Towards Better Representations with Deep/Bayes Learning

Chunyuan Li
Duke University
http://chunyuan.li
Overview

1. **Motivations**
2. **Bayesian Deep Learning**
   - Stochastic Gradient MCMCs
   - Weight uncertainty in DNNs
   - Connection to Dropout
3. **Deep Bayesian Learning**
   - Non-identifiable issues
   - ALICE algorithms
   - Unified views for bivariate GANs
4. **Intrinsic Dimension of Objective Landscape**
   - Definitions
   - Empirical Results
5. **Summary**
**Introduction**

Bayesian Deep Learning  
Deep Bayesian Learning  
Intrinsic Dimension

**Popular research topics**

**Deep (Neural Nets) & Bayesian Learning**

*Popularity of topics*

1987  
Dynamic topic modeling on NIPS papers  
2015

**Deep**  
- CNNs  
- RNNs  
- dropout  
- batch norm.

**Bayes**  
- prior  
- posterior  
- generative  
- MCMC

Geoffrey Hinton  
Thomas Bayes

Figures adapted from Teh’s talk at NIPS 2017
Increasing flexibility for representations

**Deep Learning**

- Input: $x$
- Hidden: $h$
- Output: $y$

*Shallow models (e.g., logistic regression)*

- Input: $x$
- Hidden: $h$
- Output: $y$

*Deep models (e.g., multi-layer perceptron)*

**Bayesian Learning**

- Point estimate (e.g., SGD)
- Full distribution (e.g., MCMC)
Q: Bring the best of both worlds?
A: New research topics:

- **Bayesian Deep Learning**
  
  Scalable Bayesian methods for the weight uncertainty of DNNs

- **Deep Bayesian Learning**
  
  DNNs as flexible representation methods in Bayesian models.

Seek further understanding?

- **Intrinsic Dimension of Objective Landscape**
Bayesian Deep Learning
Problem Setup

- Given data $\mathcal{D} = \{d_i\}_{i=1}^{N}$; $d_i \triangleq (x_i, y_i)$ in DNNs,
- A model with parameters $\theta$

$$\underbrace{p(\theta|\mathcal{D}) \propto p(\theta) \prod_{i=1}^{N} p(d_i|\theta)}_{\text{Posterior}} \underbrace{\text{Prior}}_{\text{Likelihood}}$$

For testing input, Bayesian predictive distribution

$$p(y|x, \mathcal{D}) = \mathbb{E}_{p(\theta|\mathcal{D})}[p(y|x, \theta)]$$

Figures adapted from Teh’s talk at NIPS 2017
In optimization, the single "best" point on training is used

\[ \theta_{MAP} = \arg\max_{\theta} \log p(\theta | \mathcal{D}) = \arg\max_{\theta} U(\theta) \]

\[ U(\theta) \triangleq - \sum_{i=1}^{N} \log p(d_i | \theta) - \log p(\theta) \]

Stochastic approximation \( \tilde{U}(\theta) \approx U(\theta) \)

The MAP approximates this expectation as

\[ p(y|x, \mathcal{D}) = \mathbb{E}_{p(\theta | \mathcal{D})}[p(y|x, \theta)] \approx p(y|x, \theta_{MAP}) \]

Parameter uncertainty is ignored
In Bayesian, the full posterior distribution after observing training set is used for prediction.

\[
p(y|x, \mathcal{D}) = \mathbb{E}_{p(\theta|\mathcal{D})}[p(y|x, \theta)] \approx \frac{1}{T} \sum_{t=1}^{T} p(y|x, \theta_t)
\]
SGLD vs. SGD

- Stochastic Gradient Langevin Dynamics (SGLD) draws samples:

\[
\theta_{t+1} = \theta_t - \epsilon_t \tilde{f}_t + \sqrt{2\epsilon_t} \xi_t
\]

where

- Step size: \( \epsilon_t \)
- Stochastic gradient: \( \tilde{f}_t = \nabla \tilde{U}_t(\theta) \)
- Gaussian noise: \( \xi_t \sim \mathcal{N}(0, I) \)

