August 28, 2020

Mila

Video prediction and image generation using dynamical modeling

Vikram Voleti

PhD student, Mila, University of Montreal

Supervisor: **Prof. Christopher Pal**

voletiv.github.io

@ (virtual) Mila, Montreal, Canada

1. Neural ODEs

2. Video Prediction
3. Image Generation
4. Mutual Interaction Minimization

1. Neural ODEs

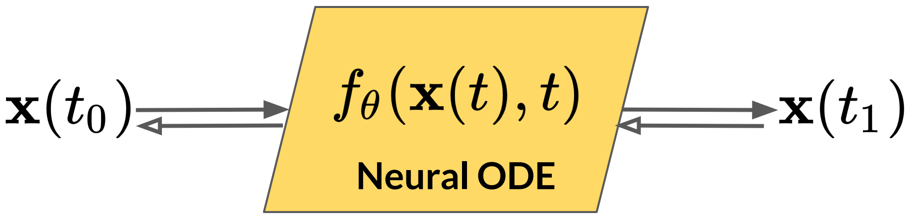
Initial value problem:

$$\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} = f_\theta(\mathbf{x}(t), t); \quad \mathbf{x}(t_0) \text{ is given}; \quad \mathbf{x}(t_1) = \ ?$$

f_θ is a neural network!

Solution:

$$(\ \mathbf{x}(t_1), \mathbf{x}(t_2), \ldots, \mathbf{x}(t_n)\ ) = \mathrm{ODESolve}(\ f_\theta(\mathbf{x}(t), t),\ \mathbf{x}(t_0),\ t_0,\ (t_1, t_2, \ldots, t_n)\ )$$



Neural ODEs are **reversible** models!

https://arxiv.org/abs/1806.07366

Adjoint method (Pontryagin et al., 1962)

Forward propagation:

$$\mathbf{x}(t_1) = \text{ODESolve}(\ f_\theta(\mathbf{x}(t), t),\ \mathbf{x}(t_0),\ t_0,\ t_1\ )$$

Compute $L(\mathbf{x}(t_1))$.

$$\mathbf{a}(t_1) = \frac{\partial L}{\partial \mathbf{x}(t_1)}$$

$$\boxed{\frac{d\mathbf{a}}{dt} = -\mathbf{a}(t)^\top \frac{\partial f(\mathbf{x}(t), t, \theta)}{\partial \mathbf{x}}}$$

Back-propagation:

$$\begin{bmatrix} \mathbf{x}(t_0) \\ \frac{\partial L}{\partial \mathbf{x}(t_0)} \\ \boxed{\frac{\partial L}{\partial \theta}} \end{bmatrix} = \text{ODESolve}\left(\begin{bmatrix} f_\theta(\mathbf{x}(t), t) \\ -\mathbf{a}(t)^\top \frac{\partial f_\theta(\mathbf{x}(t), t)}{\partial \mathbf{x}} \\ -\mathbf{a}(t)^\top \frac{\partial f_\theta(\mathbf{x}(t), t)}{\partial \theta} \end{bmatrix}, \begin{bmatrix} \mathbf{x}(t_1) \\ \frac{\partial L}{\partial \mathbf{x}(t_1)} \\ \mathbf{0}_{|\theta|} \end{bmatrix}, t_1, t_0 \right)$$

Update θ to reduce L

1. Neural ODEs

2. Video Prediction

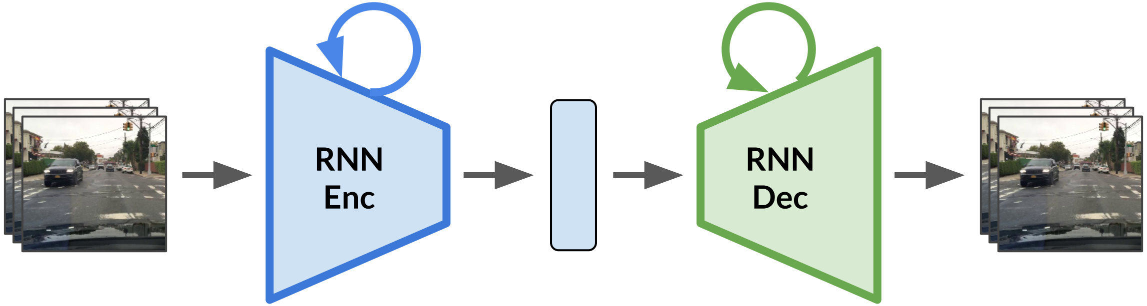
3. Image Generation

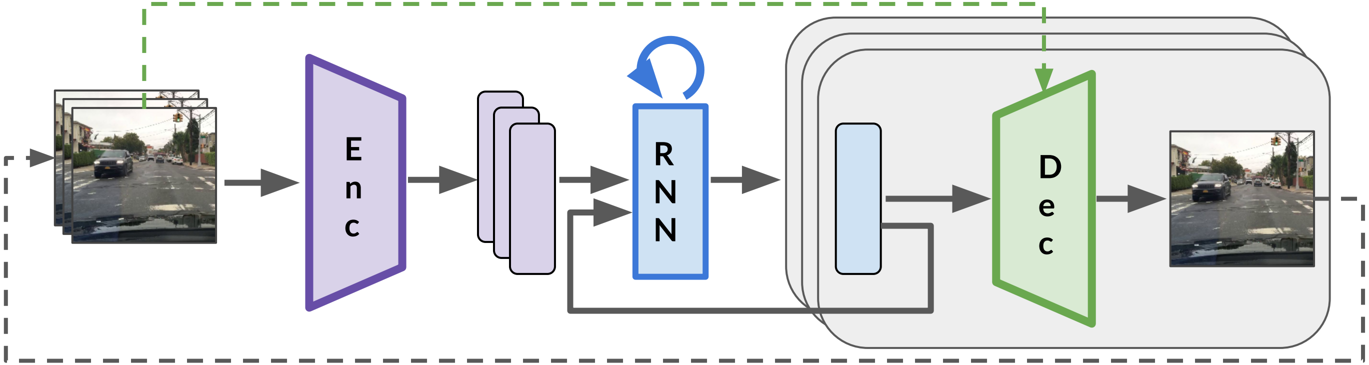
4. Mutual Interaction Minimization

2. Video Prediction

[Srivastava et al., 2015 (Moving MNIST)](#)

[Shi et al., 2016 (ConvLSTM)](#)



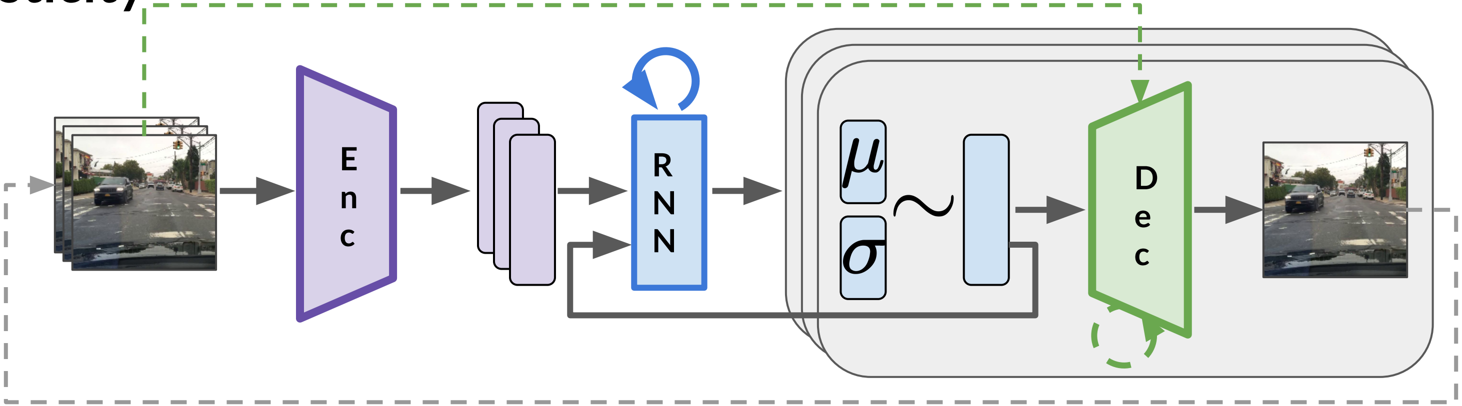


[Finn et al., 2017](#) (predicts “motion”, warps previous frame)

