Cooperative Learning of Energy-Based Model and Latent Variable Model via MCMC Teaching

Jianwen Xie [1,2], Yang Lu [2,3], Ruiqi Gao [2], Ying Nian Wu [2]

[1] Hikvision Research Institute, USA
[2] University of California, Los Angeles, USA
[3] Amazon RSML (Retail System Machine Learning) Group, USA

Feb 7, 2018

Introduction

Introduction

This paper proposes a cooperative learning algorithm to train both energy-based model and latent variable model jointly.

Bottom-up ConvNet	Top-down ConvNet
energy	**latent variables**
$\Uparrow$	$\Downarrow$
signal	signal
(a) Descriptor Net	(b) Generator Net
(Energy-based Model)	(Latent Variable Model)

Introduction

It is very challenging to learn the energy-base model (descriptor net) and the latent variable model (generator net) separately by MLE.

1. The likelihood of both models involves intractable integrals
2. The gradients of both log-likelihoods involve intractable expectations that need to be approximated by Markov chain Monte Carlo (MCMC).

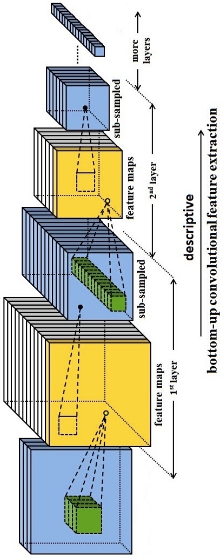
e.g.,
energy-based model $\Rightarrow$ contrastive divergence (CD)
latent variable model $\Rightarrow$ Variational inference or Adversarial learning

Introduction

- We find it much easier for the cooperative algorithm to learn both models.
- The cooperative learning algorithm interweaves the maximum likelihood algorithms for learning the two models, and each iteration consists of two steps:
  - (1 **modified contrastive divergence** for energy-based model (descriptor),
  - (2 **MCMC teaching** of the latent variable model (generator)
- Experiments show that the cooperative learning algorithm can learn realistic models of data.

Method

Energy-based model (Descriptor net)



Input RGB image

The **descriptor net** [Xie et al. 2016] is in the form of exponential tilting of a reference distribution :

$$p(X;\theta) = \frac{1}{Z(\theta)} \exp\left[f(X;\theta)\right] p_0(X),$$

where X is an image, $f(X;\theta)$ is a bottom-up ConvNet, $Z(\theta)$ is the normalizing constant, and $p_0(X)$ is the reference distribution such as Gaussian white noise,

$$p_0(X) = \frac{1}{(2\pi s^2)^{D/2}} \exp\left[-\frac{\|X\|^2}{2s^2}\right]$$

Energy-based model (Descriptor net)

The descriptor net can be written as the form of energy-based model:

$$p(X;\theta) = \frac{1}{Z(\theta)} \exp\left[-\mathcal{E}(X;\theta)\right],$$

where the energy function is:

$$\mathcal{E}(X;\theta) = \frac{\|X\|^2}{2s^2} - f(X;\theta).$$

Term $\frac{\|X\|^2}{2s^2}$ is from Gaussian distribution.

Maximum likelihood learning of the descriptor net

Suppose we observe training examples $\{X_i, i = 1, ..., n\}$.

The maximum likelihood estimation (MLE) seeks to maximize the log-likelihood function:

$$L_p(\theta) = \frac{1}{n} \sum_{i=1}^{n} \log p(X_i; \theta)$$

Maximum likelihood learning of the descriptor net

Suppose we observe training examples $\{X_i, i = 1, ..., n\}$.

The maximum likelihood estimation (MLE) seeks to maximize the log-likelihood function:

$$L_p(\theta) = \frac{1}{n}\sum_{i=1}^{n} \log p(X_i; \theta)$$

The gradient of the log-likelihood is:

$$L_p'(\theta) \quad = \quad \mathrm{E}_\theta\left[\frac{\partial}{\partial\theta}\mathcal{E}(X;\theta)\right] - \frac{1}{n}\sum_{i=1}^{n}\frac{\partial}{\partial\theta}\mathcal{E}(X_i;\theta)$$

Maximum likelihood learning of the descriptor net

Suppose we observe training examples $\{X_i, i = 1, ..., n\}$.

