Invertible Residual Networks

Jens Behrmann*
Will Grathwohl*
Ricky T. Q. Chen
David Duvenaud
Jörn-Henrik Jacobsen*

Universität Bremen
Zentrum für Technomathematik
VELUT ARBOR ÆVO
VECTOR INSTITUTE

(*equal contribution)

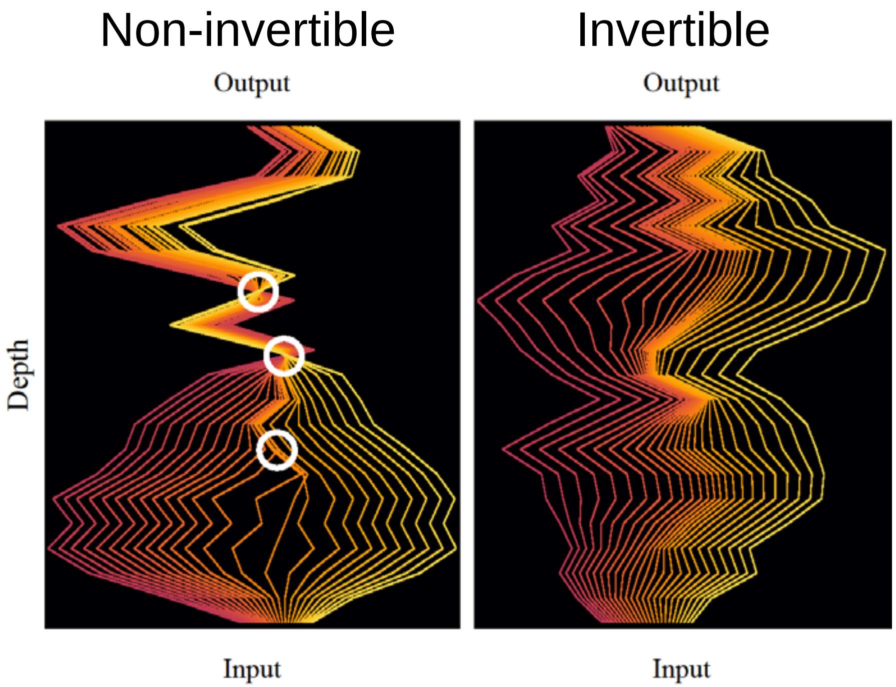
What are Invertible Neural Networks?

Invertible Neural Networks (INNs) are bijective function approximators which have a **forward mapping**

$$F_\theta : \mathbb{R}^d \to \mathbb{R}^d$$
$$x \mapsto z$$

and an **inverse mapping**

$$F_\theta^{-1} : \mathbb{R}^d \to \mathbb{R}^d$$
$$z \mapsto x$$



Why Invertible Networks?

- Mostly known because of Normalizing Flows
  - Training via maximum-likelihood and evaluation of likelihood

Generated samples from GLOW (Kingma et al. 2018)

Why Invertible Networks?

- Generative modeling via invertible mappings with exact likelihoods (Dinh et al. 2014, Dinh et a. 2016, Kingma et al. 2018, Ho et al. 2019)
  - Normalizing Flows
- Mutual information preservation

$$I(Y;X) = I(Y;F_\theta(X))$$

- Analysis and regularization of invariance (Jacobsen et al. 2019)
- Memory-efficient backprop (Gomez et al. 2017)
- Analyzing inverse problems (Ardizzone et al. 2019)

Workshop: Invertible Networks and Normalizing Flows

Invertible Networks use Exotic Architectures

- Dimension partitioning and coupling layers (Dinh et al. 2014/2016, Gomez et al. 2017, Jacobsen et al. 2018, Kingma et al. 2018)
  - Transforms one part of the input at a time
  - Choice of partitioning is important

Invertible Networks use Exotic Architectures

- Dimension partitioning and coupling layers (Dinh et al. 2014/2016, Gomez et al. 2017, Jacobsen et al. 2018, Kingma et al. 2018)
  - Transforms one part of the input at a time
  - Choice of partitioning is important
- Invertible dynamics via Neural ODEs (Chen et al. 2018, Grathwohl et al. 2019)
  - Requires numerical integration
  - Hard to tune and often slow due to need of ODE-solver

Why do we move away from standard architectures?