[Villegas et al., 2017](#) (MCnet : “content”, “motion”)

[Denton et al., 2017](#) (DrNet : “content”, “pose”)

Stochasticity



[Xue et al., 2016](#) (static "content", probabilistic "motion")

[Babaeizadeh et al., 2017](#) (SV2P : Basically [Finn et al., 2017](#) with stochasticity)

[Denton et al., 2018](#) (SVG-LP : learns a new prior per time step; has a recurrent encoder and a different recurrent decoder)

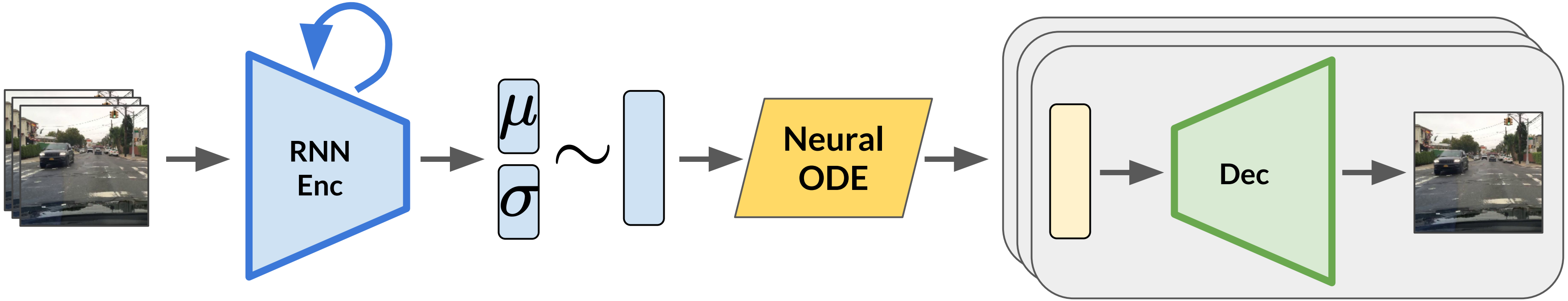
[Hsieh et al., 2018](#) (DDPAE : static "content", probabilistic "pose")

[Castrejon et al., 2019](#) (Improved VRNN : hierarchy of latent variables per time step)

[Villegas et al., 2019](#) (scaled up version of [Denton et al., 2018](#))

Mila

Can Neural ODEs model latent dynamics in video?

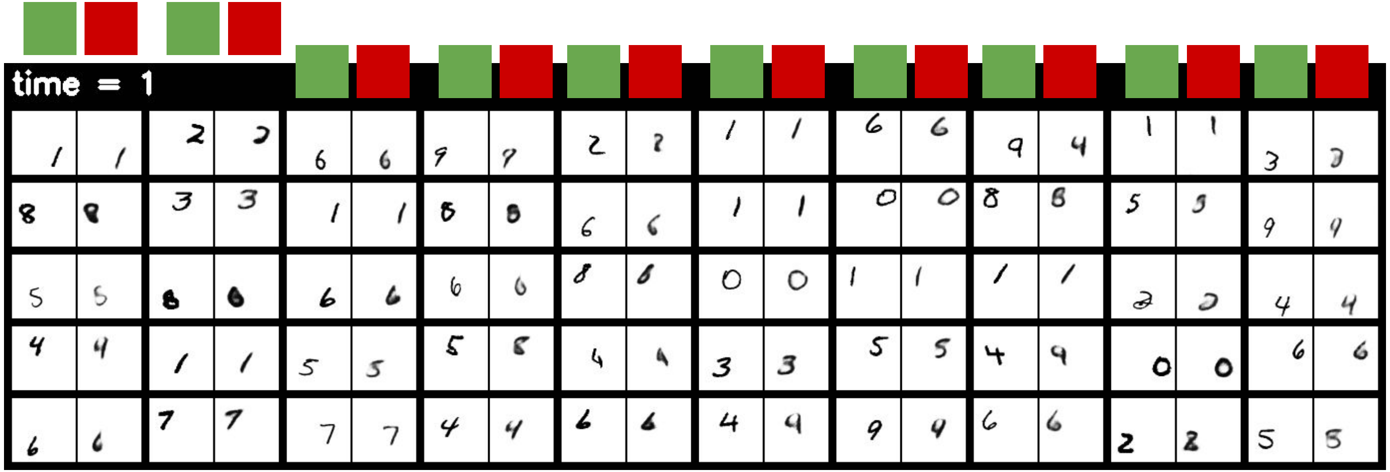


Simple Video Generation using Neural ODEs

David Kanaa*, Vikram Voleti*, Samira Kahou, Christopher Pal (NeurIPS 2019 Workshop)

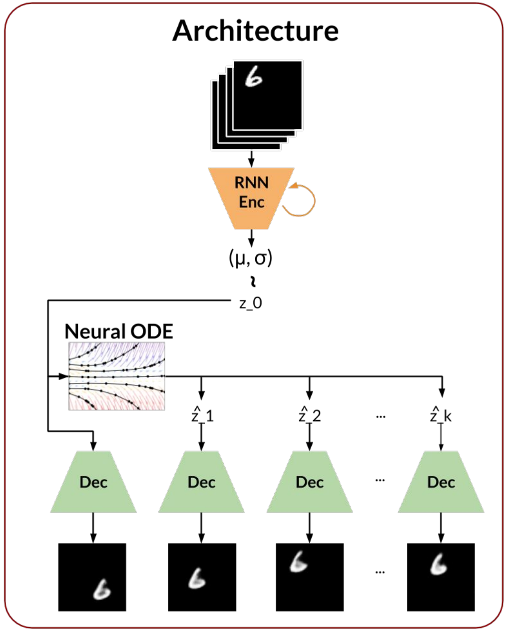
Ground truth

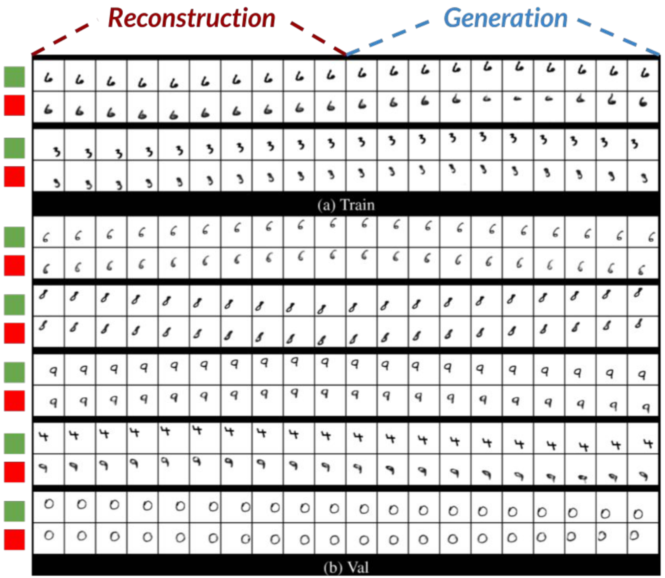
Prediction / Generation



Simple Video Generation using Neural ODEs

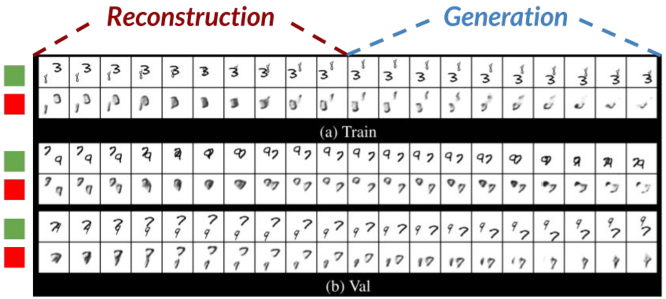
David Kanaa*, Vikram Voleti*, Samira Kahou, Christopher Pal (NeurIPS 2019 Workshop)





