A Tale of Two Latent Flows: Learning Latent Space Normalizing Flow with Short-run Langevin Flow for Approximate Inference

Jianwen Xie

Cognitive Computing Lab

with Yaxuan Zhu, Yifei Xu, Dingcheng Li, and Ping Li

Baidu Research

aaai

Outline

1. Background, Motivation, and Contributions

2. The Proposed Framework

3. Experimental Results

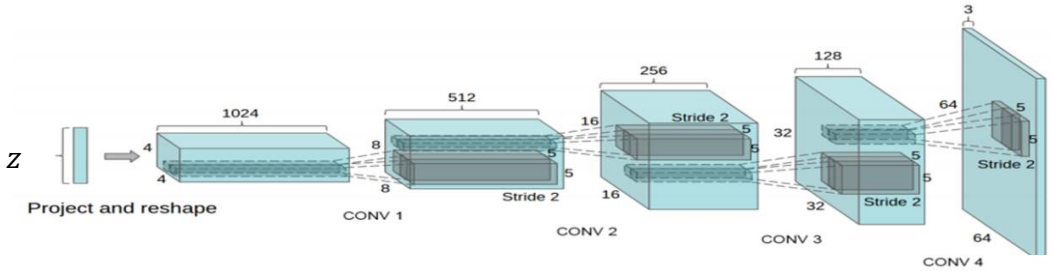
Outline

1. Background, Motivation, and Contributions

Deep Latent Variable Model

Nonlinear mapping by neural network (generator network)

$$x = g_\theta(z)$$



$g_\theta(z)$

Observed data (e.g.,images): $\{x_i, i = 1, \dots, n\}$

Corresponding latent vectors: $\{z_i, i = 1, \dots, n\}$

all x_i share the same ConvNet g_θ

We also assume z follows Gaussian distribution. (prior distribution)

Informative Prior

The Gaussian prior limits the expressivity of the latent space of data.

Can we learn a top-down generator with a non-Gaussian (informative) prior?

About the forms of the prior distributions

1) A known Gaussian prior: $z \sim N(z|0, \sigma^2 I)$, where σ is a hyperparameter.
2) A learnable Gaussian prior: $z \sim N\left(z\middle|\mu, \sigma^2 I\right)$, where μ and σ are learned from data
3) Other prior distribution: mixture Gaussian, ….
4) Energy-based prior: $z \sim p_\theta(z) = \frac{1}{Z(\theta)} \exp\left(f(z;\theta)\right)$ (Pang et al, NeurIPS, 2020)
5) Flow-based prior (our paper)

Flow-Based Model in Data Space

$$x = g_\alpha(z);\ z \sim q_0(z)$$

q_0 is a known Gaussian noise distribution. g_α is an invertible transformations where the log determinants of the Jacobians of the transformations can be explicitly obtained.

Under the change of variables, distribution of x can be expressed as

$$q_\alpha(x) = q_0(z)\left|\frac{1}{\det(Jac(g))}\right|$$

$$q_\alpha(x) = q_0(g_\alpha^{-1}(x))\left|\det(\partial g_\alpha^{-1}(x)/\partial x)\right|$$

In the flow-based model, g_α is composed of a sequence of transformations $g_\alpha = g_{\alpha 1} \cdot g_{\alpha 2} \cdots g_{\alpha m}$, therefore, we have

$$q_\alpha(x) = q_0(g_\alpha^{-1}(x))\Pi_{i=1}^{m}\left|\det(\partial h_{i-1}/\partial h_i)\right|$$

Glow: Generative flow with invertible 1x1 convolutions. *Diederik P Kingma and Prafulla Dhariwal*. NIPS 2018

Flow-Based Model in Data Space

$$x = g_\alpha(z);\ z \sim q_0(z)$$

$$q_\alpha(x) = q_0(g_\alpha^{-1}(x))|\det(\partial g_\alpha^{-1}(x)/\partial x)|$$

In general, it is intractable !!

The key idea of the flow-based model is to choose transformations g whose Jacobian is a triangle matrix, so that the computation of determinant becomes

$$|\det(\partial h_{i-1}/\partial h_i)| = \Pi|\mathrm{diag}(\partial h_{i-1}/\partial h_i)|$$

diag() takes the diagonal of the Jacobian matrix

Maximum likelihood estimation of q $\min_\alpha \mathrm{KL}(p_{\mathrm{data}}||q_\alpha)$

Contributions

(1) We propose a simple but novel deep generative model where a latent space normalizing flow, which serves as a prior, stands on a single top-down generator network.