- Partitioning, coupling layers, ODE-based approaches move further away from standard architectures
  - Many new design choices necessary and not well understood yet

- Why not use most successful discriminative architecture?

ResNets

- Use connection of ResNet and Euler integration of ODEs (Haber et al. 2018)

Making ResNets invertible

Theorem (sufficient condition for invertible residual layer):

Let $F^t_\theta(x) = x + g^t_\theta(x)$ be a residual layer, then it is invertible if

$$\mathrm{Lip}(g^t_\theta) < 1$$

where

$$\|g(x) - g(y)\|_2 \leq \mathrm{Lip}(g)\|x - y\|_2$$

Making ResNets invertible

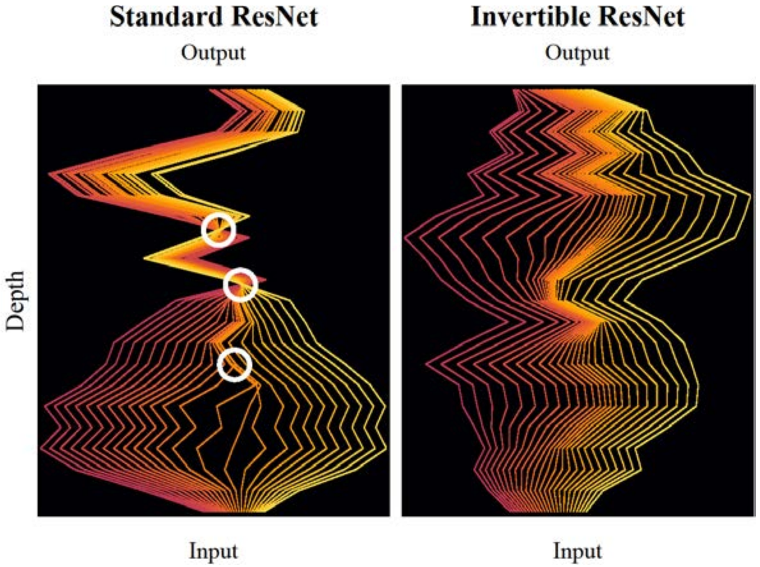
Theorem (sufficient condition for invertible residual layer):

Let $F_\theta^t(x) = x + g_\theta^t(x)$ be a residual layer, then it is invertible if

$$\mathrm{Lip}(g_\theta^t) < 1$$

where

$$\|g(x) - g(y)\|_2 \leq \mathrm{Lip}(g)\|x - y\|_2$$



Invertible Residual Networks (i-ResNet)

$$F_\theta = F_\theta^T \circ \cdots \circ F_\theta^1$$

i-ResNets: Constructive Proof

Theorem: (invertible residual layer)

Let $F(x) = x + g(x)$ be a residual layer, then it is invertible if

$$\mathrm{Lip}(g) < 1$$

Proof:

Features: $z := F(x)$

Fixed-point equation: $x = z - g(x)$

i-ResNets: Constructive Proof

Theorem: (invertible residual layer)

Let $F(x) = x + g(x)$ be a residual layer, then it is invertible if

$$\mathrm{Lip}(g) < 1$$

Proof:

Features: $z := F(x)$

Fixed-point equation: $x = z - g(x)$

→ Use fixed-point iteration:

$$x^{(0)} = z$$

$$x^{(i+1)} = z - g(x^{(i)})$$

i-ResNets: Constructive Proof

Theorem: (invertible residual layer)

Let $F(x) = x + g(x)$ be a residual layer, then it is invertible if

$$\mathrm{Lip}(g) < 1$$

Proof:

Features: $z := F(x)$

Fixed-point equation: $x = z - g(x)$

→ Use fixed-point iteration:

$$x^{(0)} = z$$

$$x^{(i+1)} = z - g(x^{(i)})$$

→ Guaranteed convergence to x if g contractive (Banach fixed-point theorem)

