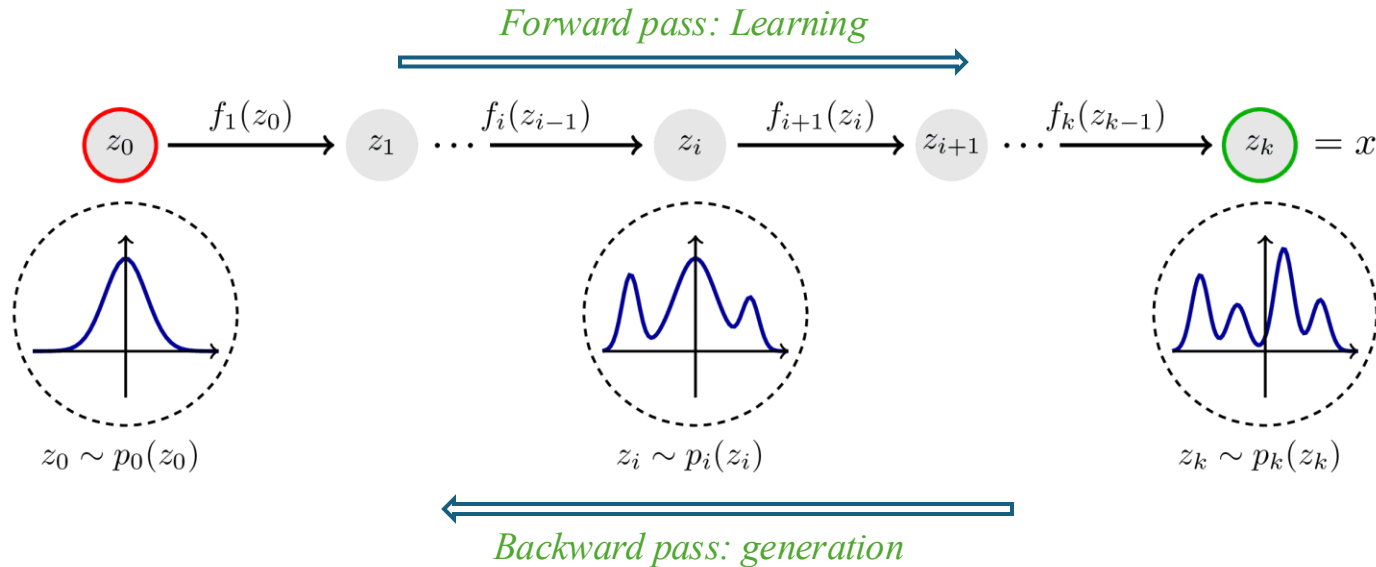
- Approximate models.
- GANs: Optimization can be **challenging and unstable**.
- VAEs: **Blurry samples** and **struggle capturing** complex data.
- AR: Generate samples **sequentially**.
- EBMs: **High variance training**, **Slow training**, and **sampling**.

### Normalizing flow Models<sup>1</sup>:

- Probabilistic models,
- High-dimensional spaces,
- Combinatorial spaces,
- Tractable
- One-to-one mapping.
- Latent space exploration (z).
- **Need Fast Invertible Function,  $y = f(x)$**
- **Intuitive Maximum Likelihood loss function**

<sup>1</sup>Bond-Taylor, Sam, Adam Leach, Yang Long, and Chris G. Willcocks. "Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models." *IEEE transactions on pattern analysis and machine intelligence* (2021).

# Normalizing Flows Models



- Coupling layers:
- Autoregressive Flows:
- PDF, ODE

$$\mathbf{x}_K = f_K \circ \dots \circ f_2 \circ f_1(\mathbf{x}_0),$$

$$\ln p_K(\mathbf{x}_K) = \ln p_0(\mathbf{x}_0) - \sum_{k=1}^K \ln \left| \det \frac{\partial f_k}{\partial \mathbf{x}_{k-1}} \right|.$$