The maximum likelihood estimation (MLE) seeks to maximize the log-likelihood function:

$$L_p(\theta) = \frac{1}{n} \sum_{i=1}^{n} \log p(X_i; \theta)$$

The gradient of the log-likelihood is:

$$L'_p(\theta) \quad = \quad \mathrm{E}_\theta \left[\frac{\partial}{\partial \theta} \mathcal{E}(X; \theta) \right] - \frac{1}{n} \sum_{i=1}^{n} \frac{\partial}{\partial \theta} \mathcal{E}(X_i; \theta)$$

$\frac{\partial}{\partial \theta} \log Z(\theta) = \mathrm{E}_\theta \left[\frac{\partial}{\partial \theta} \mathcal{E}(X; \theta) \right]$, which is analytically intractable.

We use MCMC (e.g., Langevin dynamics) to sample $\{\tilde{X}_i, i = 1, ..., \tilde{n}\}$ from $p(X;\theta)$. The Langevin revision is:

$$X_{\tau+1} = \underbrace{X_\tau - \frac{\delta^2}{2}\frac{\partial}{\partial X}\mathcal{E}(X_\tau;\theta)}_{\text{find } X \text{ to minimize } \mathcal{E} \text{ via gradient descent}} + \underbrace{\mathrm{N}(0, \delta^2 I_D)}_{\text{brownian motion}}$$

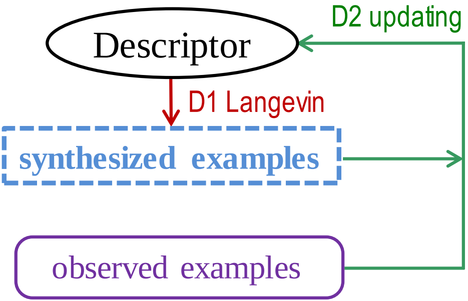
MCMC approximation

We use MCMC (e.g., Langevin dynamics) to sample $\{\tilde{X}_i, i = 1, ..., \tilde{n}\}$ from $p(X;\theta)$. The Langevin revision is:

$$X_{\tau+1} = \underbrace{X_\tau - \frac{\delta^2}{2}\frac{\partial}{\partial X}\mathcal{E}(X_\tau;\theta)}_{\text{find } X \text{ to minimize } \mathcal{E} \text{ via gradient descent}} + \underbrace{\mathrm{N}(0, \delta^2 I_D)}_{\text{brownian motion}}$$

MCMC approximation of the gradient:

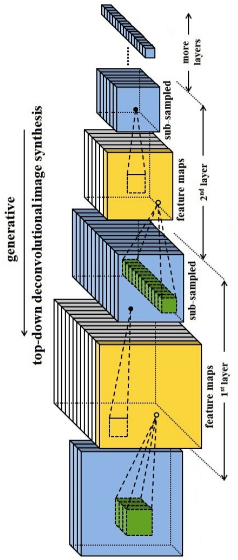
$$L'_p(\theta) \approx \underbrace{\frac{1}{\tilde{n}}\sum_{i=1}^{\tilde{n}} \frac{\partial}{\partial\theta}\mathcal{E}(\tilde{X}_i;\theta)}_{\text{synthesized statistics}} - \underbrace{\frac{1}{n}\sum_{i=1}^{n} \frac{\partial}{\partial\theta}\mathcal{E}(X_i;\theta)}_{\text{observed statistics}}$$



D1: Dreaming

D2: Make the dreaming more realistic

Latent variable model (Generator net)



The generator net [Kingma, Welling, 2013 on VAE; Goodfellow, et al. 2014 on GAN] is in the form of

$$h \sim \mathrm{N}(0, I_d),$$
$$X = g(h; \alpha) + \epsilon, \ \epsilon \sim \mathrm{N}(0, \sigma^2 I_D).$$

$g(h; \alpha)$ is a top-down ConvNet.

