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

\[
\begin{align*}
\text{Bottom-up ConvNet} & \quad \text{Top-down ConvNet} \\
\text{energy} & \quad \text{latent variables} \\
\uparrow & \quad \downarrow \\
\text{signal} & \quad \text{signal} \\
(a) \text{ Descriptor Net} & \quad (b) \text{ Generator Net} \\
(\text{Energy-based Model}) & \quad (\text{Latent Variable Model})
\end{align*}
\]
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
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
The descriptor net [Xie et al. 2016] is in the form of exponential tilting of a reference distribution:

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

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 \( \{X_i, i = 1, \ldots, n\} \).

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 \( \{X_i, i = 1, \ldots, n\} \).

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:

\[
L'_p(\theta) = \mathbb{E}_{\theta} \left[ \frac{\partial}{\partial \theta} \mathcal{E}(X; \theta) \right] - \frac{1}{n} \sum_{i=1}^{n} \frac{\partial}{\partial \theta} \mathcal{E}(X_i; \theta)
\]
Maximum likelihood learning of the descriptor net

Suppose we observe training examples \( \{X_i, i = 1, \ldots, n\} \).

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:

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

\[
\frac{\partial}{\partial \theta} \log Z(\theta) = \mathbb{E}_\theta \left[ \frac{\partial}{\partial \theta} \mathcal{E}(X; \theta) \right], \text{ which is analytically intractable.}
\]
We use MCMC (e.g., Langevin dynamics) to sample 
\( \{\tilde{X}_i, i = 1, \ldots, \tilde{n}\} \) from \( p(X; \theta) \). The Langevin revision is:

\[
X_{\tau+1} = X_{\tau} - \frac{\delta^2}{2} \frac{\partial}{\partial X} \mathcal{E}(X_{\tau}; \theta) + \mathcal{N}(0, \delta^2 I_D)
\]

find \( X \) to minimize \( \mathcal{E} \) via gradient descent

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

\[
X_{\tau+1} = X_\tau - \frac{\delta^2}{2} \frac{\partial}{\partial X} \mathcal{E}(X_\tau; \theta) + \mathcal{N}(0, \delta^2 I_D)
\]

find \( X \) to minimize \( \mathcal{E} \) via gradient descent

MCMC approximation of the gradient:

\[
L'_p(\theta) \approx \frac{1}{\tilde{n}} \sum_{i=1}^{\tilde{n}} \frac{\partial}{\partial \theta} \mathcal{E}(\tilde{X}_i; \theta) - \frac{1}{n} \sum_{i=1}^{n} \frac{\partial}{\partial \theta} \mathcal{E}(X_i; \theta)
\]
CoopNets

Jianwen Xie$^{1,2}$, Yang Lu$^{2,3}$, Ruiqi Gao$^2$, Ying Nian Wu$^2$

Introduction

Method
Energy-based model
Latent variable model
Cooperative Learning

Experiment
Texture synthesis
Object synthesis
Scene synthesis
Interpolation
Synthesis quality
Pattern completion

Conclusion

Analysis by synthesis

Descriptor

D1: Dreaming
D2: Make the dreaming more realistic

D1 Langevin

observed examples

synthesized examples

D2 updating
The generator net [Kingma, Welling, 2013 on VAE; Goodfellow, et al. 2014 on GAN] is in the form of

\[ h \sim N(0, I_d), \]
\[ X = g(h; \alpha) + \epsilon, \quad \epsilon \sim N(0, \sigma^2 I_D). \]

\( g(h; \alpha) \) 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

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

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

\[
\frac{\partial}{\partial \alpha} \log q(X; \alpha) = \mathbb{E}_{q(h|X; \alpha)} \left[ \frac{\partial}{\partial \alpha} \log q(h, X; \alpha) \right]. \tag{2}
\]
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

\[
L'_q(\alpha) = \mathbb{E}_{q(h|X; \alpha)} \left[ \frac{\partial}{\partial \alpha} \log q(h, X; \alpha) \right].
\]
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

$$L'_q(\alpha) = E_{q(h|X;\alpha)} \left[ \frac{\partial}{\partial \alpha} \log q(h, X;\alpha) \right].$$

$$\approx \frac{1}{n} \sum_{i=1}^{n} \frac{\partial}{\partial \alpha} \log q(h_i, X_i;\alpha)$$

$$= \frac{1}{n} \sum_{i=1}^{n} \frac{1}{\sigma^2} (X_i - g(h_i;\alpha)) \frac{\partial}{\partial \alpha} g(h_i;\alpha),$$
Analysis by inference

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
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.
Exp 2: Object synthesis

**Figure**: Synthesized images generated by the model learned from 10,000 face images.
Exp 3: Scene synthesis

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

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.
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.