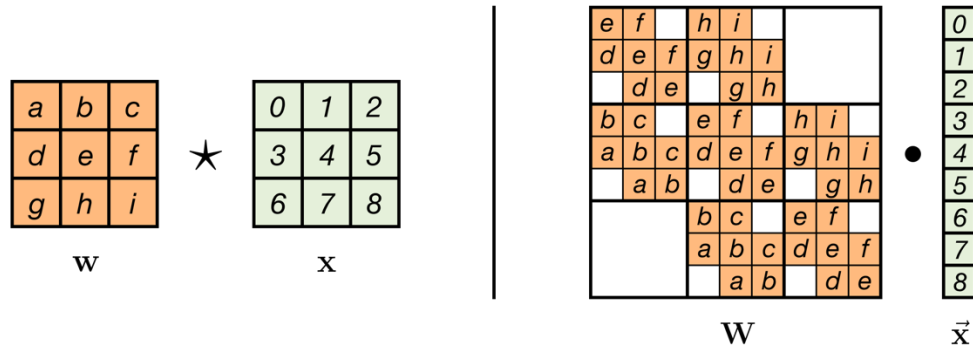
Problem flow models:

- Restricted triangular Jacobian,
- All inputs cannot interact with each other.

<sup>1</sup>Bond-Taylor, Sam, Adam Leach, Yang Long, and Chris G. Willcocks. "Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models." *IEEE transactions on pattern analysis and machine intelligence* (2021).

# Complexity of finding the Inverse of Convolution

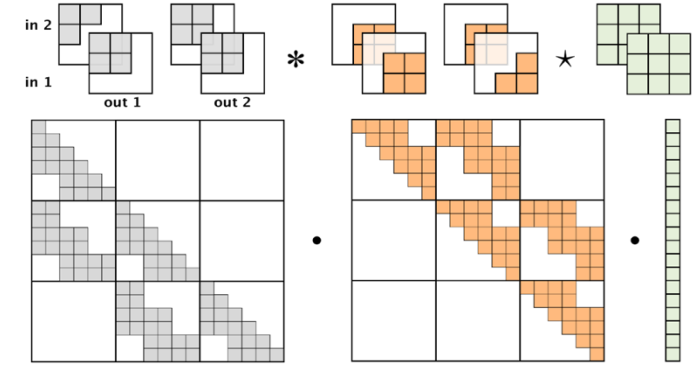
Std. Convolution:



Gaussian Elimination  $O(n^4)$

$n$ : input size  
 $k$ : inv kernel size

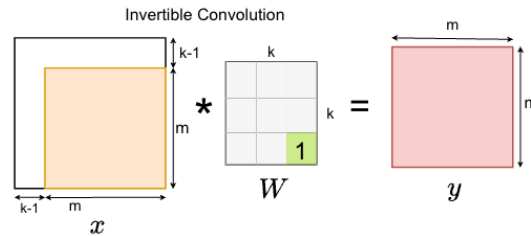
Emerging Convolution<sup>1</sup>:



QR decomposition  $O(n^3)$

E. Hoogeboom et. al. ICML'19

CInC Convolution<sup>2</sup>(our):



$$\begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix} * \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} = \begin{bmatrix} w_{22} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ w_{21} & w_{22} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & w_{21} & w_{22} & 0 & 0 & 0 & 0 & 0 & 0 \\ w_{12} & 0 & 0 & w_{22} & 0 & 0 & 0 & 0 & 0 \\ w_{11} & w_{12} & 0 & w_{21} & w_{22} & 0 & 0 & 0 & 0 \\ 0 & w_{11} & w_{12} & 0 & w_{21} & w_{22} & 0 & 0 & 0 \\ 0 & 0 & 0 & w_{12} & 0 & 0 & w_{22} & 0 & 0 \\ 0 & 0 & 0 & w_{11} & w_{12} & 0 & w_{21} & w_{22} & 0 \\ 0 & 0 & 0 & 0 & w_{11} & w_{12} & 0 & w_{21} & w_{22} \end{bmatrix} * \begin{bmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{31} \\ x_{32} \\ x_{33} \end{bmatrix} = \begin{bmatrix} y_{11} \\ y_{12} \\ y_{13} \\ y_{21} \\ y_{22} \\ y_{23} \\ y_{31} \\ y_{32} \\ y_{33} \end{bmatrix}$$