Recursive factor analysis, piecewise linear factor analysis

Unsupervised learning: latent factors unknown

MLE learning of generator net

The joint density of generator net is:

$$q(h, X; \alpha) = q(h)q(X|h; \alpha)$$

and

$$\log q(h, X; \alpha) = -\frac{1}{2\sigma^2}\|X - g(h;\alpha)\|^2 - \frac{1}{2}\|h\|^2 + \text{constant}. \quad (1)$$

Given observed data $\{X_i, i = 1, ..., n\}$, MLE learning is to maximize log-likelihood $L_q(\alpha) = \frac{1}{n}\sum_{i=1}^{n} \log q(X_i; \alpha)$.

$$\frac{\partial}{\partial \alpha} \log q(X; \alpha) = \mathrm{E}_{\underbrace{q(h|X;\alpha)}_{\text{posterior}}}\left[\frac{\partial}{\partial \alpha} \log q(h, X; \alpha)\right]. \quad (2)$$

Alternating back-propagation

Langevin inference (sample from posterior $q(h|X;\alpha)$)

$$h_{\tau+1} = h_\tau + \frac{\delta^2}{2}\frac{\partial}{\partial h}\log q(h_\tau, X;\alpha) + N(0, \delta^2 I_D)$$

Gradient of the log-likelihood

$$L_q^{'}(\alpha) \quad = \quad \mathrm{E}_{q(h|X;\alpha)}\left[\frac{\partial}{\partial \alpha}\log q(h, X;\alpha)\right].$$

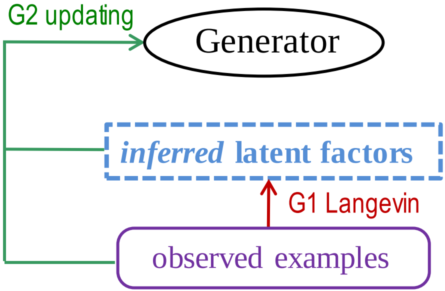
Alternating back-propagation

Langevin inference (sample from posterior $q(h|X;\alpha)$)

$$h_{\tau+1} = h_\tau + \frac{\delta^2}{2}\frac{\partial}{\partial h}\log q(h_\tau, X;\alpha) + N(0, \delta^2 I_D)$$

Gradient of the log-likelihood

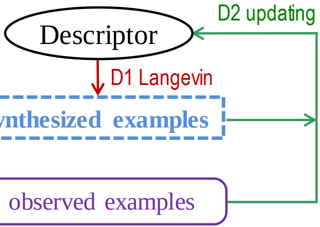
$$\begin{aligned} L_q'(\alpha) &= \mathrm{E}_{q(h|X;\alpha)}\left[\frac{\partial}{\partial\alpha}\log q(h, X;\alpha)\right]. \\ &\approx \frac{1}{n}\sum_{i=1}^{n}\frac{\partial}{\partial\alpha}\log q(h_i, X_i;\alpha) \\ &= \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\sigma^2}\left(X_i - g(h_i;\alpha)\right)\frac{\partial}{\partial\alpha}g(h_i;\alpha), \end{aligned}$$



G1: thinking, explaining-away reasoning

G2: make the thinking more accurate

Difficulty in learning descriptor net

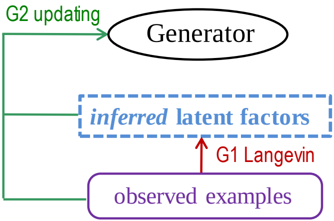


D1 needs to dream hard, but generator is a better dreamer

D2 learns without thinking

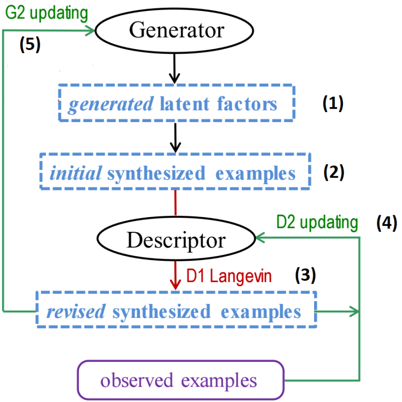
Difficulty in learning generator net

Jianwen Xie [1,2], Yang Lu [2,3], Ruiqi Gao [2], Ying Nian Wu [2]



G1 needs to think hard, but descriptor does not need to think
G2 learns only if latent factors known

Cooperative learning (CoopNets)



(Step 1,2) Generator $\rightarrow$ initial draft;

(Step 3) Descriptor $\rightarrow$ revised draft

(Step 4) Descriptor shifts from revised towards observed

(Step 5) Generator reconstructs the revised, knowing latent factors.

Analogy

Student (generator) writes initial draft of a paper.
Teacher (descriptor) revises the paper.
Student learns from revision.
Teacher learns from outside review (observed data).
Teacher guides the student, but the student does the most of the work.

Experiment

Exp 1: Texture synthesis

Figure: For each category, the first image is the observed image. The other three are synthesized images by CoopNets.

