Advanced Deep Learning

Confronting the Partition Function

U Kang
Seoul National University

In This Lecture

- The Log-Likelihood Gradient
- Stochastic Maximum Likelihood and Contrastive Divergence
- Pseudolikelihood
- Noise-Contrastive Estimation
- Estimating the Partition Function

Motivation

- Consider an unnormalized probability distribution:

$$\tilde{p}(\mathbf{x}; \boldsymbol{\theta}).$$

- That we normalize using a partition function:

$$Z(\boldsymbol{\theta}) = \int \tilde{p}(\boldsymbol{x}) d\boldsymbol{x} \quad \text{or} \quad \sum_{\boldsymbol{x}} \tilde{p}(\boldsymbol{x}).$$

  - Intractable for many interesting models

→ We discuss techniques that one can use in the training and the evaluation of models with intractable partition functions

Outline

- **The Log-Likelihood Gradient**
- Stochastic Maximum Likelihood and Contrastive Divergence
- Pseudolikelihood
- Noise-Contrastive Estimation
- Estimating the Partition Function

Log-Likelihood Gradient (LLG)

- The LLG is written:

$$\nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}; \boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\mathbf{x}; \boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}} \log Z(\boldsymbol{\theta}).$$

Decomposition in positive phase and negative phase, which are phases of learning.

Log-Likelihood Gradient (LLG) Identity over Discrete x

$$\nabla_{\boldsymbol{\theta}} \log Z = \frac{\nabla_{\boldsymbol{\theta}} Z}{Z} = \frac{\nabla_{\boldsymbol{\theta}} \sum_{\mathbf{x}} \tilde{p}(\mathbf{x})}{Z} = \frac{\sum_{\mathbf{x}} \nabla_{\boldsymbol{\theta}} \tilde{p}(\mathbf{x})}{Z}$$

$$= \frac{\sum_{\mathbf{x}} \nabla_{\boldsymbol{\theta}} \exp\left(\log \tilde{p}(\mathbf{x})\right)}{Z}$$

$$= \frac{\sum_{\mathbf{x}} \exp\left(\log \tilde{p}(\mathbf{x})\right) \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\mathbf{x})}{Z}$$

$$= \frac{\sum_{\mathbf{x}} \tilde{p}(\mathbf{x}) \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\mathbf{x})}{Z} = \sum_{\mathbf{x}} p(\mathbf{x}) \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\mathbf{x})$$

$$= \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x})} \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\mathbf{x})$$

Log-Likelihood Gradient (LLG)
A Very Useful Identity

- This identity $\nabla_{\boldsymbol{\theta}} \log Z = \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x})} \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\mathbf{x})$ is very useful as it is the basis for many Monte Carlo methods. It plays a big role in the approximate maximizing of the likelihood of models with intractable partition function.

Log-Likelihood Gradient (LLG)

- In the Monte Carlo approach to learning models:
  - increase the log of the unnormalized distribution for x drawn from the data **in the positive phase.**
  - decrease the partition function by decreasing this same log but drawn from the model distribution **in the negative phase**.

$$\nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}; \boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\mathbf{x}; \boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}} \log Z(\boldsymbol{\theta}).$$

$$\nabla_{\boldsymbol{\theta}} \log Z = \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x})} \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\mathbf{x})$$

Outline

- ☑ The Log-Likelihood Gradient
- ☐ **Stochastic Maximum Likelihood and Contrastive Divergence**
- ☐ Pseudolikelihood
- ☐ Noise-Contrastive Estimation
- ☐ Estimating the Partition Function

Contrastive Divergence and Stochastic Maximum Likelihood

$$\nabla_{\boldsymbol{\theta}} \log Z = \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x})} \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\mathbf{x})$$

- How to implement this equation?
- We will see three approaches:
  1. A naive Monte Carlo Markov Chain (MCMC) algorithm
  2. The contrastive divergence algorithm
  3. The stochastic maximum likelihood algorithm

A Naive MCMC Algorithm

Algorithm 18.1 A naive MCMC algorithm for maximizing the log-likelihood with an intractable partition function using gradient ascent

Set ϵ, the step size, to a small positive number.

Set k, the number of Gibbs steps, high enough to allow burn in. Perhaps 100 to train an RBM on a small image patch.

while not converged **do**

  Sample a minibatch of m examples $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\}$ from the training set

  $\mathbf{g} \leftarrow \frac{1}{m} \sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\mathbf{x}^{(i)}; \boldsymbol{\theta})$.