Back Substitution  $O(n^2)$

FInC Convolution<sup>4</sup>(our):

$$\begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix} * \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} = \begin{bmatrix} w_{22} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ w_{21} & w_{22} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & w_{21} & w_{22} & 0 & 0 & 0 & 0 & 0 & 0 \\ w_{12} & 0 & 0 & w_{22} & 0 & 0 & 0 & 0 & 0 \\ w_{11} & w_{12} & 0 & w_{21} & w_{22} & 0 & 0 & 0 & 0 \\ 0 & w_{11} & w_{12} & 0 & w_{21} & w_{22} & 0 & 0 & 0 \\ 0 & 0 & 0 & w_{12} & 0 & 0 & w_{22} & 0 & 0 \\ 0 & 0 & 0 & w_{11} & w_{12} & 0 & w_{21} & w_{22} & 0 \\ 0 & 0 & 0 & 0 & w_{11} & w_{12} & 0 & w_{21} & w_{22} \end{bmatrix} * \begin{bmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{31} \\ x_{32} \\ x_{33} \end{bmatrix} = \begin{bmatrix} y_{11} \\ y_{12} \\ y_{13} \\ y_{21} \\ y_{22} \\ y_{23} \\ y_{31} \\ y_{32} \\ y_{33} \end{bmatrix}$$

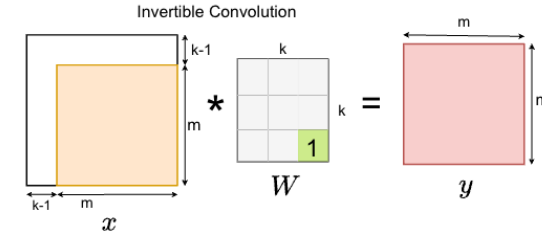
Back Substitution and Parallel  $O(k^2n)$

$k$ : input size

# Backpropagation algorithm for the Inverse of Convolution Layer

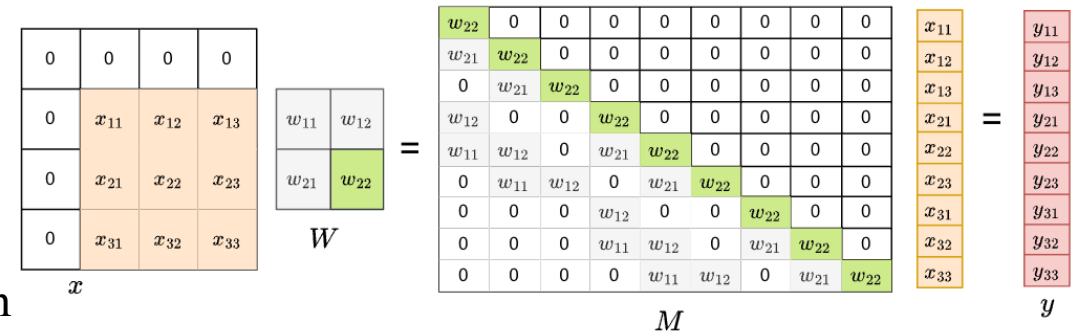
For training:  $x = y * W'$  Inverse of Std Conv.

For sampling:  $y = x * W$  Std. Convolution



where  $w' = \text{inv}(W)$ .

- Advantages:
- Independent of the size (m)
  - Fast sampling
  - Backprop algorithm for the Inverse of Convolution



Inverse of Convolution layer:

$$y_p = x_p + \sum_{q \in \Delta(p)} W_{(k,k)-p+q} \cdot x_q \quad (1)$$

Using chain rule of differentiation, we get that

$$\frac{\partial L}{\partial y_p} = \sum_q \frac{\partial L}{\partial x_q} \times \frac{\partial x_q}{\partial y_p} \quad (2)$$

L is the loss.

**Theorem 1.**

$$\frac{\partial x_q}{\partial y_p} = \begin{cases} 1 - \sum_{q \in \Delta(p)} W_{(k,k)-p+q} \cdot \frac{\partial x_q}{\partial y_p} & \text{if } p = q \\ 0 & \text{if } q \notin p \\ - \sum_{r \in \Delta(p)} W_{(k,k)-r} \frac{\partial x_{p-r'}}{\partial y_p} & \text{otherwise.} \end{cases}$$

**Theorem 2.**

$$\frac{\partial x_q}{\partial W_a} = \begin{cases} 0 & \text{if } q \leq a \\ - \sum_{q' \in \Delta_q(a)} W_{q'-a} \cdot \frac{\partial x_{q-q'}}{\partial W_a} - x_{q-a} & \text{if } q > a \end{cases}$$