Inverting i-ResNets

- Inversion method from proof
- Fixed-point iteration:
  - Init:

$$x^{(0)} = z$$

  - Iteration:

$$x^{(i+1)} = z - g(x^{(i)})$$

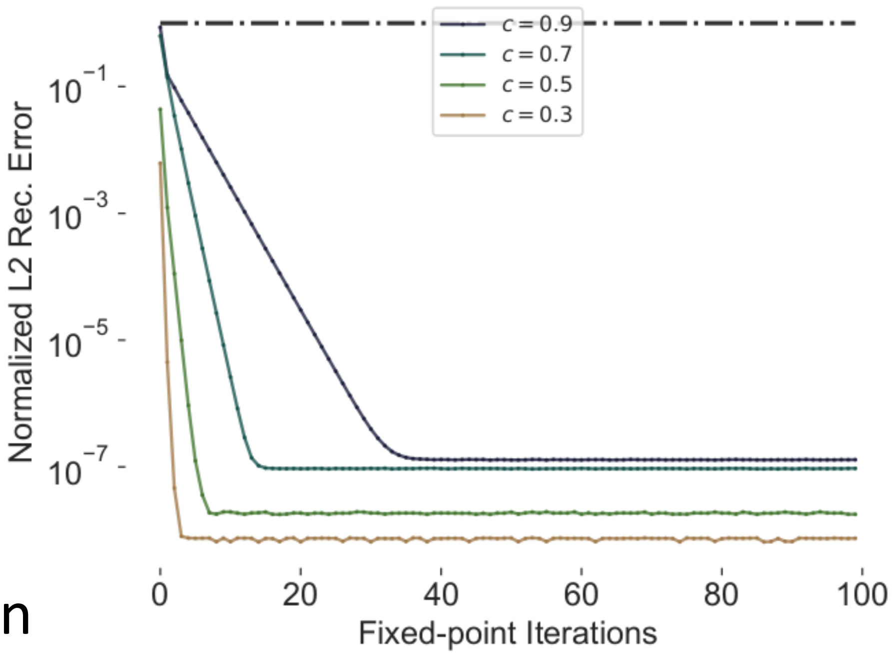
Inverting i-ResNets

- Inversion method from proof
- Fixed-point iteration:
  - Init:

$$x^{(0)} = z$$

  - Iteration:

$$x^{(i+1)} = z - g(x^{(i)})$$



- Rate of convergence depends on Lipschitz constant
- In practice: cost of inverse is 5-10 forward passes

How to build i-ResNets

- Satisfy Lip-condition: data-independent upper bound

$$g = W_3 \circ \phi \circ W_2 \circ \phi \circ W_1 \circ \phi$$

$$\mathrm{Lip}(g) \leq \|W_3\|_2 \cdot \|W_2\|_2 \cdot \|W_1\|_2$$

How to build i-ResNets

- Satisfy Lip-condition: data-independent upper bound

$$g = W_3 \circ \phi \circ W_2 \circ \phi \circ W_1 \circ \phi$$

$$\mathrm{Lip}(g) \leq \|W_3\|_2 \cdot \|W_2\|_2 \cdot \|W_1\|_2$$

- Spectral normalization (Miyato et al. 2018, Gouk et al. 2018)

$$\tilde{W} = c\frac{W}{\hat{\sigma_1}}, \quad 0 < c < 1$$

$\hat{\sigma_1}$ approx of largest singular value via power-iteration

How to build i-ResNets

- Satisfy Lip-condition: data-independent upper bound

$$g = W_3 \circ \phi \circ W_2 \circ \phi \circ W_1 \circ \phi$$

$$\mathrm{Lip}(g) \leq \|W_3\|_2 \cdot \|W_2\|_2 \cdot \|W_1\|_2$$

- Spectral normalization (Miyato et al. 2018, Gouk et al. 2018)