- SGLD is the SG-MCMC analog to SGD

\[
\theta_{t+1} = \theta_t - \epsilon_t \tilde{f}_t
\]
Sampling Dynamics

Approximated Histogram

Sampling Procedure of SGLD
<table>
<thead>
<tr>
<th>Algorithms</th>
<th>SG-MCMC</th>
<th>Optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic</td>
<td>SGLD</td>
<td>SGD</td>
</tr>
<tr>
<td>Precondition</td>
<td>pSGLD</td>
<td>Adam/RMSprop/Adagrad</td>
</tr>
<tr>
<td>Momentum</td>
<td>SGHMC</td>
<td>SGD with momentum</td>
</tr>
<tr>
<td>Thermostat</td>
<td>SGNHT</td>
<td>Santa</td>
</tr>
</tbody>
</table>

**C Li, C Chen, D Carlson, L Carin. AAAI 2016.** [Oral Presentation](#)
Preconditioned Stochastic Gradient Langevin Dynamics for Deep Neural Networks

C Chen, D Carlson, Z. Gan, C Li, L Carin. [AISTATS 2016. Oral Presentation](#)
Bridging the Gap between Stochastic Gradient MCMC and Stochastic Optimization
Preconditioned SGLD

Leverage historical gradients to construct a preconditioner

- Preconditioner: approximate geometry information.
- Preconditioner constructed as diagonal matrix.
- Adjust the step size, according to the local geometry.

Any preconditioning optimization algorithms (e.g., RMSprop/Adagrad/K-FAC) as scalable sampling methods
Toy distribution

- $\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} a & 0 \\ 0 & 1 \end{bmatrix}\right)$. The goal is to estimate the covariance matrix.

- When the covariance matrix of a target distribution is mildly rescaled, we do not have to choose a new step size for pSGLD.

(a) $a = 0.16, \epsilon = 0.3$
(b) Scale up $a = 2, \epsilon = 0.3$
(c) Scale down $a = 0.5, \epsilon = 0.3$
Modern architectures and domains

- CNNs in Computer Vision
- RNNs in Natural Language Processing

Advantages

- Prevent Over-fitting
- Uncertainty in Predictions

C Li, A Stevens, C Chen, Y Pu, Z. Gan, L Carin. CVPR 2016, Spotlight Presentation
Learning Weight Uncertainty with Stochastic Gradient MCMC for Shape Classification

Z. Gan*, C Li*, C Chen, Y Pu, Q Su, L Carin. ACL 2017, Oral Presentation
Scalable Bayesian Learning of Recurrent Neural Networks for Language Modeling
Advantage 1: Prevent Over-fitting

- **Interpretation of Dropout**
  - Gaussian Dropout as $SG-MCMC$
  - Binary Dropout combined with $SG-MCMC$

\[
dropout \subset dropconnect, \quad \text{Binary dropout} \approx \text{Gaussian dropout}
\]

By combining dropConnect and Gaussian corruption, the update rule:

\[
\theta_{t+1} = \xi_0 \odot \theta_t - \frac{\epsilon}{2} \tilde{f}_t = \theta_t - \frac{\epsilon}{2} \tilde{f}_t + \xi_0',
\]

where \(\xi_0' \sim \mathcal{N} \left(0, \frac{p}{(1-p)} \text{diag}(\theta_t^2)\right)\)

- In training: Dropout/DropConnect and SGLD share the same form of update rule, with the difference being that the level of injected noise is different
- In testing: Bayesian model averaging; Fast approximation in Dropout
Advantage 1: Prevent Over-fitting

- **Performance**
  - Optimization converges faster on training, but overfit
  - Uncertainty learned in training prevent over-fitting on testing
Advantage 2: Uncertainty in Prediction

- **Beyond Prediction Means**
  - Uncertainty is the std of multiple predictions
  - High uncertainty predictions tend to be on the boundary of mainfolds

*t-SNE embedding of prediction mean and std*
Deep Bayesian Learning

C Li, H Liu, C Chen, Y Pu, L. Chen, R Henao, L Carin. NIPS 2017
ALICE: Towards Understanding Adversarial Learning for Joint Distribution Matching
Deep Generative Models

**T1: Latent Variable Inference**

\[ X \rightarrow Z \rightarrow X \]

- Variational Autoencoders (VAE)

**T2: Sample generation**

\[ Z \rightarrow X \]

- Generative Adversarial Networks (GAN)