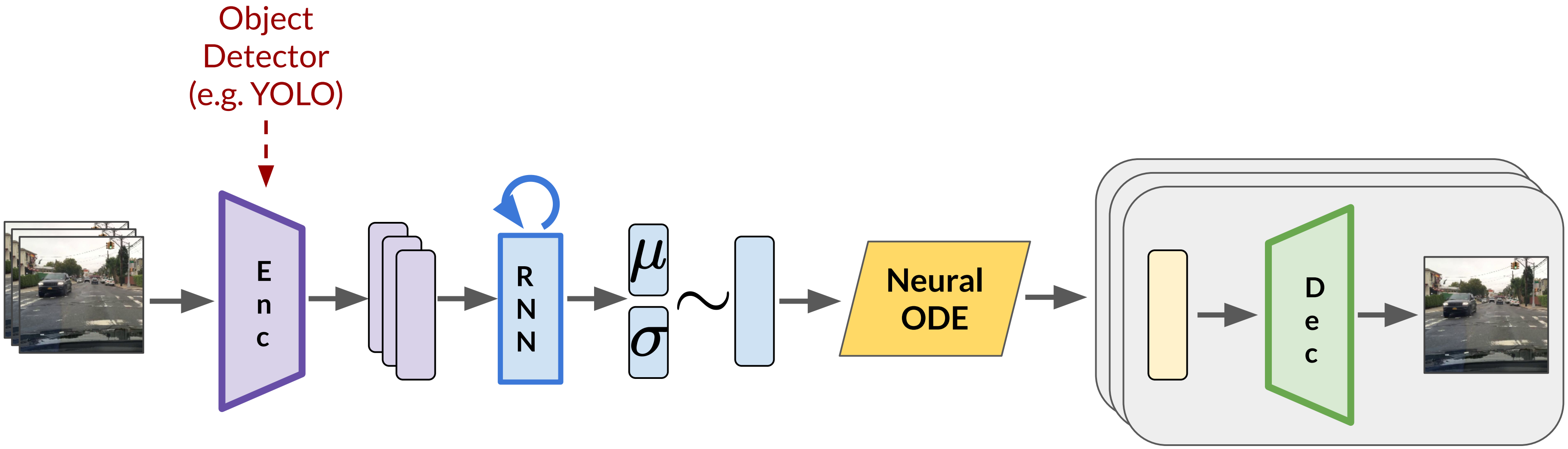
Moving MNIST 1-digit

Training on ONLY reconstruction!



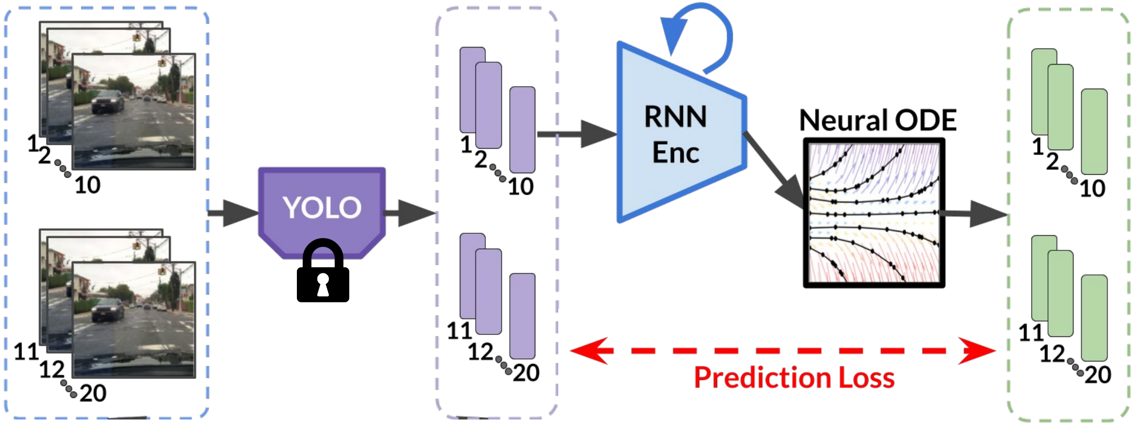
Moving MNIST 2-digit

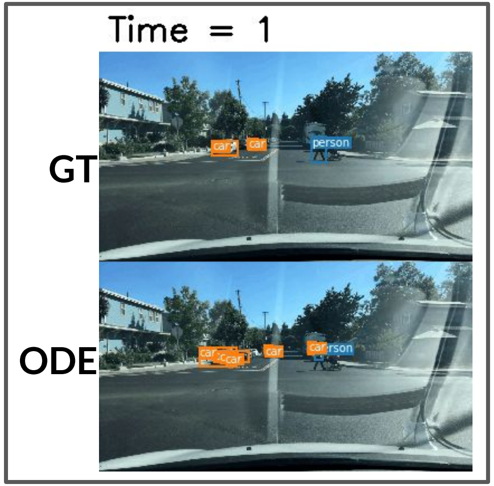
Can Neural ODEs model object dynamics in video?