<table>
<thead>
<tr>
<th>Method</th>
<th>( n = 50 )</th>
<th>( n = 100 )</th>
<th>( n = 300 )</th>
<th>( n = 500 )</th>
<th>( n = 700 )</th>
<th>( n = 900 )</th>
<th>( n = 1100 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoopNets</td>
<td>2.66±.13</td>
<td>3.04±.13</td>
<td>3.41±.13</td>
<td>3.48±.08</td>
<td>3.59±.11</td>
<td>3.65±.07</td>
<td>3.79±.15</td>
</tr>
<tr>
<td>DCGAN</td>
<td>2.26±.16</td>
<td>2.50±.15</td>
<td>3.16±.15</td>
<td>3.05±.12</td>
<td>3.13±.09</td>
<td>3.34±.05</td>
<td>3.47±.06</td>
</tr>
<tr>
<td>EBGAN</td>
<td>2.23±.17</td>
<td>2.40±.14</td>
<td>2.62±.08</td>
<td>2.46±.09</td>
<td>2.65±.04</td>
<td>2.64±.04</td>
<td>2.75±.08</td>
</tr>
<tr>
<td>W-GAN</td>
<td>1.80±.09</td>
<td>2.19±.12</td>
<td>2.34±.06</td>
<td>2.62±.08</td>
<td>2.86±.10</td>
<td>2.88±.07</td>
<td>3.14±.06</td>
</tr>
<tr>
<td>VAE</td>
<td>1.62±.09</td>
<td>1.63±.06</td>
<td>1.65±.05</td>
<td>1.73±.04</td>
<td>1.67±.03</td>
<td>1.72±.02</td>
<td>1.73±.02</td>
</tr>
<tr>
<td>InfoGAN</td>
<td>2.21±.04</td>
<td>1.73±.01</td>
<td>2.15±.03</td>
<td>2.42±.05</td>
<td>2.47±.05</td>
<td>2.44±.05</td>
<td>2.08±.04</td>
</tr>
<tr>
<td>Method of</td>
<td>2.44±.27</td>
<td>2.38±.13</td>
<td>2.42±.09</td>
<td>2.94±.11</td>
<td>3.02±.06</td>
<td>3.08±.08</td>
<td>3.15±.06</td>
</tr>
<tr>
<td>Algorithm G</td>
<td>1.72±.07</td>
<td>1.94±.09</td>
<td>2.32±.09</td>
<td>2.40±.06</td>
<td>2.45±.05</td>
<td>2.54±.05</td>
<td>2.61±.06</td>
</tr>
<tr>
<td>Persistent CD</td>
<td>1.30±.08</td>
<td>1.94±.03</td>
<td>1.80±.02</td>
<td>1.53±.02</td>
<td>1.45±.04</td>
<td>1.35±.02</td>
<td>1.51±.02</td>
</tr>
</tbody>
</table>
Exp 5: Synthesis quality

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.
Exp 6: Pattern completion

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.
Exp 6: Pattern completion

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

<table>
<thead>
<tr>
<th>task</th>
<th>CoopNets</th>
<th>DCGAN</th>
<th>MRF$\ell_1$</th>
<th>MRF$\ell_2$</th>
<th>inter1</th>
<th>inter2</th>
<th>inter3</th>
<th>inter4</th>
<th>inter5</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>M30</td>
<td>0.115</td>
<td>0.211</td>
<td>0.132</td>
<td>0.120</td>
<td>0.120</td>
<td>0.265</td>
<td>0.120</td>
<td>0.120</td>
</tr>
<tr>
<td></td>
<td>M40</td>
<td>0.124</td>
<td>0.212</td>
<td>0.148</td>
<td>0.135</td>
<td>0.135</td>
<td>0.314</td>
<td>0.135</td>
<td>0.135</td>
</tr>
<tr>
<td></td>
<td>M50</td>
<td>0.136</td>
<td>0.214</td>
<td>0.178</td>
<td>0.170</td>
<td>0.166</td>
<td>0.353</td>
<td>0.164</td>
<td>0.164</td>
</tr>
</tbody>
</table>
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!