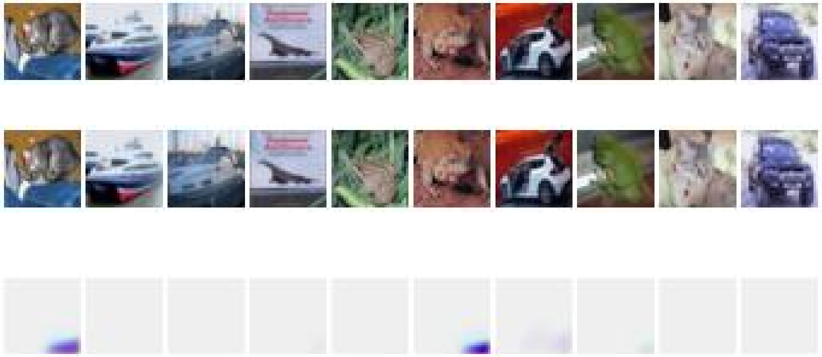
$$\tilde{W} = c\frac{W}{\hat{\sigma_1}}, \quad 0 < c < 1$$

$\hat{\sigma_1}$ approx of largest singular value via power-iteration

```
def invertible_residual_block(self):
    layers = []
    layers.append(nn.ReLU)
    layers.append(spectral_norm(nn.Linear(in_dim, hidden_dim)))
    layers.append(nn.ReLU)
    layers.append(spectral_norm(nn.Linear(hidden_dim, in_dim)))
```

Validation

- Reconstructions



CIFAR10 Data

Reconstructions: i-ResNet

Reconstructions: standard ResNet

Classification Performance

		ResNet-164	Vanilla	$c = 0.9$
Classification Error %	MNIST	-	0.38	0.40
	CIFAR10	5.50	6.69	6.78
	CIFAR100	24.30	23.97	24.58
Guaranteed Inverse		**No**	**No**	**Yes**

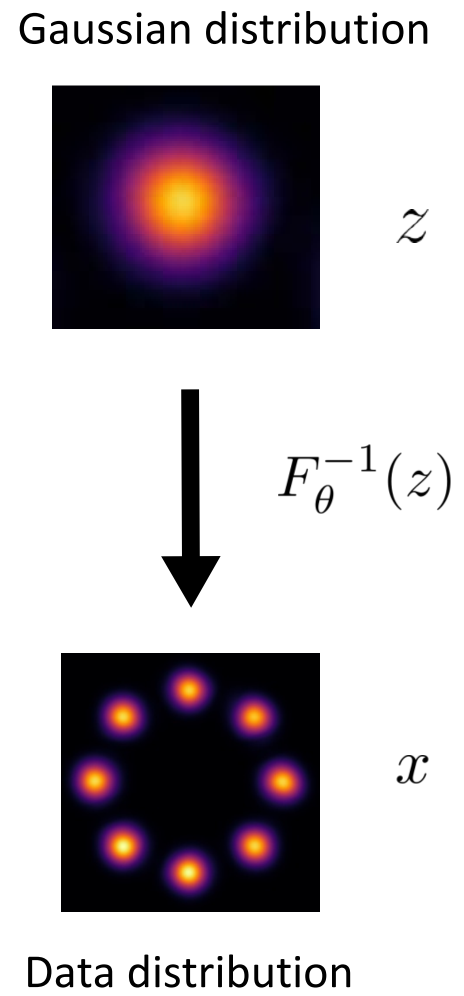
- Competetive performance
- But what do we get additionally?