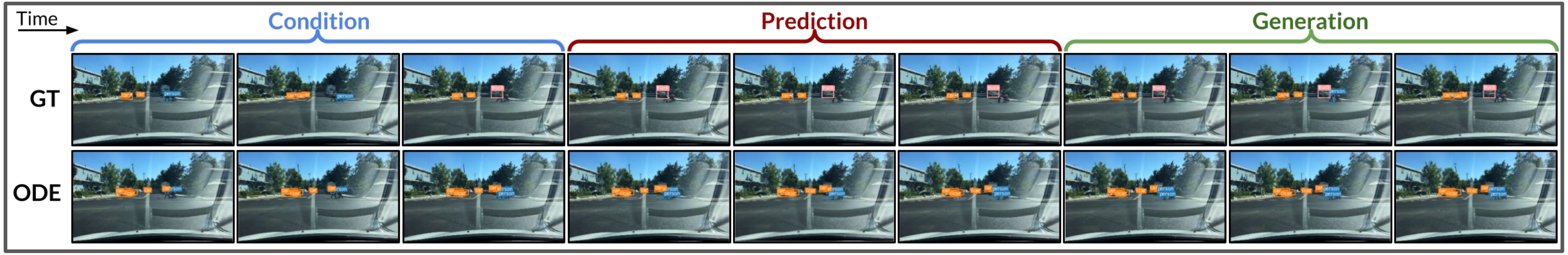


Mila

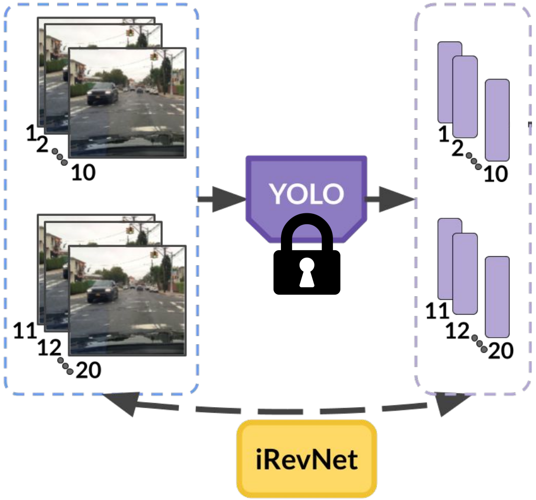
1) Object Dynamics

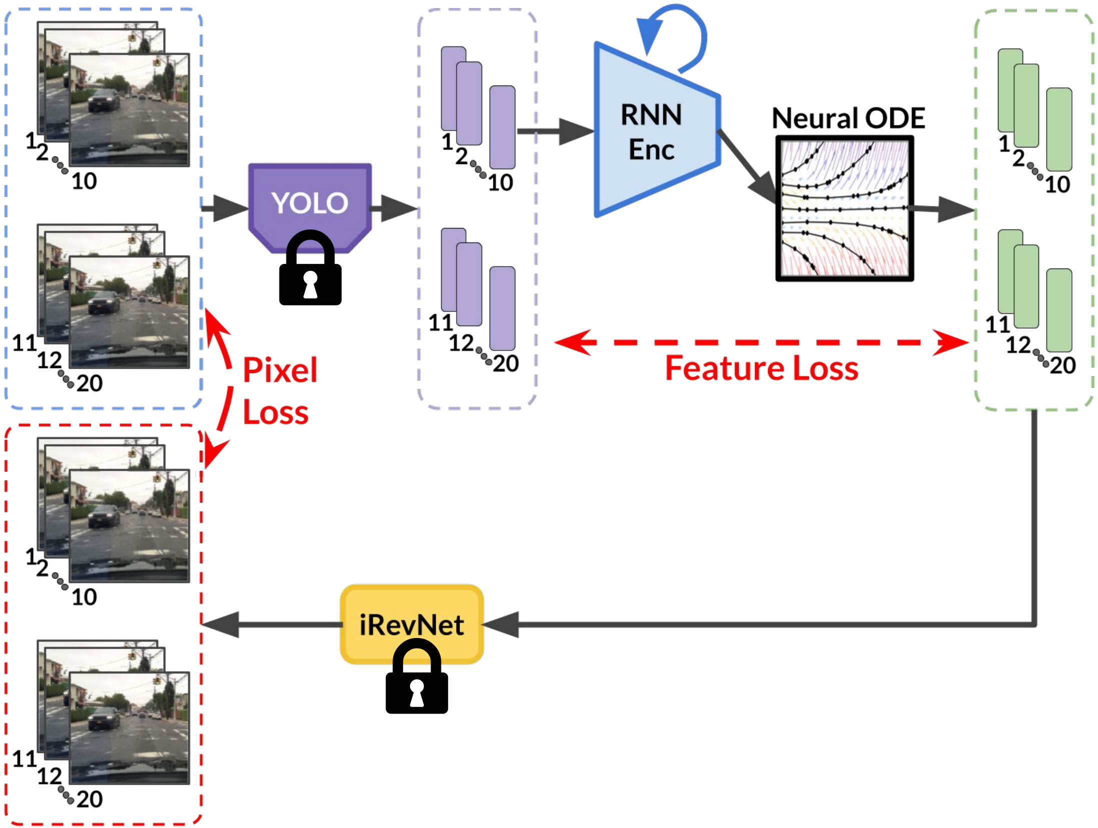




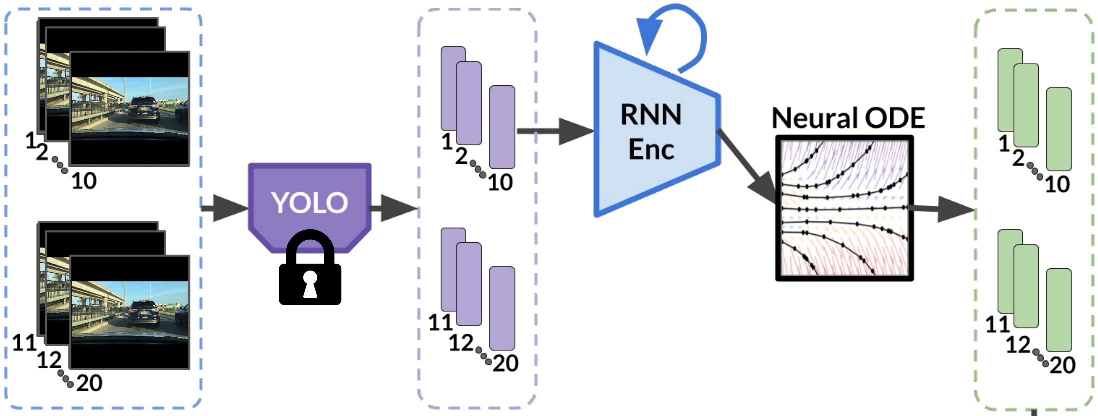


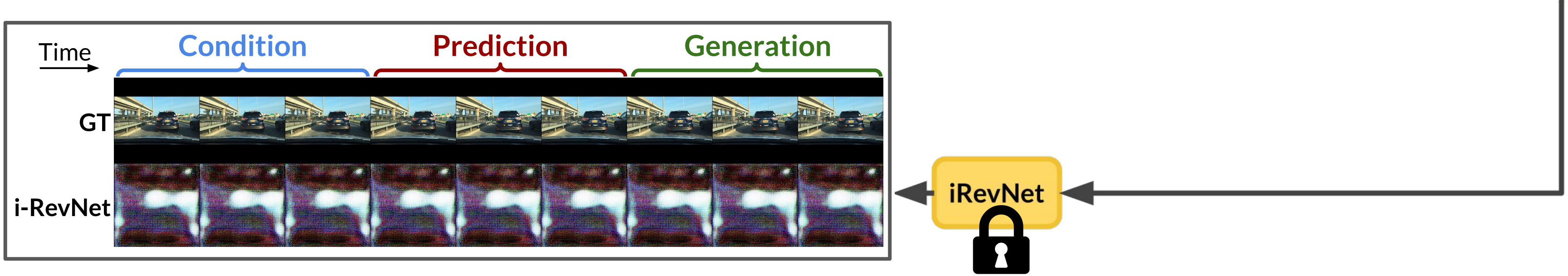
2) Video Prediction from Object Dynamics





2) Video prediction from Object Dynamics





Future work

Improve feature predictions

- Alternate loss formulation
- Add adversarial loss
- Increase time steps
- Add stochasticity

Improve video predictions

- Semantic segmentation
- Image optimization
- Alternate decoders
- End-to-end training

Optical flow estimation using Neural ODEs

$$\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} = f_\theta(\mathbf{x}(t), t)$$

Euler discretization

$$\mathbf{x}_{n+1} = \mathbf{x}_n + h\ \underbrace{f_\theta(\mathbf{x}_n, t_n)}_{\text{flow}}$$

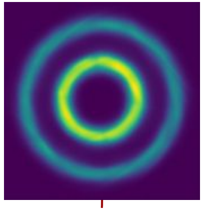
$$\mathbf{x}_{n+1} = \mathrm{warp}(\mathbf{x}_n,\ \underbrace{f_\theta(\mathbf{x}_n, t_n)}_{\text{Optical flow}})$$

1. Neural ODEs
2. Video Prediction
3. **Image Generation**
4. Mutual Interaction Minimization

Can Neural ODEs model continuous flows?