  Initialize a set of m samples $\{\tilde{\mathbf{x}}^{(1)}, \ldots, \tilde{\mathbf{x}}^{(m)}\}$ to random values (e.g., from a uniform or normal distribution, or possibly a distribution with marginals matched to the model's marginals).

  for $i = 1$ to k **do**

    for $j = 1$ to m **do**

      $\tilde{\mathbf{x}}^{(j)} \leftarrow$ gibbs_update($\tilde{\mathbf{x}}^{(j)}$).

    end for

  end for

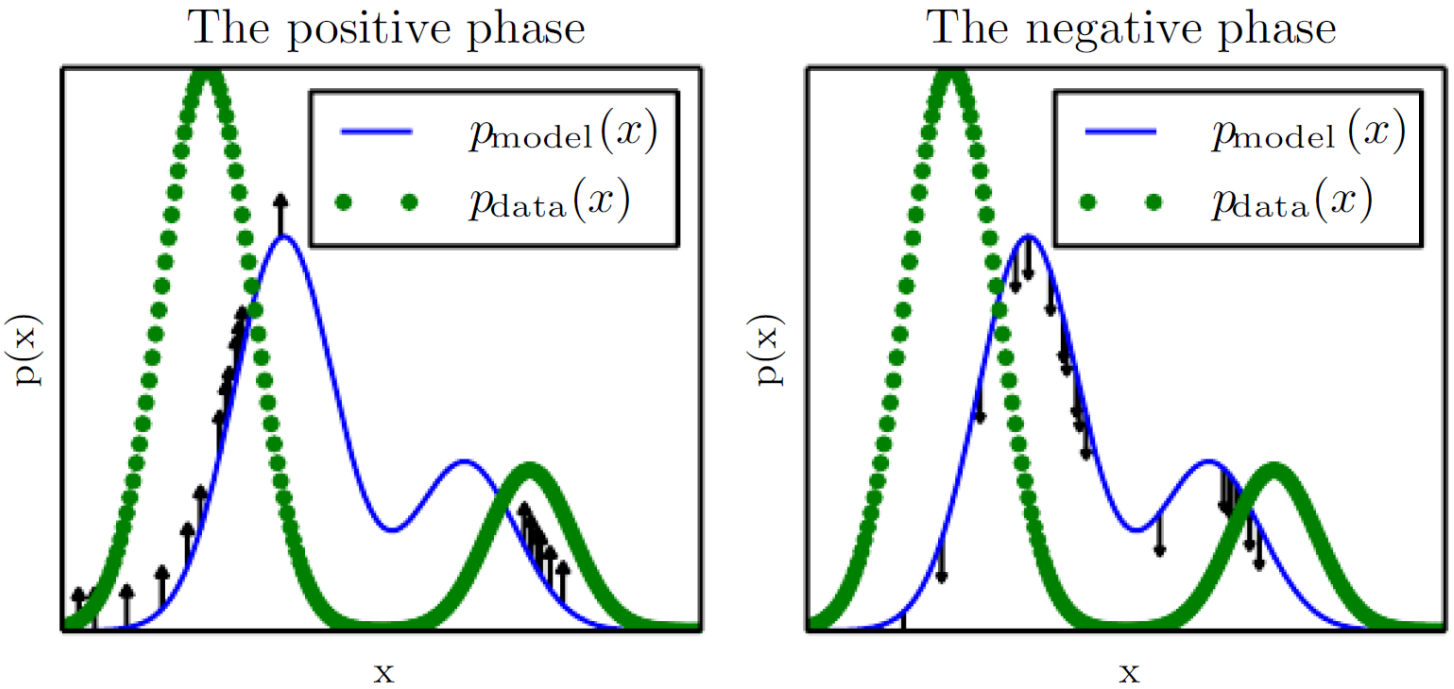
  $\mathbf{g} \leftarrow \mathbf{g} - \frac{1}{m} \sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\tilde{\mathbf{x}}^{(i)}; \boldsymbol{\theta})$.

  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \epsilon \mathbf{g}$.

end while

A Naive MCMC Algorithm

- Interpretation
  - Trying to achieve balance between two forces, one pushing up on the model distribution where the data occurs, and another pushing down on the model distribution where the model samples occur



A Naive MCMC Algorithm

- Random initialization each time we need the gradient
- We obtain accurate result, but the cost is so high that it is computationally infeasible
- This comes mainly from the random initialization at each step
- The other algorithms try to approximate this procedure

Contrastive Divergence (CD)

- As initializing randomly was way too costly, the CD algorithm initializes the Markov chain at each step with samples from the data distribution.
- As they are available in the data set, this gives us a free initialization.