(2) We propose a natural and principled maximum likelihood learning algorithm to jointly train the latent space normalizing flow and the top-down network with MCMC-based inference over latent variables.

(3) We also propose to use short-run Langevin flow as an approximate inference for efficient training.

(4) We provide theoretical understanding of the proposed learning framework.

(5) We provide extensive and strong experimental results on different aspects, including image synthesis, inference, reconstruction, inpainting and recovery, to validate the effectiveness of the proposed models and learning algorithms.

Outline

2. The Proposed Framework

Latent Space Flow-Based Prior Model

$x \in R_D$: observed signal (such as an image). $z \in R_d$: latent vector.

The joint distribution of (x, z): $p_\theta(x, z) = p_\alpha(z)p_\beta(x|z)$

(i) Flow-based prior model: $z_0 \sim q_0(z_0),\ z = f_\alpha(z_0)$

$q_0(z_0) = \mathcal{N}(0, I_d).\ f_\alpha : \mathbb{R}^d \to \mathbb{R}^d$ is an invertible or bijective function

(ii) Top-down generation model: $x = g_\beta(z) + \epsilon$

$\theta = (\alpha, \beta)$

Latent Space Flow-Based Prior Model

$x \in R_D$: observed signal (such as an image). $z \in R_d$: latent vector.

The joint distribution of (x, z): $p_\theta(x, z) = p_\alpha(z)p_\beta(x|z)$

(i) Flow-based prior model: $z_0 \sim q_0(z_0),\ z = f_\alpha(z_0)$

$$p_\alpha(z) = q_0(z_0)\frac{dz_0}{dz} = q_0(f_\alpha^{-1}(z))\left|\det\left(\frac{\partial f_\alpha^{-1}(z)}{\partial z}\right)\right| = q_0(f_\alpha^{-1}(z))\prod_{l=1}^{L}\left|\det\left(\frac{\partial z_{i-l}}{\partial z_l}\right)\right|$$

(ii) Top-down generation model: $x = g_\beta(z) + \epsilon$

The flow-based prior model has two nice properties that are suitable for a prior:
(1) analytically tractable normalized density (good for training)
(2) easy to draw samples from by using ancestral sampling (good for testing)

Generative Learning of Flow-Based Prior Model

$$\theta = (\alpha, \beta)$$

$$x = g_\beta(z) + \epsilon \qquad z \sim p_\alpha(z) \qquad \epsilon \sim \mathcal{N}(0, \sigma^2 I_D)$$

Log-likelihood

$$\mathcal{L}(\theta) = \sum_{i=1}^{N} \log p_\theta(x_i)$$

Gradient for a training example

$$\begin{aligned} \nabla_\theta \log p_\theta(x) &= \mathbb{E}_{p_\theta(z|x)} \left[\nabla_\theta \log p_\theta(x, z) \right] \\ &= \mathbb{E}_{p_\theta(z|x)} [\nabla_\theta (\log p_\alpha(z) + \log p_\beta(x|z))] \end{aligned}$$

(1) (2)

Generative Learning of Flow-Based Prior Model

$$\theta = (\alpha, \beta) \qquad \boxed{x = g_\beta(z) + \epsilon \qquad z \sim p_\alpha(z) \qquad \epsilon \sim \mathcal{N}(0, \sigma^2 I_D)}$$

(1) Learning prior model α

$$\nabla_\alpha \log p_\theta(x) = \mathbb{E}_{p_\theta(z|x)}[\nabla_\alpha \log p_\alpha(z)]$$

(2) Learning generation model β

$$\nabla_\beta \log p_\theta(x) = \mathbb{E}_{p_\theta(z|x)}[\nabla_\beta \log p_\beta(x|z)]$$

Generative Learning of Flow-Based Prior Model

$$\theta = (\alpha, \beta)$$

$$x = g_\beta(z) + \epsilon \qquad z \sim p_\alpha(z) \qquad \epsilon \sim \mathcal{N}(0, \sigma^2 I_D)$$

(1) Learning prior model α

$$\nabla_\alpha \log p_\theta(x) = \mathbb{E}_{p_\theta(z|x)}[\nabla_\alpha \log p_\alpha(z)] = \mathbb{E}_{p_\theta(z|x)}[\nabla_\alpha l_\alpha(z)]$$

$$\begin{aligned} \log p_\alpha(z) &= \log q_0(f_\alpha^{-1}(z)) + \textstyle\sum_{l=1}^{L} \log |\det(\partial z_{l-1}/\partial z_l)| \\ &= \log q_0(z_0) + \textstyle\sum_{l=1}^{L} \mathrm{sum}(\log |\mathrm{diag}(\partial z_{l-1}/\partial z_l)|) \end{aligned}$$