Continuous Normalizing Flows

Target distribution

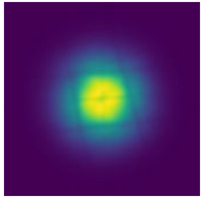


(such as real image manifold)

$\mathbf{x}_{im}$ ⇄ $f_\theta(\mathbf{x}(t), t)$ Neural ODE ⇄ $\mathbf{z}$

Sample from target distribution (such as an image)

Noise distribution



Sample from noise distribution (such as Gaussian)

"FFJORD"

(Free-Form Jacobian Of Reversible Dynamics)

https://arxiv.org/abs/1810.01367

3. Image Generation

FFJORD (ICLR 2019)

Change of variables:

$$\log p(\mathbf{x}_{im}) - \log p(\mathbf{z}) = \log\det\left|\frac{\mathrm{d}f_\theta}{\mathrm{d}\mathbf{x}(t)}\right|$$

Instantaneous change of variables:

$$\frac{\partial \log p(\mathbf{x}(t))}{\partial t} = -\mathrm{Tr}\left(\frac{\partial f_\theta}{\partial \mathbf{x}(t)}\right)$$

Initial value: $\begin{bmatrix} \mathbf{x}_{im} \\ 0 \end{bmatrix}$

$$\begin{bmatrix} \mathbf{z} \\ \log p(\mathbf{x}_{im}) - \log p(\mathbf{z}) \end{bmatrix} = \int_{t_0}^{t_1} \begin{bmatrix} f_\theta(\mathbf{x}(t), t) \\ -\mathrm{Tr}\left(\frac{\partial f_\theta}{\partial \mathbf{x}(t)}\right) \end{bmatrix} \mathrm{d}t$$

CIFAR10		ImageNet64	
BPD	**Time**	**BPD**	**Time**
3.40	≥5 days	-	-

https://arxiv.org/abs/1810.01367

How to Train your Neural ODE (ICML 2020)

Introduces 2 regularization terms:
1) Kinetic energy of flow
2) Jacobian norm of flow

$$\mathcal{K}(\theta) = \int_{t_0}^{t_1} \|f(\mathbf{x}(t), t, \theta)\|_2^2 \,\mathrm{d}t$$

$$\mathcal{B}(\theta) = \int_{t_0}^{t_1} \|\epsilon^\top \nabla_z f(\mathbf{x}(t), t, \theta)\|_2^2 \,\mathrm{d}t$$

CIFAR10		ImageNet64	
BPD	**Time**	**BPD**	**Time**
3.38	31.84	3.83	64.1

https://arxiv.org/abs/2002.02798

STEER

Introduces temporal regularization:

$$\mathbf{x}(t_1) = \mathbf{x}(t_0) + \int_{t_0}^{T} f_\theta(\mathbf{x}(t), t)\mathrm{d}t = \mathrm{ODESolve}(\mathbf{x}(t_0), f_\theta, t_0, T)$$

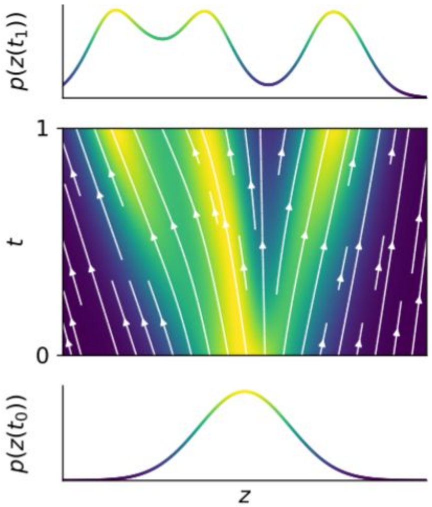
$$T \sim \mathrm{Uniform}(t_1 - b, t_1 + b)$$
$$b < t_1 - t_0$$

CIFAR10		ImageNet64	
BPD	**Time**	**BPD**	**Time**
3.39	22.24	-	-

https://arxiv.org/abs/2006.10711

3. Image Generation

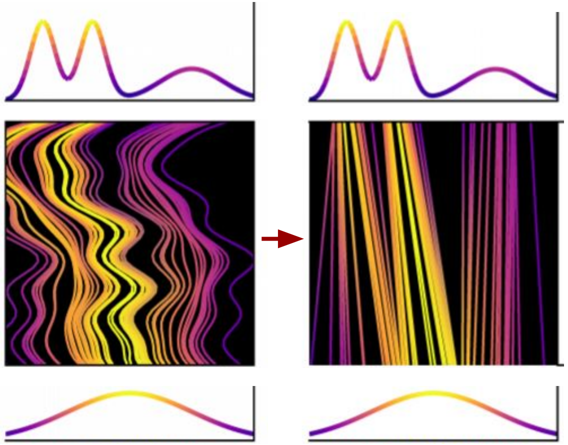
FFJORD (ICLR 2019)



CIFAR10		ImageNet64	
BPD	**Time**	**BPD**	**Time**
3.40	≥5 days	-	-

https://arxiv.org/abs/1810.01367

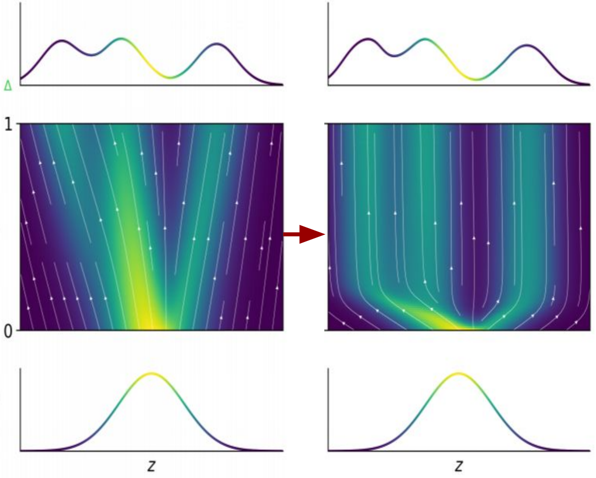
How to Train your Neural ODE (ICML 2020)