Figure: Synthesized images generated by the model learned from 10,000 face images.

Exp 3: Scene synthesis

Figure: The left pannel displays the observed images randomly selected from the dataset. The right pannel displays some synthesized images by the CoopNets algorithm.

Jianwen Xie [1,2], Yang Lu [2,3], Ruiqi Gao [2], Ying Nian Wu [2]

Exp 3: Scene synthesis

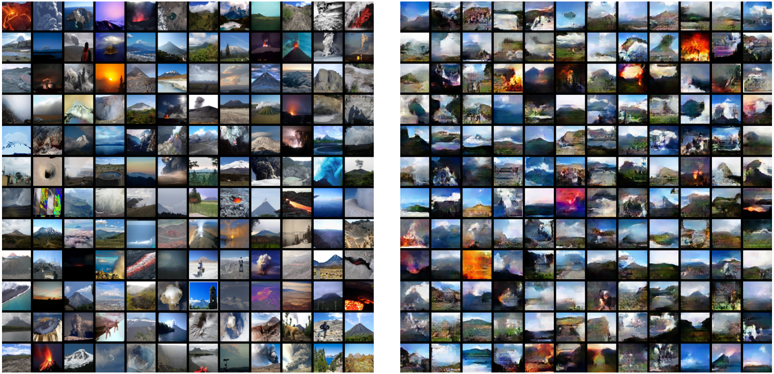


Figure: The left pannel displays the observed images randomly selected from the dataset. The right pannel displays some synthesized images by the CoopNets algorithm.

Exp 3: Scene synthesis

Jianwen Xie [1,2], Yang Lu [2,3], Ruiqi Gao [2], Ying Nian Wu [2]

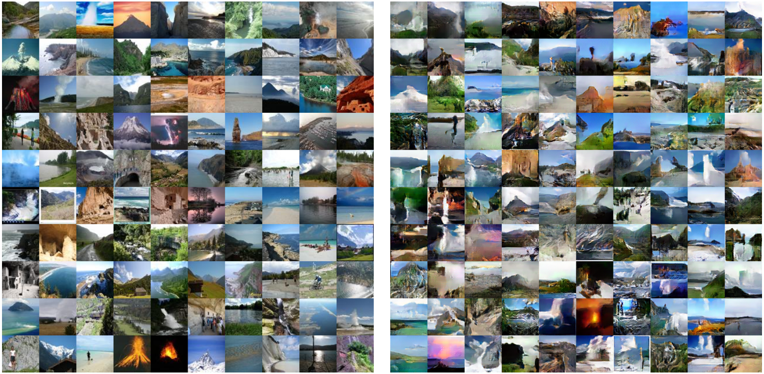


Figure: Images generated by CoopNets learned from 10 Imagenet scene categories. The training set consists of 1100 images randomly sampled from each category.

Exp 4: Interpolation

We learn smooth generator model that traces the manifold of the data distribution.

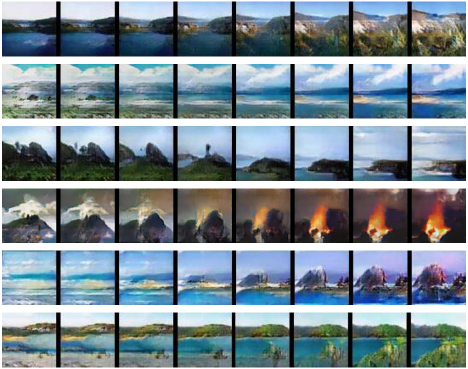


Figure: Interpolation in the latent space by the generator net learned by cooperative training algorithm

Exp 5: Synthesis quality

We evaluate the synthesis quality by the Inception score [Salimans et al., 2016], and make a comparison with some baseline methods.

Table: Inception scores of different methods on learning from 10 Imagenet scene categories. n is the number of training images randomly sampled from each category.