Given a datapoint z, computing its log-likelihood only need one pass of the inverse function f_α^{-1}

Generative Learning of Flow-Based Prior Model

$$\theta = (\alpha, \beta) \qquad x = g_\beta(z) + \epsilon \qquad z \sim p_\alpha(z) \qquad \epsilon \sim \mathcal{N}(0, \sigma^2 I_D)$$

(2) Learning generation model β

$$\nabla_\beta \log p_\theta(x) = \mathbb{E}_{p_\theta(z|x)}[\nabla_\beta \log p_\beta(x|z)]$$

$$\nabla_\beta \log p_\beta(x|z) = \nabla_\beta(-\tfrac{1}{2\sigma^2}||x - g_\beta(z)||^2 + \text{const}) = \tfrac{1}{\sigma^2}(x - g_\beta(z))\nabla_\beta g_\beta(z)$$

Because $p_\beta(x|z)$ is in the form of $\mathcal{N}(g_\beta(z), \sigma^2 I_D)$

Generative Learning of Flow-Based Prior Model

(1) Learning prior model α $\quad \nabla_\alpha \log p_\theta(x) = \mathbb{E}_{p_\theta(z|x)}[\nabla_\alpha \log p_\alpha(z)]$

(2) Learning generation model β $\quad \nabla_\beta \log p_\theta(x) = \mathbb{E}_{p_\theta(z|x)}[\nabla_\beta \log p_\beta(x|z)]$

Sampling from $p_\theta(z|x)$ can be achieved by *Langevin dynamics* that iterates

$$z_{(k+1)} = z_{(k)} + \xi \nabla_z \log p_\theta(z_{(k)}|x) + \sqrt{2\xi}\epsilon_{(k)}, \quad z_{(0)} \sim q_0(z), \epsilon_{(k)} \sim \mathcal{N}(0, I_d),$$

$$\nabla_z \log p_\theta(z|x) = \nabla_z \log p_\alpha(z) + \nabla_z \log p_\beta(x|z) = \nabla_z l_\alpha(z) + \frac{1}{\sigma^2}(x - g_\beta(z))\nabla_z g_\beta(z)$$

prior model generation model

Generative Learning of Flow-Based Prior Model

Algorithm 1 Maximum likelihood learning of latent space flow-based prior model

Input: (1) Observed signals for training $\{x_i\}_i^N$; (2) Maximal number of learning iterations T; (3) Numbers of Langevin steps for posterior K; (4) Langevin step size ξ for the posterior ; (5) Learning rates for flow-based prior model and the generation model $\{\eta_\alpha, \eta_\beta\}$.

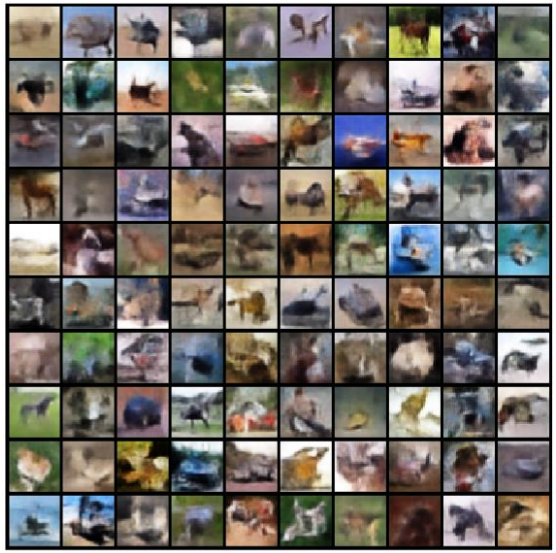
Output: Parameters α for the generation model and β for the flow-based prior model

1: Randomly initialize α and β
2: **for** $t \leftarrow 1$ to T **do**
3: Sample a batch of observed examples $\{x_i\}_i^n$
4: For each x_i, sample the posterior $z_i \sim p_\theta(z|x_i)$ using K Langevin steps in Eq.(7) with a step size ξ.
5: Update flow-based prior by Adam with the gradient $\nabla\alpha$ in Eq.(5) and a learning rate γ_α.
6: Update generation model by Adam with the gradient $\nabla\beta$ in Eq.(6) and a learning rate γ_β.
7: **end for**