CIFAR10		ImageNet64	
BPD	**Time**	**BPD**	**Time**
3.38	31.84	3.83	64.1

https://arxiv.org/abs/2002.02798

STEER

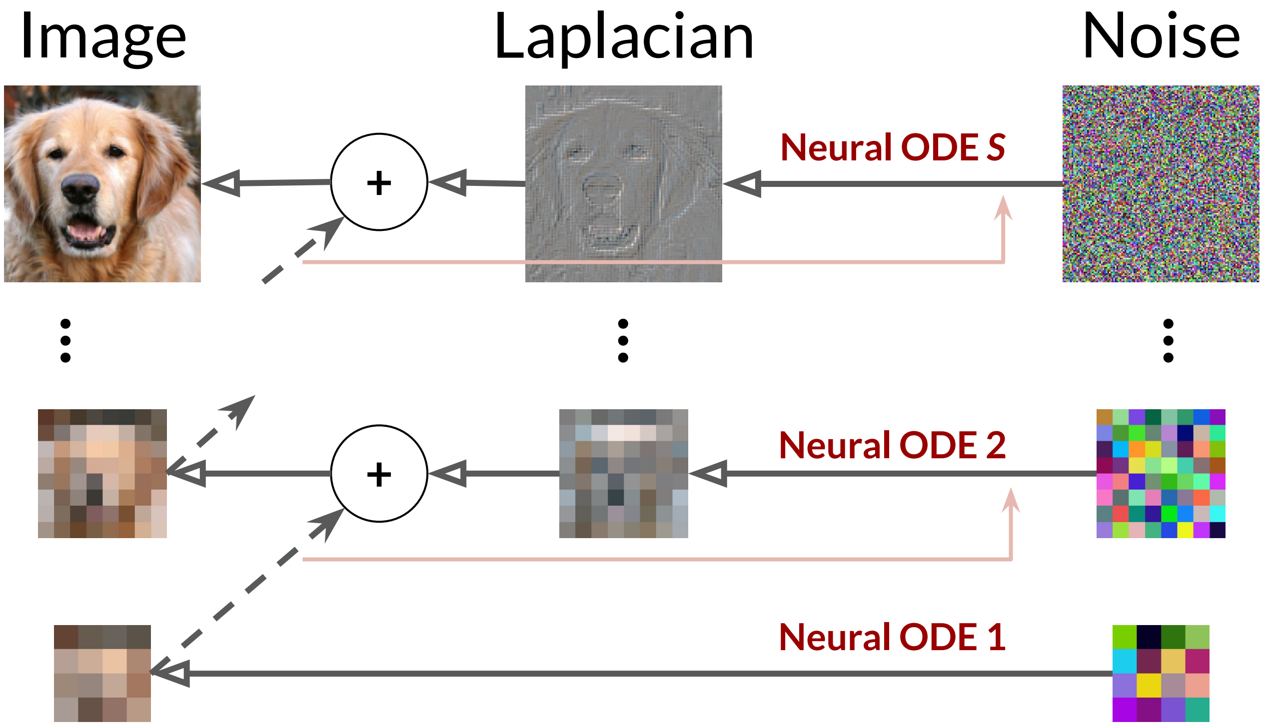


CIFAR10		ImageNet64	
BPD	**Time**	**BPD**	**Time**
3.39	22.24	-	-

https://arxiv.org/abs/2006.10711

Can Neural ODEs model multi-scale image flows?

Generation



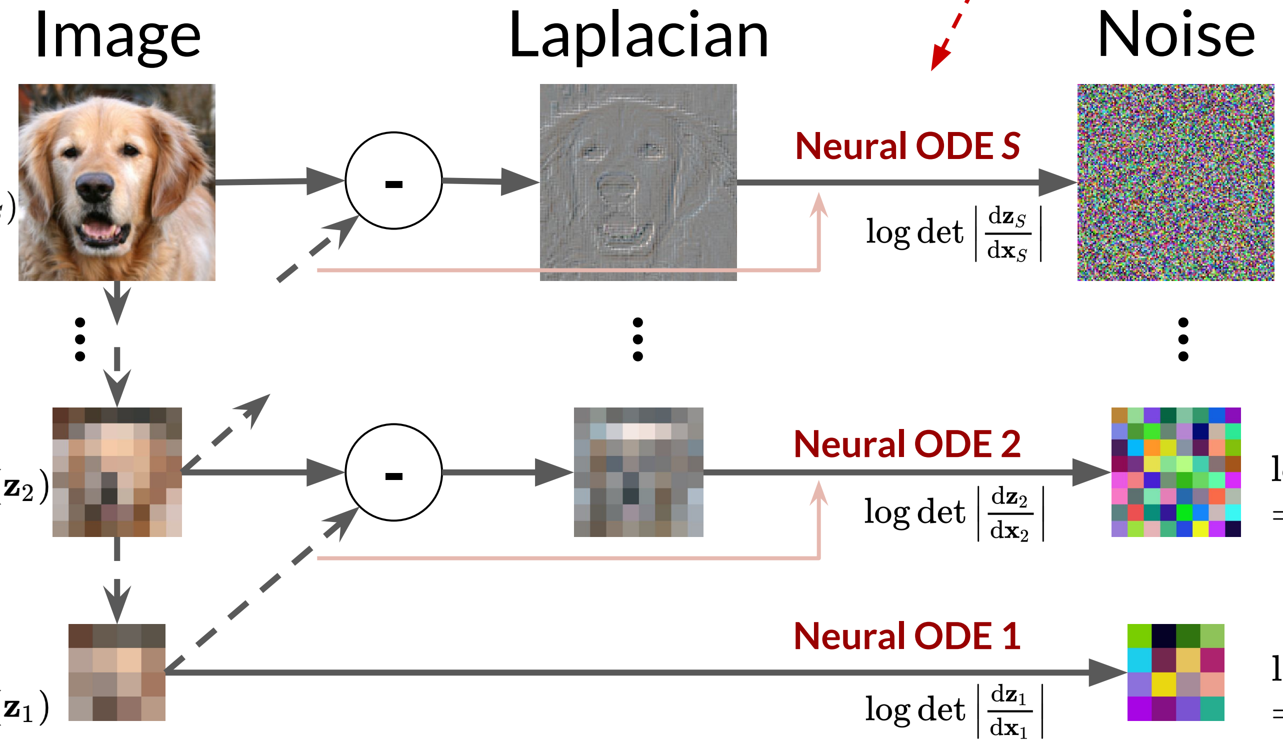
Training **jointly**

Memory issues!

$$\uparrow \log p(\mathbf{x}_S) = \log\det\left|\frac{\mathrm{d}\mathbf{z}_S}{\mathrm{d}\mathbf{x}_S}\right| + \log p(\mathbf{z}_S) + \log p(\mathbf{x}_{S-1})$$

$$\log p(\mathbf{x}_2) = \log\det\left|\frac{\mathrm{d}\mathbf{z}_2}{\mathrm{d}\mathbf{x}_2}\right| + \log p(\mathbf{z}_2) + \log p(\mathbf{x}_1)$$

$$\log p(\mathbf{x}_1) = \log\det\left|\frac{\mathrm{d}\mathbf{z}_1}{\mathrm{d}\mathbf{x}_1}\right| + \log p(\mathbf{z}_1)$$