Deep Bayesian Learning

Introduction

Bayesian Deep Learning

Non-identifiable Issues

ALICE

A Unified View

Results

Chunyuan Li
### Adversarially Learned Inference (ALI)

**ALI:** Discriminator takes in pair-wise samples $(\mathbf{x}, \tilde{\mathbf{z}})$ and $(\mathbf{x}, \mathbf{z})$

**Joint distribution matching:** $p(\mathbf{x}, \mathbf{z}) = q(\mathbf{x}, \mathbf{z})$

**GAN:** Discriminator takes in samples: $\mathbf{x}$ and $\tilde{\mathbf{x}}$

**Marginal distribution matching:** $p(\mathbf{x}) = q(\mathbf{x})$

**Important details:** Universal distribution approximators for the sampling procedure of conditionals $\tilde{\mathbf{x}} \sim p_\theta(\mathbf{x}|\mathbf{z})$ and $\tilde{\mathbf{z}} \sim q_\phi(\mathbf{z}|\mathbf{x})$ are carried out as:

- $\tilde{\mathbf{x}} = g_\theta(\mathbf{z}, \epsilon)$, $\mathbf{z} \sim p(\mathbf{z})$, $\epsilon \sim \mathcal{N}(0, \mathbf{I})$, and
- $\tilde{\mathbf{z}} = g_\phi(\mathbf{x}, \zeta)$, $\mathbf{x} \sim q(\mathbf{x})$, $\zeta \sim \mathcal{N}(0, \mathbf{I})$, 

Chunyuan Li
Non-identifiable Issues

- Joint distribution matching as shape matching of two probability measures

- The matched joint distribution can still have arbitrary shape

Issues: The correlation between $x$ and $z$ is not specified.
Problem Illustration

- In (a), for $0 < \delta < 1$, we can generate “realistic” $x$ from any sample of $p(z)$, but with poor reconstruction.

Any $\delta \in [0, 1]$ is a valid solution of ALI?!

Many applications require meaningful mappings.

1. In unsupervised learning, the inferred latent code can reconstruct its $x$ itself with high probability. $\delta \rightarrow 1$ or $\delta \rightarrow 0$
In (b) \( \delta = 1 \) or (c) \( \delta = 0 \), only one of the solutions will be meaningful in supervised learning.

Many applications require meaningful mappings.

In supervised learning, the task-specified correspondence between samples imposes restrictions on the mappings.
Adversarially Learned Inference with Conditional Entropy (ALICE)

$$\min_{\theta, \phi} \max_{\omega} \mathcal{L}_{\text{ALICE}}(\theta, \phi, \omega) = \mathcal{L}_{\text{ALI}}(\theta, \phi, \omega) + \mathcal{L}_{\text{CE}}(\theta, \phi).$$  \hfill (1)

- **CE enforces correlation between random variables**

- **CE is intractable in practice**

- Four algorithms are proposed to bound or approximate CE

Code:
https://github.com/ChunyuanLI/ALICE
Unsupervised Learning

In unsupervised learning, cycle-consistency is considered to upperbound CE:

1. **Explicit cycle-consistency**  Prescribed the distribution forms, e.g. $\ell_k$-norm
2. **Implicit cycle-consistency**  Adversarially learned “perfect” reconstruction

\[
\min_{\theta, \phi} \max_{\eta} \mathcal{L}_{\text{Cycle}}^A(\theta, \phi, \eta) = \mathbb{E}_{x \sim q(x)}[\log \sigma(f_\eta(x, x))] \\
+ \mathbb{E}_{\hat{x} \sim p_\theta(x|z), z \sim q_\phi(z|x)} \log(1 - \sigma(f_\eta(x, \hat{x}))).
\]

\[
H_{q_\phi}(x | z) + \mathbb{E}_{q_\phi(z)}[\text{KL}(q_\phi(x | z) || p_\theta(x | z))] = -\mathbb{E}_{q_\phi(x, z)}[\log p_\theta(x | z)]
\]

- **Comments**
  - Explicit method is easy to train, but could generate “blurred” samples
  - Implicit method is difficult to train, but potentially more “realistic” samples
In semi-supervised learning, the pairwise information is leveraged to approximate CE:

1. **Explicit mapping** Prescribed the forms, $\ell_k$-norm or standard supervised losses
2. **Implicit mapping** Implicit mapping via conditional GAN

$$\min_{\theta} \max_{\mathcal{X}} \mathcal{L}^A_{\text{Map}}(\theta, \mathcal{X}) = \mathbb{E}_{x, z \sim \bar{p}(x, z)}[\log \sigma(f_\mathcal{X}(x, z))] + \mathbb{E}_{\hat{x} \sim p_\theta(\hat{x} | z)}[\log(1 - \sigma(f_\mathcal{X}(\hat{x}, z)))]$$  \hfill (3)

**Diagram:**
- **Paired data**
- **Unpaired data**

ALICE with cross-domain mappings
ALICE with cycle-consistency (when necessary)
A Unified Perspective

Joint distribution matching

Conditional GAN

One-step

Paired samples

Unpaired samples

Setup

Method

ALI/BiGAN

CycleGAN/DiscoGAN/DualGAN

Two-step

Chunyuan Li
**A Unified Perspective**

**Joint distribution matching**

- **ALI is equivalent to CycleGAN** (Two GAN Losses + Two Cycle Losses)

\[
H[q_\phi(x|z)] + \mathbb{E}_{q_\phi(z)}[\text{KL}(q_\phi(x|z)\|p_\theta(x|z))] = -\mathbb{E}_{q_\phi(x,z)}[\log p_\theta(x|z)]
\]

**Proof:**

\[
\text{CycleGAN} \begin{cases} 
\text{GAN Loss} & \Rightarrow \text{Marginal Matching} \\
\text{Cycle Loss} & \Rightarrow \text{Conditional Matching} 
\end{cases} \Rightarrow \text{Joint Matching}
\]
A Unified Perspective

Joint distribution matching

- **Conditional GAN is doing joint distribution matching**

When the optimum in (3) is achieved, \( \tilde{\pi}(x, z) = p_{\theta^*}(x, z) = q_{\phi^*}(x, z) \).

One can leverage the empirically-defined distributions \( \tilde{\pi}(x, z) \) implied by paired data, to resolve the ambiguity issues in unsupervised bivariate GANs.

**Proof:**

\[
\min_{\theta} \max_{\chi} \mathcal{L}^A_{\text{Map}}(\theta, \chi) = \mathbb{E}_{x, z \sim \tilde{\pi}(x, z)} [\log \sigma(f_{\chi}(x, z)) + \mathbb{E}_{\hat{x} \sim p_\theta(\hat{x} | z)} \log(1 - \sigma(f_{\chi}(\hat{x}, z)))] \rightarrow \tilde{\pi}(x, z) = p_{\theta^*}(x, z)
\]
All these bivariate GAN models are learning to match the joint distributions: either using different methods, or in different problem setups.
Results: Unsupervised Learning

Grid search over a set of hyper-parameters for 576 experiments

Figure: Generation (c) and reconstruction (d) results on toy data (a,b).
Results: Unsupervised Learning

Sensitivity to hyper-parameters $\lambda$

$\mathcal{L}_{\text{ALICE}} = \mathcal{L}_{\text{ALI}} + \lambda \mathcal{L}_{\text{CE}}$

Toy dataset

MNIST

CIFAR-10

Generation $\uparrow$

Reconstruction $\downarrow$

(a) Toy dataset: image generation

(b) Toy dataset: image reconstruction

(c) MNIST: image generation

(d) MNIST: image reconstruction

(e) CIFAR: image generation

(f) CIFAR: image reconstruction
Results: Semi-supervised Learning

ALICE for painting the cartoon “Alice’s Wonderland”, based on edges

Edge domain

Cartoon domain

Training set: two domains (edges and cartoon) and two modes (Alice and Rabbit)

ALICE: one pair in each mode is leveraged to resolve ambiguity

CycleGAN: mixing colors due to the non-identifiable issue

Code:
https://github.com/ChunyuanLi/Alice4Alice
Measure the **Intrinsic Dimension of Objective Landscapes**

C Li, H Farkhoor, R Liu, J Yosinski. **ICLR** 2018
Measure the Intrinsic Dimension of Objective Landscapes
Deep/Bayesian learning achieves better representations via increasing model complexity.