Generative models via Normalizing Flows

Maximum-Likelihood Generative Modeling with i-ResNets

- We can define a simple generative model as

$$z \sim p_Z(z)$$
$$x = F_\theta^{-1}(z)$$



Maximum-Likelihood Generative Modeling with i-ResNets

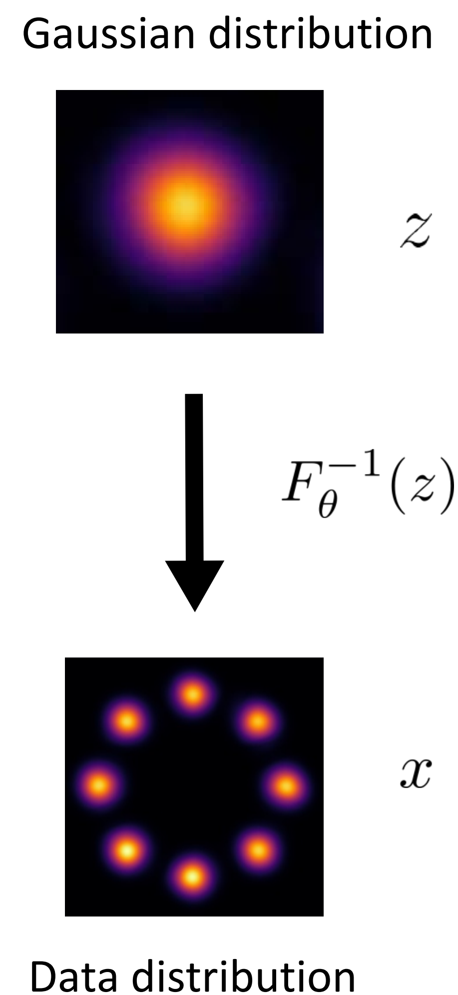
- We can define a simple generative model as

$$z \sim p_Z(z)$$
$$x = F_\theta^{-1}(z)$$

- Maximization (and evaluation) of likelihood via change-of-variables

$$\log p_X(x) = \log p_Z(F_\theta(x)) + \log|\det J_{F_\theta}(x)|$$

... if F_θ is invertible



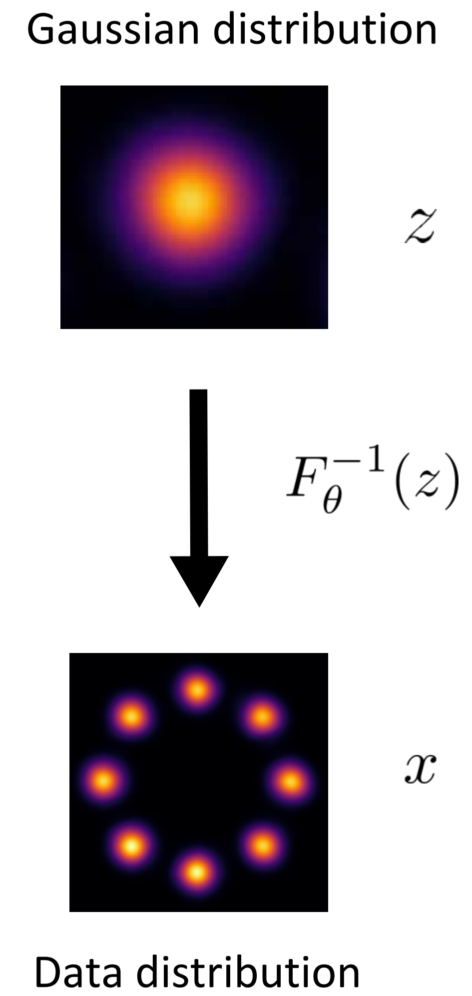
Maximum-Likelihood Generative Modeling with i-ResNets

- Maximization (and evaluation) of likelihood via change-of-variables

$$\log p_X(x) = \log p_Z(F_\theta(x)) + \log|\det J_{F_\theta}(x)|$$

... if F_θ is invertible

- Challenges:
  - Flexible invertible models
  - Efficient computation of log-determinant