	$n = 50$	$n = 100$	$n = 300$	$n = 500$	$n = 700$	$n = 900$	$n = 1100$
CoopNets	**2.66±.13**	**3.04±.13**	**3.41±.13**	**3.48±.08**	**3.59±.11**	**3.65±.07**	**3.79±.15**
DCGAN	2.26±.16	2.50±.15	3.16±.15	3.05±.12	3.13±.09	3.34±.05	3.47±.06
EBGAN	2.23±.17	2.40±.14	2.62±.08	2.46±.09	2.65±.04	2.64±.04	2.75±.08
W-GAN	1.80±.09	2.19±.12	2.34±.06	2.62±.08	2.86±.10	2.88±.07	3.14±.06
VAE	1.62±.09	1.63±.06	1.65±.05	1.73±.04	1.67±.03	1.72±.02	1.73±.02
InfoGAN	2.21±.04	1.73±.01	2.15±.03	2.42±.05	2.47±.05	2.29±.03	2.08±.04
Method of	2.44±.27	2.38±.13	2.42±.09	2.94±.11	3.02±.06	3.08±.08	3.15±.06
Algorithm G	1.72±.07	1.94±.09	2.32±.09	2.40±.06	2.45±.05	2.54±.05	2.61±.06
Persistent CD	1.30±.08	1.94±.03	1.80±.02	1.53±.02	1.45±.04	1.35±.02	1.51±.02

Other criteria to evaluate the synthesis quality:

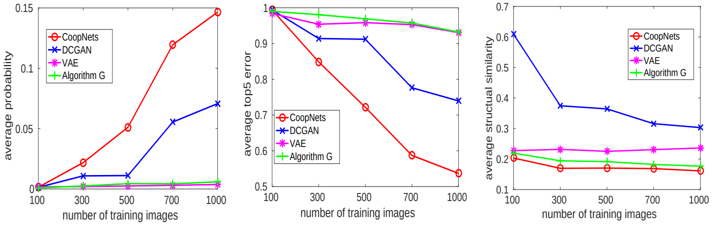


Figure: **Left**: Average softmax class probability on single Imagenet category versus the number of training images. **Middle**: Top 5 classification error. **Right**: Average pairwise structural similarity.

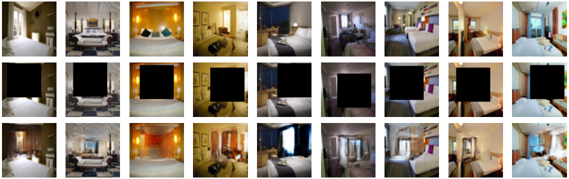
We conduct an experiment on learning from face images, and then testing the learned model on completing the occluded testing images.

Figure: The first row displays the original images. The second row displays the images with occluded pixels. The third row displays the recovered images by the learned generator net.

Table: Comparison of recovery performances of different methods in 3 experiments

	task	CoopNets	DCGAN	MRFℓ_1	MRFℓ_2	inter1	inter2	inter3	inter4	inter5
error	M30	**0.115**	0.211	0.132	0.134	0.120	0.120	0.265	0.120	0.120
	M40	**0.124**	0.212	0.148	0.149	0.135	0.135	0.314	0.135	0.135
	M50	**0.136**	0.214	0.178	0.179	0.170	0.166	0.353	0.164	0.164
PSNR	M30	**16.893**	12.116	15.739	15.692	16.203	16.635	9.524	16.665	16.648
	M40	**16.098**	11.984	14.834	14.785	15.065	15.644	8.178	15.698	15.688
	M50	**15.105**	11.890	13.313	13.309	13.220	14.009	7.327	14.164	14.161

Exp 6: Pattern completion



hotel room

forest road

Figure: The first row displays the original images. The second row displays the images with occluded pixels. The third row displays the recovered images by the learned generator net.

Conclusion

Conclusion

- Two networks feed each other the synthesized data in the learning process, including initial, revised, and reconstructed synthesized data.

Conclusion

+ Two networks feed each other the synthesized data in the learning process, including initial, revised, and reconstructed synthesized data.
+ The learning process interweaves the existing maximum likelihood learning algorithms for the two networks.

Conclusion

+ Two networks feed each other the synthesized data in the learning process, including initial, revised, and reconstructed synthesized data.
+ The learning process interweaves the existing maximum likelihood learning algorithms for the two networks.
+ The MCMC transitions for the descriptor are memorized and reproduced by the generator via ancestral sampling.

Conclusion

- Two networks feed each other the synthesized data in the learning process, including initial, revised, and reconstructed synthesized data.
- The learning process interweaves the existing maximum likelihood learning algorithms for the two networks.
- The MCMC transitions for the descriptor are memorized and reproduced by the generator via ancestral sampling.
- MCMC teaching by the descriptor model turns the unsupervised learning of the generator model into supervised learning.

Thank you!