How many parameters are really needed?
Objective: \( U(\theta, D) \triangleq - \sum_{i=1}^{N} \log p(d_i | \theta) - \log p(\theta) \)

- **Datasets**: \( D = \{d_i\}_{i=1}^{N} \) where \( d_i \triangleq (x_i, y_i) \)
- **Neural Nets**: \( \theta \in \mathbb{R}^D \)

One Example of **Objective Landscapes**
**Direct vs. Subspace Training**

**Direct Training:** Training in the Original Weight Space

\[ x \rightarrow \text{Neural Nets} \rightarrow y \]

**Subspace Training:** Training in a Low Dimensional Orthogonal Space

\[ \theta^{(D)} = \theta_0^{(D)} + P\theta^{(d)} \]

- **Projection matrix**
- **Subspace parameters**

Initial \( \theta_0^{(D)} \) \n
Final \( \theta^{(D)} \)
As we increase $d$, we generally observe a sharp increase in network performance. This $d$ is the **Intrinsic Dimension**!

1d random line search; Hard to find a good solution

The entire space is spanned; Any available solutions can be discovered
A simple objective with **1000 parameters** to optimize:

\[ J(\theta) = \sum_{r=1}^{10} \| 1 - \sum_{i=10r-99}^{100r} \theta_i \|^2 \]

- **10 constraint**: Provide the pair-wise fitting constraints
- **Every 100 weights sum to one**: Provide the functional constraints

**A Toy Problem**

Generalize the concept to **Neural Nets**:
- **Datasets**: Provide the pair-wise fitting constraints
- **Neural Nets Architectures**: Provide the functional constraints
Some networks are very compressible

- 2-layer Fully Connected Networks (FC) on MNIST

```
D = 199, 210
```

\[ d_{\text{int}} = 750 \]

Redundancy of the solution:
\[ S \geq D - d_{\text{int}} \]

- 750 is less than the number of input pixels (784)
  (A lot of image pixels are always black)

- High compression rate: 0.4%. Storage only requires 750 parameters + 1 seed

- Highly redundant solution: 
  \[ S > 199,210 - 750 = 198,460 \]
MNIST with 20 different FC’s

Depth = \{1, 2, 3, 4, 5\}

Width = \{50, 100, 200, 400\}

- A **stable** metric across a family of models
- Every extra parameter added to the native space just goes directly toward increasing the redundancy of the solution set
Objective: $U(\theta, D)$

- **Fitness of Network Architectures**
  Fixing the dataset, $d_{\text{int}}$ indicates the fitness of network architectures to the tasks.

  **Case Study:**
  To achieve >50% validation accuracy on CIFAR, FC, LeNet and ResNet approximately requires $d_{\text{int}}$ as 9K, 2.9K and 1K, respectively.

- **Difficulty of Tasks/Datasets**
  Fixing the architecture, $d_{\text{int}}$ indicates the difficulty level of specific tasks

  **Case Study:**
  The intrinsic dimension of 2-layer FC for MNIST and CIFAR is 750 and 9K, respectively.

  Shuffled-label MNIST: 190K; ImageNet: >800K
Evolutionary Strategies (ES)

$d_{\text{int}} = 6,000 \sim \text{CIFAR}$  
$d_{\text{int}} = 700 \sim \text{MNIST}$  
$d_{\text{int}} = 4$

Atari Pong  
Humanoid  
Inverted Pendulum

The low $d_{\text{int}}$ suggests why random search and gradient-free methods work
Recent Development

- **Applications to Guiding Model Selection**

- **More Resource**
  - **UBER** Blog: [https://eng.uber.com/intrinsic-dimension/](https://eng.uber.com/intrinsic-dimension/)
  - **Code**: [https://github.com/uber-research/intrinsic-dimension](https://github.com/uber-research/intrinsic-dimension)
  - **YouTube** Video
Summary: Learning Trajectory

- Machine Learning
  - (Scalable) Bayesian Learning
    - Computer Graphics
    - Computer Vision
  - Generative Models
    - SG-MCMC
    - SG-VI
    - S-EP
  - Reinforcement Learning
  - Deep Learning
    - NLP

Chunyuan Li
Thanks!

Chunyuan Li

Duke University

Email: chunyuan.li@hotmail.com
Web: http://chunyuan.li/