But:

-Initially, the negative phase is not accurate because the model distribution and the data distribution are not close: we need to wait for the positive phase to increase the model's probability for the data

-Then, when the positive phase had time to act, the negative phase start becoming accurate.

The Contrastive Divergence (CD) Algorithm

Algorithm 18.2 The contrastive divergence algorithm, using gradient ascent as the optimization procedure

Set ϵ, the step size, to a small positive number.
Set k, the number of Gibbs steps, high enough to allow a Markov chain sampling from $p(\mathbf{x};\boldsymbol{\theta})$ to mix when initialized from p_{data}. Perhaps 1–20 to train an RBM on a small image patch.
while not converged **do**
  Sample a minibatch of m examples $\{\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(m)}\}$ from the training set
  $\mathbf{g} \leftarrow \frac{1}{m}\sum_{i=1}^{m}\nabla_{\boldsymbol{\theta}}\log\tilde{p}(\mathbf{x}^{(i)};\boldsymbol{\theta})$.
  for $i = 1$ to m **do**
    $\tilde{\mathbf{x}}^{(i)} \leftarrow \mathbf{x}^{(i)}$.
  end for
  for $i = 1$ to k **do**
    for $j = 1$ to m **do**
      $\tilde{\mathbf{x}}^{(j)} \leftarrow \text{gibbs\_update}(\tilde{\mathbf{x}}^{(j)})$.
    end for
  end for
  $\mathbf{g} \leftarrow \mathbf{g} - \frac{1}{m}\sum_{i=1}^{m}\nabla_{\boldsymbol{\theta}}\log\tilde{p}(\tilde{\mathbf{x}}^{(i)};\boldsymbol{\theta})$.
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \epsilon\mathbf{g}$.
end while

The CD Algorithm

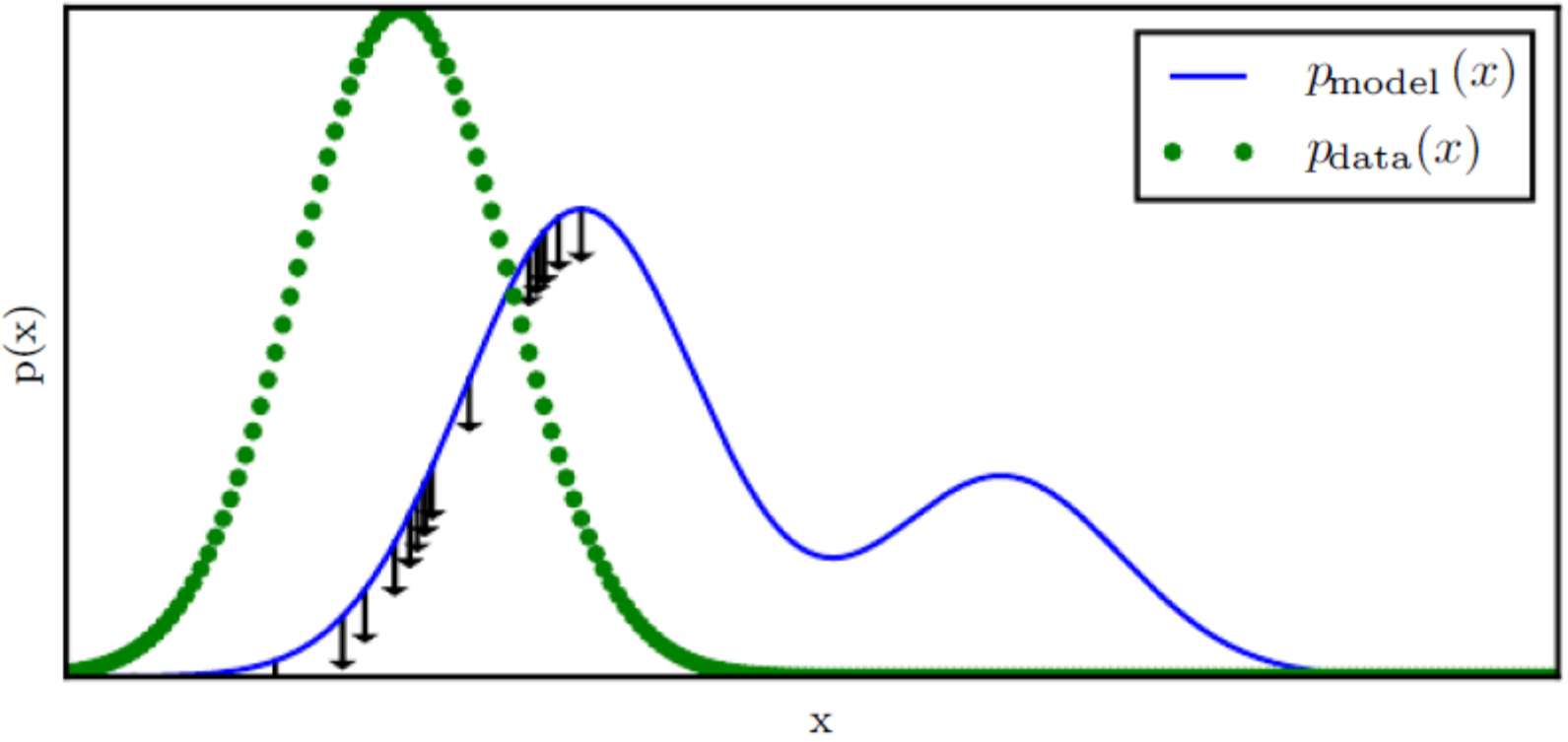
- Why CD fails to implement the correct negative phase:

→ It fails to suppress spurious modes (*i.e.* modes that are present in the model distribution but absent in the data distribution)

→ This is because modes in the model distribution that are far from the data distribution will not be visited by Markov chains initialized at training points, unless k is very large

The CD Algorithm

- How the negative phase of CD can fail to supress spurious modes



Stochastic Maximum Likelihood (SML)

Algorithm 18.3 The stochastic maximum likelihood / persistent contrastive divergence algorithm using gradient ascent as the optimization procedure

Set ϵ, the step size, to a small positive number.

Set k, the number of Gibbs steps, high enough to allow a Markov chain sampling from $p(\mathbf{x}; \boldsymbol{\theta} + \epsilon \mathbf{g})$ to burn in, starting from samples from $p(\mathbf{x}; \boldsymbol{\theta})$. Perhaps 1 for RBM on a small image patch, or 5–50 for a more complicated model like a DBM.

Initialize a set of m samples $\{\tilde{\mathbf{x}}^{(1)}, \ldots, \tilde{\mathbf{x}}^{(m)}\}$ to random values (e.g., from a uniform or normal distribution, or possibly a distribution with marginals matched to the model's marginals).

while not converged **do**

  Sample a minibatch of m examples $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\}$ from the training set

  $\mathbf{g} \leftarrow \frac{1}{m} \sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\mathbf{x}^{(i)}; \boldsymbol{\theta})$.

  for $i = 1$ to k **do**

    for $j = 1$ to m **do**

      $\tilde{\mathbf{x}}^{(j)} \leftarrow \text{gibbs\_update}(\tilde{\mathbf{x}}^{(j)})$.

    end for

  end for

  $\mathbf{g} \leftarrow \mathbf{g} - \frac{1}{m} \sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\tilde{\mathbf{x}}^{(i)}; \boldsymbol{\theta})$.

  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \epsilon \mathbf{g}$.

end while

Stochastic Maximum Likelihood (SML)

- Strategy introduced to resolve problems of CD (failing to suppress spurious modes)
- Main difference: the Markov chains are initialized at each gradient step with their state from the previous gradient step.
  - This is called the **SML approach**.
  - It was first discovered under the name SML in the applied mathematics and statistics community,
  - And then independently rediscovered, under the name of **persistent contrastive divergence** (PCD, PCD-k) in the deep learning community.

Stochastic Maximum Likelihood (SML)

- Basic idea: as long as the steps taken by the algo are small, the model from the previous step will be similar to the model from the current step.

→ Therefore, the samples from the previous models distribution will be very close to being fair samples from the current distribution.

→ So a Markov chain initialized with these samples will not require much time to mix.

Stochastic Maximum Likelihood (SML)

Advantages:

- Each Markov chain is continually updated throughout the learning process, and not restarted at each step of the gradient as with the previous algos:

→ The chains are free to wander far enough to find all the model's modes.

→ Therefore, SML is considerably more resistent than CD to forming models with spurious modes.

Stochastic Maximum Likelihood (SML)

Drawback:

- SML is vulnerable to becoming inaccurate if the algo can move faster than the Markov chain can mix between steps.
- → It can happen when the number of gibbs update is too small or the step size too large.
- → But there is no rule independant of the problems to determine the acceptable range for those parameters

Stochastic Maximum Likelihood (SML)

Be careful:

- When one evaluates the samples from model trained with the SML, be careful!

→ The samples have to be drawn starting from a fresh Markov chain itself initialized from a random starting point after the model is done training

→ The samples present in the persistent negative chains used for training have been influenced by several recent versions of the model

→ **If this is not done, the model can appear as having greater capacity than it actually does.**

Questions?