Efficient Estimation of Likelihood

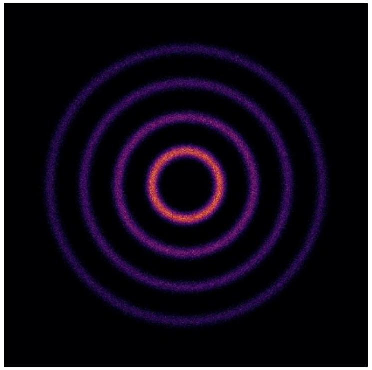
- Likelihood with log-determinant of Jacobian

$$\log p_X(x) = \log p_Z(F_\theta(x)) + \log |\det J_{F_\theta}(x)|$$

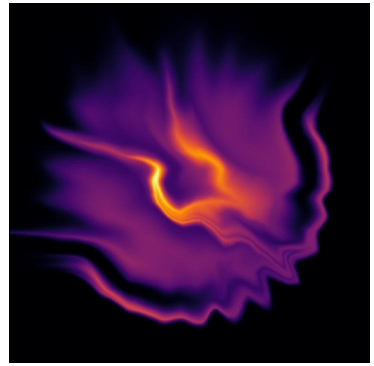
- Previous approaches:
  - exact computation of log-determinant via constraining architecture to be triangular
    (Dinh et al. 2016, Kingma et al. 2018)
  - ODE-solver and estimation only of trace of Jacobian
    (Grathwohl et al. 2019)

- We propose an **efficient estimator for i-ResNets** based on trace-estimation and truncation of a power series