# Inverse-Flow: Inverse of Convolution and its Backpropagation Algorithm<sup>1</sup>

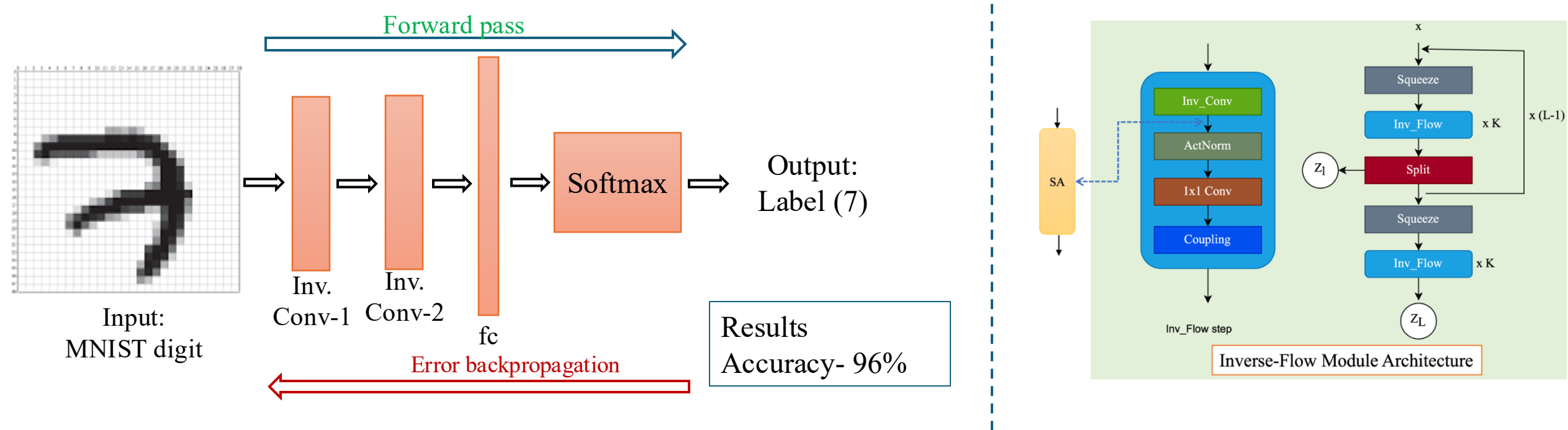
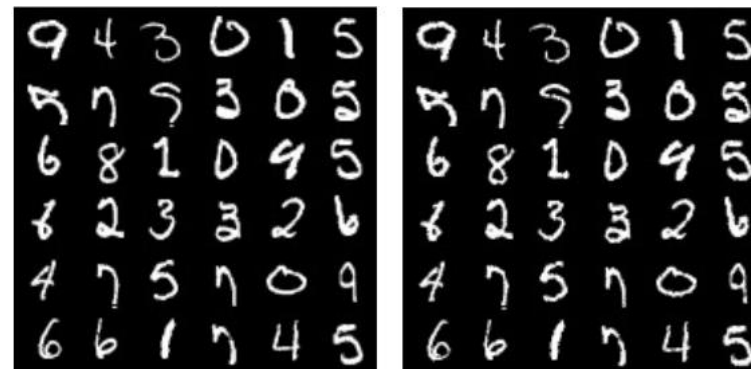
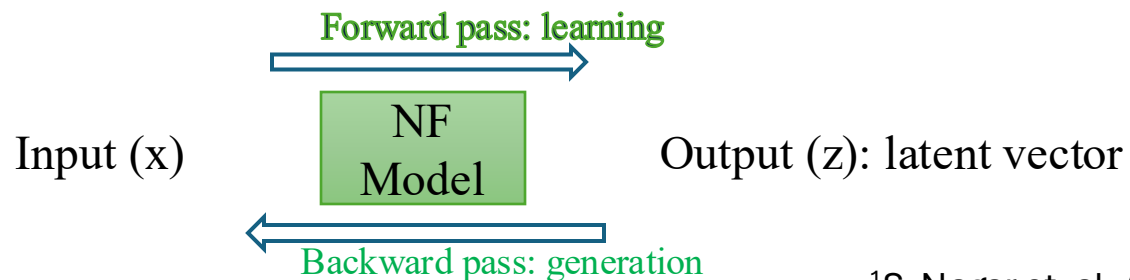


Image Generation using the above-proposed Inverse of Convolution Layers and Backpropagation Algorithm



<sup>1</sup>S. Nagar et. al. AISTATS'25

# Inverse-Flow: Inverse of Convolution and Backpropagation Algorithm

## Results:

Table 3: Performance comparison for MNIST with block size ( $K = 16$ ) and number of blocks ( $L = 2$ ).