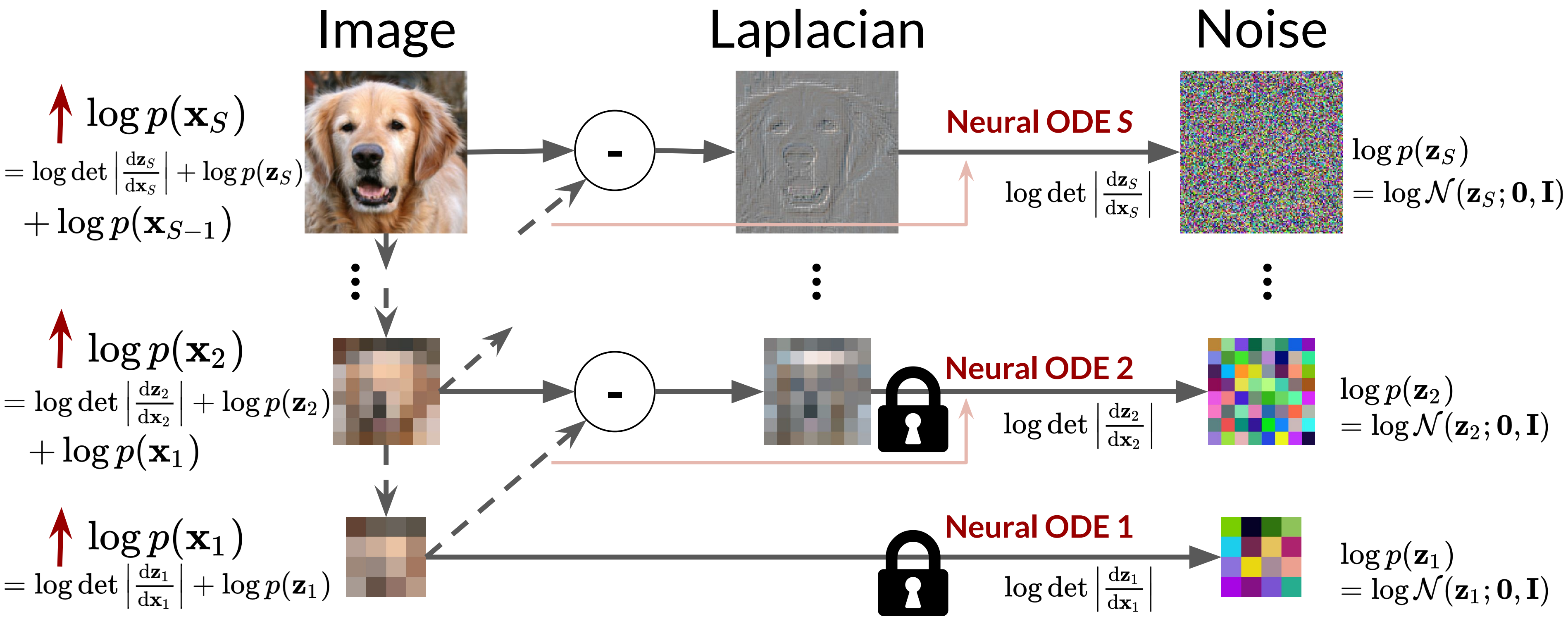


$$\log p(\mathbf{z}_S) = \log \mathcal{N}(\mathbf{z}_S; \mathbf{0}, \mathbf{I})$$

$$\log p(\mathbf{z}_2) = \log \mathcal{N}(\mathbf{z}_2; \mathbf{0}, \mathbf{I})$$

$$\log p(\mathbf{z}_1) = \log \mathcal{N}(\mathbf{z}_1; \mathbf{0}, \mathbf{I})$$