Outline

3. Experimental Results

Task 1: Image Synthesis and Reconstruction



(a) CIFAR10

(b) SVHN

(c) Celeba

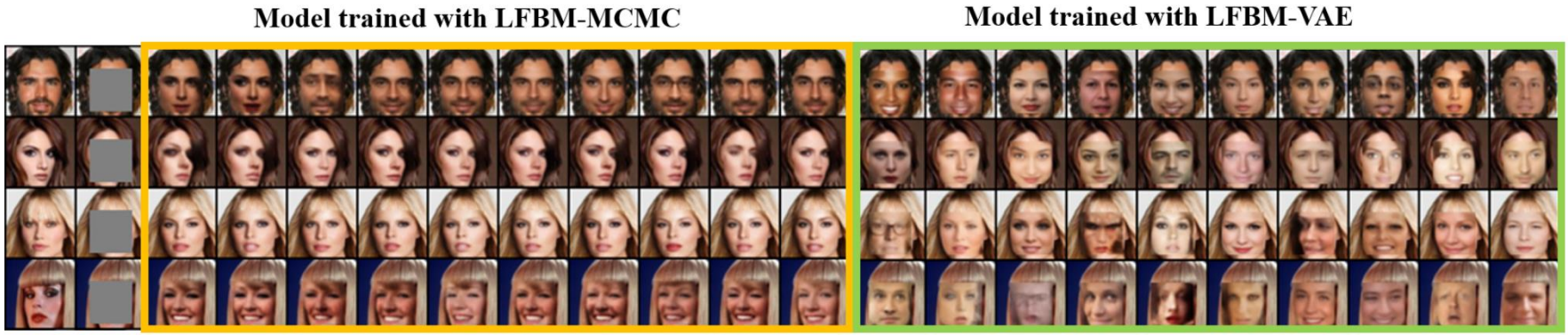
Generated samples from CIFAR10 (32×32), SVHN, (32×32) and Celeba (64× 64) datasets. Samples are obtained from the model trained with Langevin flow as approximate inference (LFBM)

Task 1: Image Synthesis and Reconstruction

The comparison results on different datasets. The MSE and FID (smaller is better) are used to test the quality of the reconstructed images and the synthesized images, respectively.

Models		VAE	2sVAE	RAE	SRI	SRI (L=5)	ABP	LEBM	LFBM	
									VAE	MCMC
SVHN	MSE	0.019	0.019	0.014	0.018	0.011	-	0.008	0.005	**0.005**
	FID	46.78	42.81	40.02	44.86	35.23	49.71	29.44	24.96	**23.64**
Cifar10	MSE	0.057	0.056	0.027	-	-	0.018	0.020	0.020	**0.016**
	FID	106.37	72.90	74.16	-	-	90.30	70.15	69.70	**66.41**
CelebA	MSE	0.021	0.021	0.018	0.020	0.015	-	0.013	0.014	**0.011**
	FID	65.75	44.40	40.95	61.03	47.95	51.50	37.87	33.64	**33.64**

Task 2: Supervised Image Inpainting



Supervised image inpainting results on the CelebA dataset. Images in the first column are the original images. Images in the second column are the masked images to be inpainted. Images in column 3 to column 12 (yellow panel) are inpainting results using the learned LFBM-MCMC model. Images in column 13 to column 22 (green panel) are inpainting results using the trained LFBM-VAE. For each panel, different columns correspond to different initializations of the inference process.

Task 3: Anomaly Detection

- Likelihood-based anomaly detection is another task that can help evaluate the proposed model.
- With a well-learned model from the normal data, we can detect the anomalous data by firstly sampling the latent code of the given testing image from the posterior distribution by the Langevin dynamics, and then computing the logarithm of the joint probability.
- The joint probability $p(z, x)$ should be high for the normal images and low for the anomalous ones.