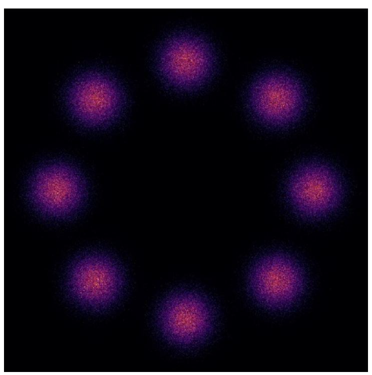
Generative Modeling Results

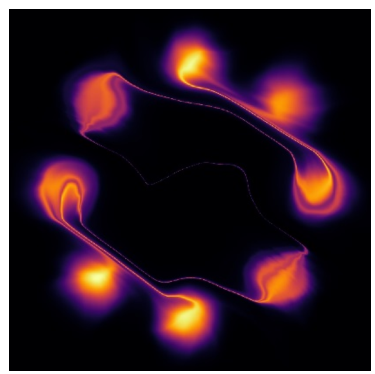


Data Samples

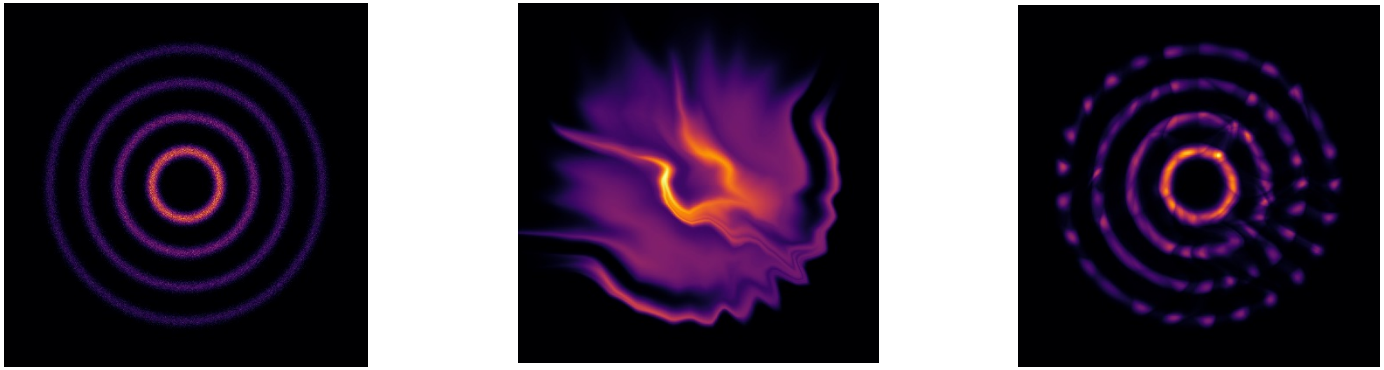


GLOW

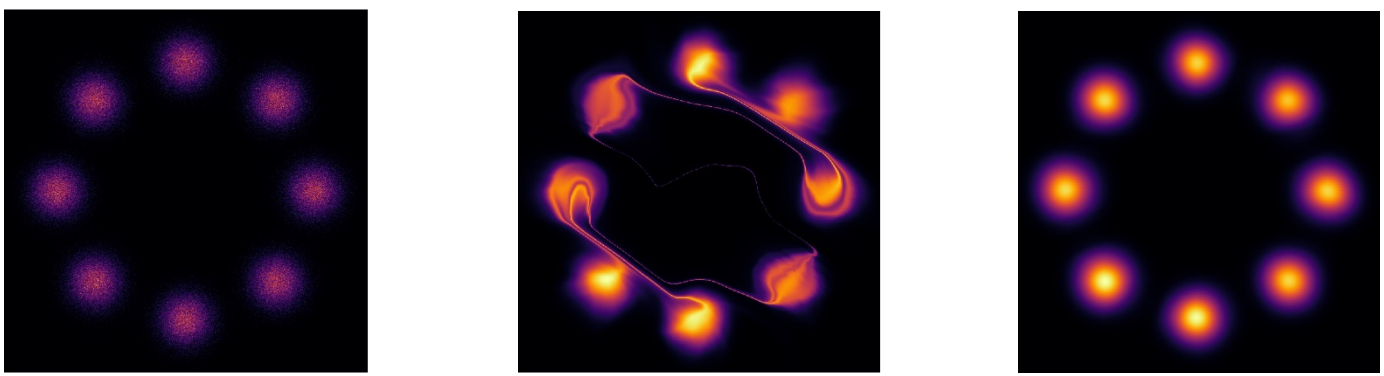




Generative Modeling Results

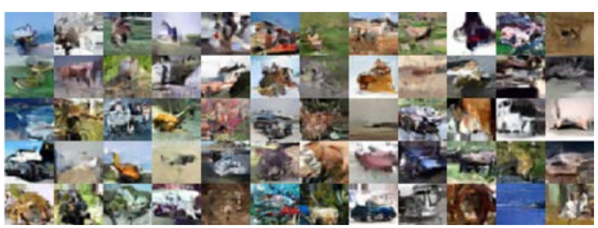


Data Samples　　GLOW　　i-ResNets



Generative Modeling Results

Method	MNIST	CIFAR10
NICE (Dinh et al., 2014)	4.36	4.48†
MADE (Germain et al., 2015)	2.04	5.67
MAF (Papamakarios et al., 2017)	1.89	4.31
Real NVP (Dinh et al., 2017)	1.06	3.49
Glow (Kingma & Dhariwal, 2018)	1.05	3.35
FFJORD (Grathwohl et al., 2019)	0.99	3.40
i-ResNet	1.06	3.45



GLOW (Kingma et al. 2018)

FFJORD (Grathwohl et al. 2019)

i-ResNet

i-ResNets Across Tasks

- i-ResNet as an architecture which **works well both in discriminative and generative modeling**

Affine Glow 1×1 Conv	Additive Glow Reverse	i-ResNet Glow-Style	i-ResNet 164
12.63	12.36	8.03	6.69

- i-ResNets are generative models which use the best discriminative architecture
- Promising for:
  - Unsupervised pre-training
  - Semi-supervised learning

Drawbacks

- Iterative inverse
  - Fast convergence in practice
  - Rate depends on Lip-constant and not on dimension
- Requires estimation of log-determinant
  - Due to free-form of Jacobian
  - Properties of i-ResNets allows to design efficient estimator

Conclusion

- Simple modification makes ResNets invertible
- Stability is guaranteed by construction

- New class of likelihood-based generative models
  - without structural constraints
- Excellent performance in discriminative/ generative tasks
  - with one unified architecture

- Promising approach for:
  - unsupervised pre-training
  - semi-supervised learning
  - tasks which require invertibility

See us at Poster #11 (Pacific Ballroom)

Paper:

Code:

Follow-up work:

Residual Flows for Invertible Generative Modeling

Invertible Networks and Normalizing Flows, workshop on Saturday (contributed talk)