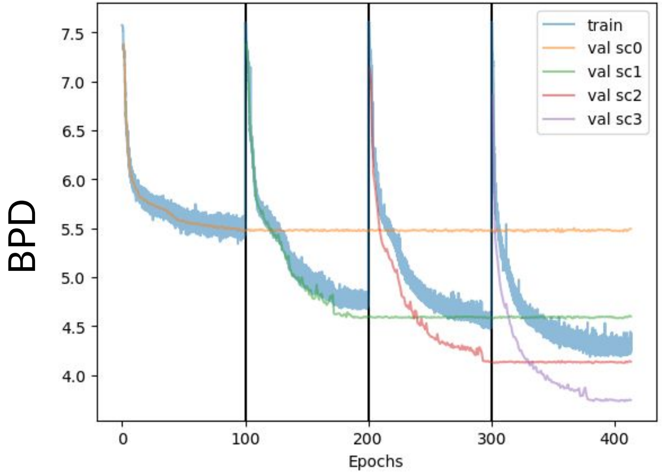
Training each scale consecutively

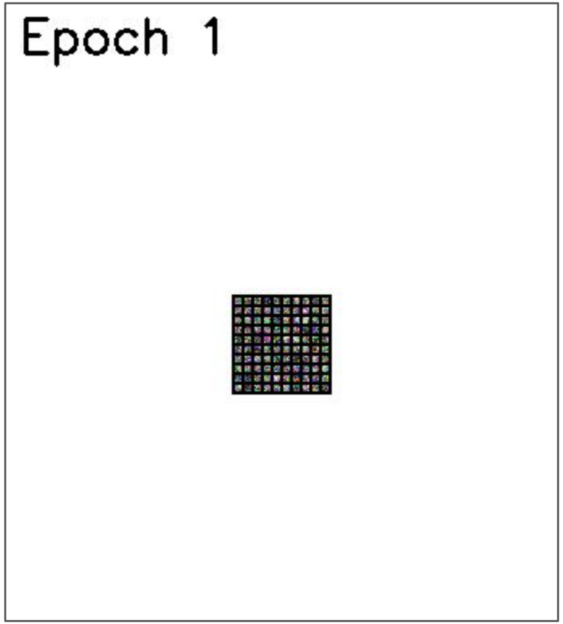


CIFAR10

4 scales: 4x4, 8x8, 16x16, 32x32

BPD	Time (hrs)
3.73	40

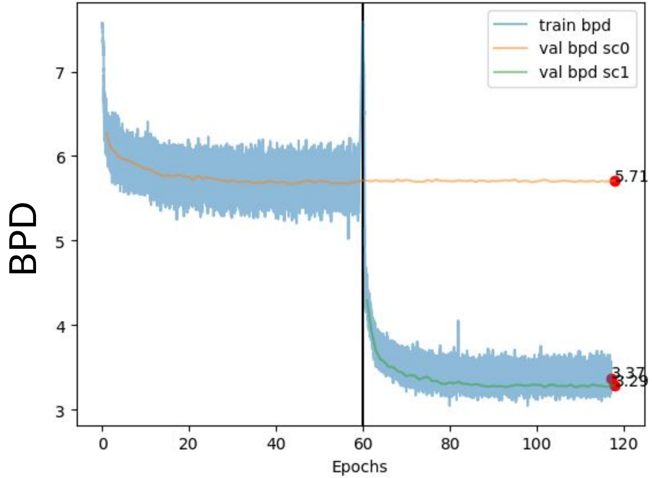


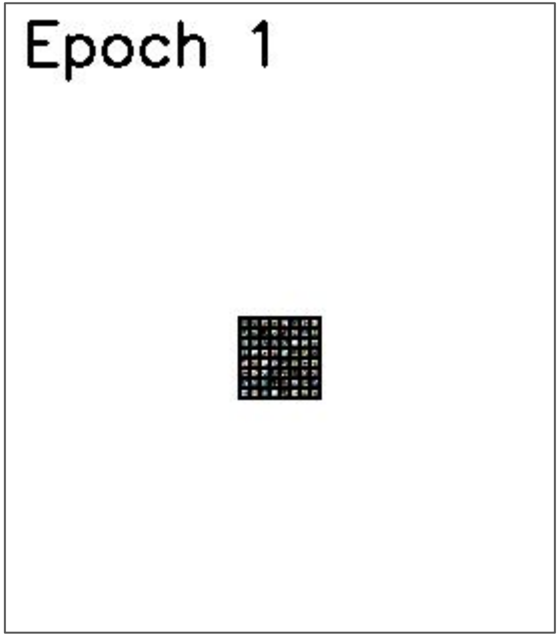


CIFAR10

2 scales: 3x3, 32x32

BPD	Time (hrs)
3.28	13.97

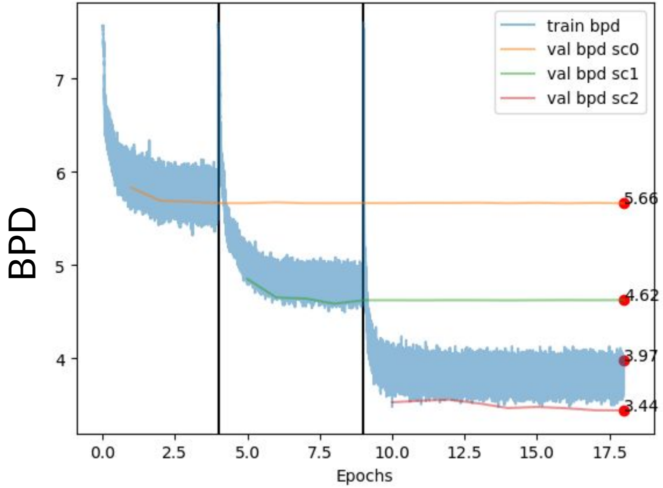


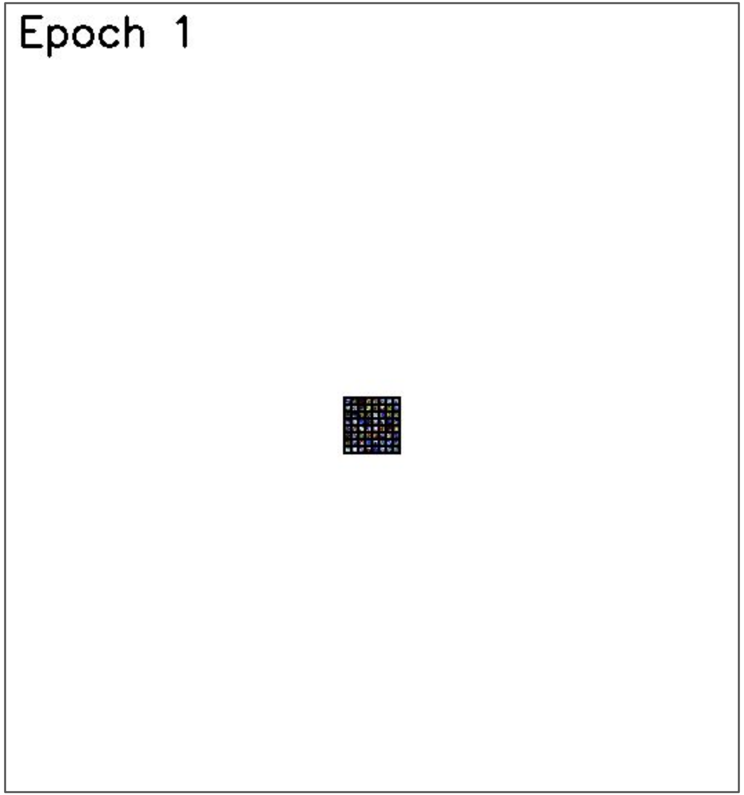


ImageNet64

3 scales: 3x3, 13x13, 64x64

BPD	Time (hrs)
3.48	44.8





3. Image Generation

Mila

	CIFAR10			ImageNet64		
	BPD	Time	FID	BPD	Time	FID
RealNVP	3.49	-	-	3.98	-	-
Glow	3.35	-	-	3.81	-	-
Flow++	3.09	-	-	3.69	-	-
ANF	3.05	-	-	3.66	-	-
FFJORD	3.40	≥5 days	-	-	-	-
FFJORD + STEER	3.40	86.34	-	-	-	-
FFJORD RNODE	3.38	31.84	-	3.83*	64.1*	-
FFJORD RNODE + STEER	3.397	22.24	-	-	-	-
Multiscale FFJORD RNODE (Ours) using only 1 GPU	**3.28**	**13.97**	143.14	**3.48**	**44.8**	249.4

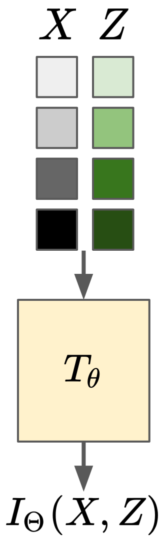
* used 4 GPUs

Future work

- Combine with other regularization methods (STEER, TayNODE)
- Investigate BPD vs FID
- Add adversarial loss
- Perform joint training
- **Multi-scale Video Generation**

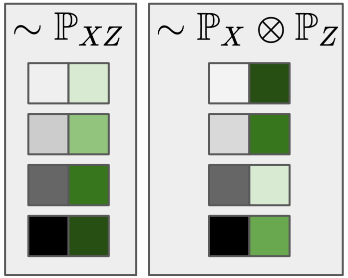
1. Neural ODEs
2. Video Prediction
3. Image Generation
4. **Mutual Interaction Minimization**

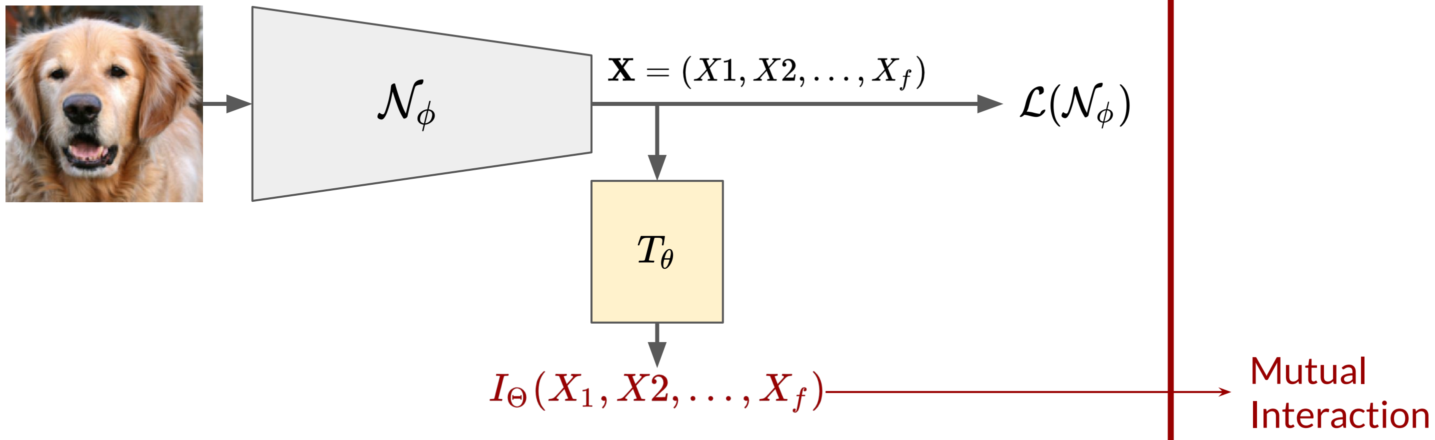
4. Mutual Interaction Minimization



MINE: (Mutual Information Neural Estimation)

$$I_\Theta(X, Z) = \sup_{\theta \in \Theta} \mathbb{E}_{\mathbb{P}_{XZ}}[T_\theta] - \log(\mathbb{E}_{\mathbb{P}_X \otimes \mathbb{P}_Z}[e^{T_\theta}])$$

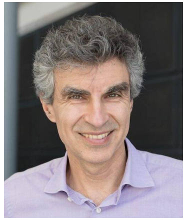




$$\min_{\mathcal{N}_\phi} \max_{T_\theta} \mathcal{L}(\mathcal{N}_\phi) + \mathbb{E}_{\mathbb{P}_{\mathbf{X}}}[T_\theta] - \log(\mathbb{E}_{\mathbb{P}_{X_1} \otimes \mathbb{P}_{X_2} \otimes \cdots \otimes \mathbb{P}_{X_f}}[e^{T_\theta}])$$

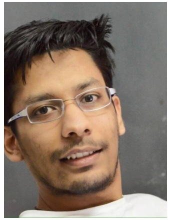
Acknowledgments

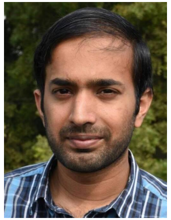

Christopher Pal


Yoshua Bengio


Xavier Bouthillier


Florian Golemo

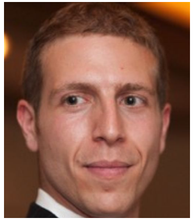

Anirudh Goyal


Krishna Murthy J


Samira Kahou


David Kanaa


Vincent Michalski


Derek Nowrouzezahrai


Adam Oberman


Doina Precup


Pascal Vincent


Sanja Fidler


Christopher Finlay


Florian Shkurti


Graham Taylor

Thank you!

voletiv.github.io