Method	ST	NLL	BPD	Param	Inverse
SNF	99 $\pm$ 2.1	699	1.28	10.1	approx
FIncFlow	90 $\pm$ 2.2	655	1.15	10.2	exact
MintNet	320 $\pm$ 2.8	630	0.98	125.9	approx
Emerging	814 $\pm$ 6.2	640	1.09	11.4	exact
→ Inverse-Flow	52 $\pm$ 1.3	710	1.31	1.6	exact

Table 6: Performance comparison for CIFAR dataset with block size ( $K = 16$ ) and number of blocks ( $L = 2$ ). SNF uses approx for inverse, and MintNet uses autoregressive functions. \*time in seconds.

Method	BPD	ST	FT	Param
SNF	3.52	16.8 $\pm$ 2.7	609 $\pm$ 5.4	1.682
MintNet	3.51	25.0* $\pm$ 1.5	2458 $\pm$ 6.2	12.466
Woodbury	3.48	7654.4 $\pm$ 13.5	119 $\pm$ 2.5	12.49
MaCow	3.40	790.8 $\pm$ 4.3	1080 $\pm$ 6.6	2.68
CInC Flow	3.46	1710.0 $\pm$ 9.5	615 $\pm$ 5.0	2.62
Butterfly Flow	3.39	311.8 $\pm$ 4.0	1325 $\pm$ 7.5	12.58
FInc Flow	3.59	194.8 $\pm$ 2.5	548 $\pm$ 6.2	2.72
→ Inverse-Flow	3.57	91.6 $\pm$ 6.5	722 $\pm$ 7.0	1.76

## Ablation Study:

input size	Sampling Time (ST)	Forward Time (FT)	GPU memory (GB)
256x256	16.54 $\pm$ 0.21	427 $\pm$ 20	8.025
128x128	16.22 $\pm$ 0.23	342 $\pm$ 15	3.822
64x64	15.94 $\pm$ 0.14	264 $\pm$ 11	1.316
32x32	15.53 $\pm$ 0.29	224 $\pm$ 11	0.375
16x16	15.55 $\pm$ 0.12	211 $\pm$ 10	0.131

Table 7: Inverse-Flow:  $K = 2$ ,  $L = 32$ , Sample size = 100, batch size = 100. All times in milliseconds

batch size	ST	FT	memory (GB)
128	19.65 $\pm$ 0.19	487 $\pm$ 20	6.383
64	18.99 $\pm$ 0.22	349 $\pm$ 22	3.361
32	18.83 $\pm$ 0.12	339 $\pm$ 16	1.809
16	18.48 $\pm$ 0.10	335 $\pm$ 09	1.037
8	18.74 $\pm$ 0.15	345 $\pm$ 11	0.654
4	18.04 $\pm$ 0.40	333 $\pm$ 21	0.457
2	17.71 $\pm$ 0.12	258 $\pm$ 11	0.379
1	16.22 $\pm$ 0.19	235 $\pm$ 10	0.328

Table 9: Inverse Flow:  $K = 2$ ,  $L=32$ . Sample size =100, batch size = 100. For 3x256x256, one NVIDIA GeForce GTX 1080 Ti GPU goes out of memory. All times in milliseconds

kernel size	Sampling Time (ST)	Forward Time (FT)	Params (M)
11x11	17.87 $\pm$ 0.42	2428 $\pm$ 28	6.068
9x9	17.36 $\pm$ 0.22	1762 $\pm$ 30	5.146
7x7	19.20 $\pm$ 0.10	1461 $\pm$ 23	4.407
5x5	20.79 $\pm$ 0.15	934 $\pm$ 17	3.856
3x3	18.11 $\pm$ 0.20	402 $\pm$ 08	3.487
2x2	17.90 $\pm$ 0.25	364 $\pm$ 07	3.372

Table 10: Inverse Flow model size and sampling time for different kernel sizes: Model Arch,  $K = 2$ ,  $L=32$ . Sample size = 100, batch size = 100. All times in milliseconds



Questions?