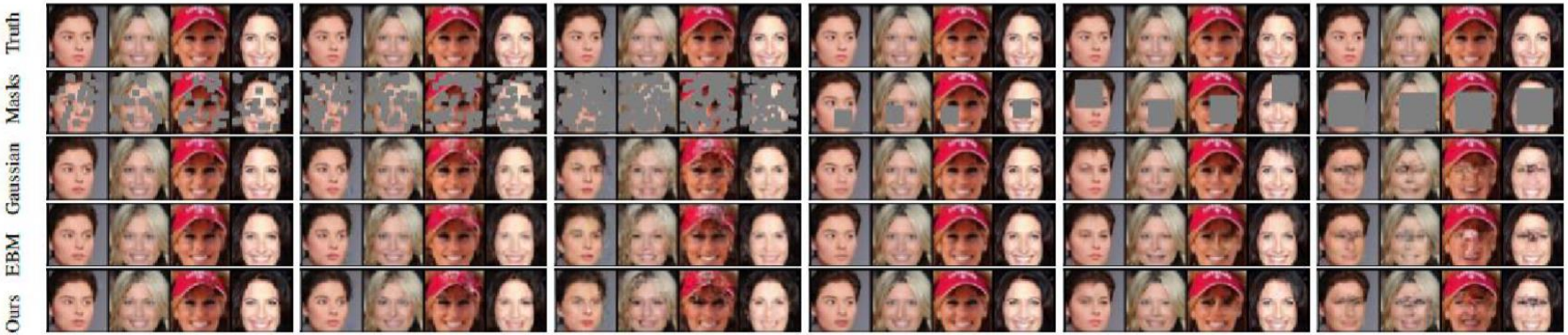
Task 3: Anomaly Detection

We treat each class in the MNIST dataset as an anomalous class and leave the others as normal. We train the model only with the normal data.

Then the model is tested with both the normal and anomalous data. To evaluate the performance, we use $\log p_\theta(x, z)$ as our decision function to compute the area under the precision-recall curve (AUPRC).

Heldout Digit	1	4	5	7	9
VAE (Kingma and Welling 2014)	0.063	0.337	0.325	0.148	0.104
MEG (Kumar et al. 2019)	0.281 ± 0.035	0.401 ± 0.061	0.402 ± 0.062	0.290 ± 0.040	0.342 ± 0.034
BiGAN-σ (Zenati et al. 2018b)	0.287 ± 0.023	0.443 ± 0.029	0.514 ± 0.029	0.347 ± 0.017	0.307 ± 0.028
EBM-VAE (Han et al. 2020)	0.297 ± 0.033	0.723 ± 0.042	0.676 ± 0.041	0.490 ± 0.041	0.383 ± 0.025
LEBM (Pang et al. 2020)	0.336 ± 0.008	0.630 ± 0.017	0.619 ± 0.013	0.463 ± 0.009	0.413 ± 0.010
ABP (Han et al. 2017)	0.095 ± 0.028	0.138 ± 0.037	0.147 ± 0.026	0.138 ± 0.021	0.102 ± 0.033
LFBM (ours)	$\mathbf{0.349 \pm 0.002}$	$\mathbf{0.812 \pm 0.007}$	$\mathbf{0.823 \pm 0.009}$	$\mathbf{0.682 \pm 0.004}$	$\mathbf{0.514 \pm 0.008}$

Task 4: Unsupervised Image Recovery



(a) salt-pepper 30% (b) salt-pepper 50% (c) salt-pepper 70% (d) mask size 20×20 (e) mask size 30×30 (f) mask size 40×40

A comparison of unsupervised image recovery results by different methods on training images with different levels of occlusions. In each panel, the first row shows some original images that are used in the training process, the second row shows the corresponding occluded images with a certain occlusion level, and the third, fourth and fifth rows show the recovered images by models using Gaussian prior, EBM prior and normalizing flow prior, respectively.

Task 4: Unsupervised Image Recovery

Image synthesis by models learned from incomplete images. Each panel represents a different level of occlusion.

Task 4: Unsupervised Image Recovery

Salt and pepper mask			
Occ %	30%	50%	70%
flow (ours)	**0.0244**	**0.0317**	**0.0464**
EBM	0.0256	0.0319	0.0465
Gaussian	0.0259	0.0326	0.0472
Single region mask			
mask size	20×20	30×30	40×40
flow (ours)	0.0420	0.0587	**0.0864**
EBM	0.0429	0.0684	0.0957
Gaussian	**0.0404**	**0.0572**	0.0918

MSEs of methods with different priors in unsupervised image recovery

Salt and pepper mask			
Occ %	30%	50%	70%
flow (ours)	**46.2**	**59.14**	**86.77**
EBM	52.78	61.91	88.27
Gaussian	153.01	156.71	172.77
Single region mask			
mask size	20×20	30×30	40×40
flow (ours)	**42.39**	**47.52**	**72.47**
EBM	49.16	51.59	77.39
Gaussian	150.95	146.41	184.53

FIDs of methods with different priors in unsupervised image recover

Thank You