Towards Better Representations with Deep/Bayes Learning

Chunyuan Li

Duke University

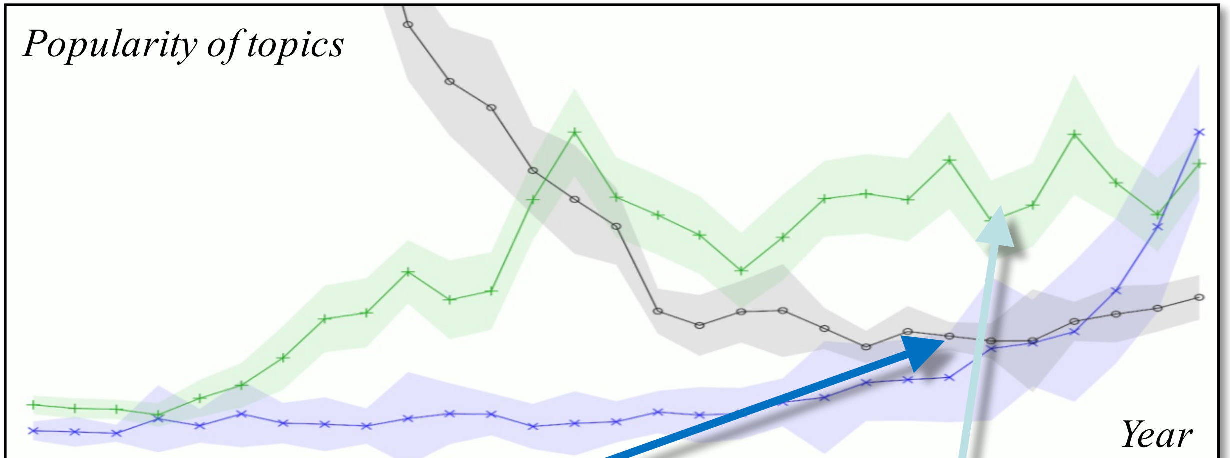
http://chunyuan.li

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**

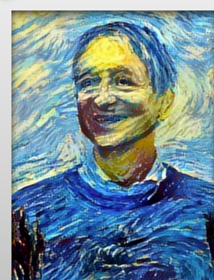
Popular research topics

Deep (Neural Nets) & Bayesian Learning



1987 Dynamic topic modeling on NIPS papers 2015

Deep
CNNs
RNNs
dropout
batch norm.
...



Geoffrey Hinton

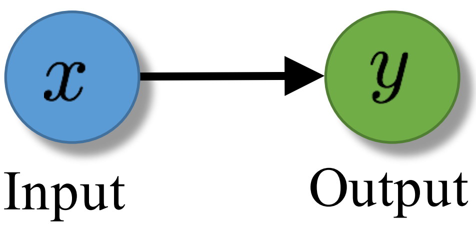
Bayes
prior
posterior
generative
MCMC
...

Thomas Bayes

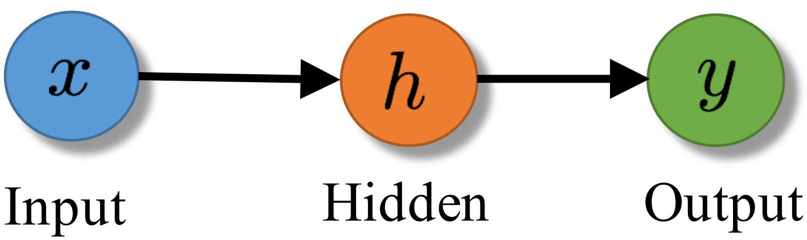
Figures adapted from Teh's talk at NIPS 2017

Increasing flexibility for representations

Deep Learning

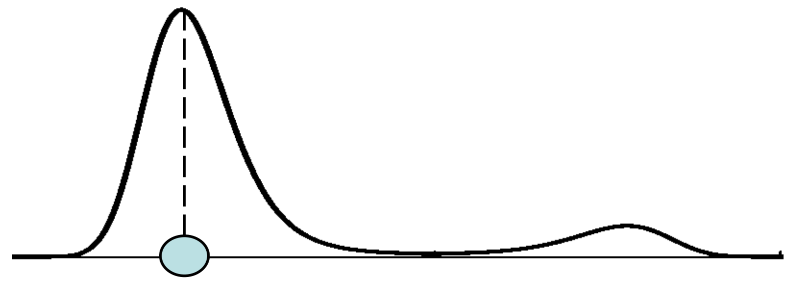


Shallow models (e.g., logistic regression)

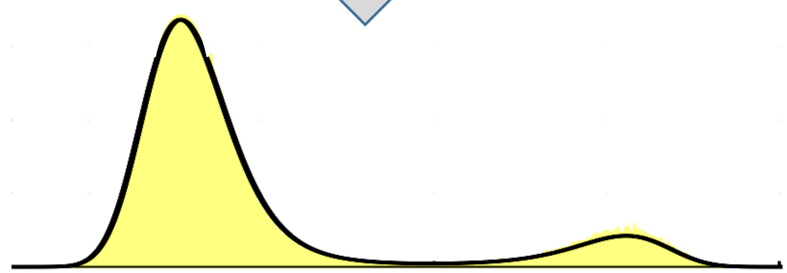


Deep models (e.g., multi-layer perceptron)

Bayesian Learning

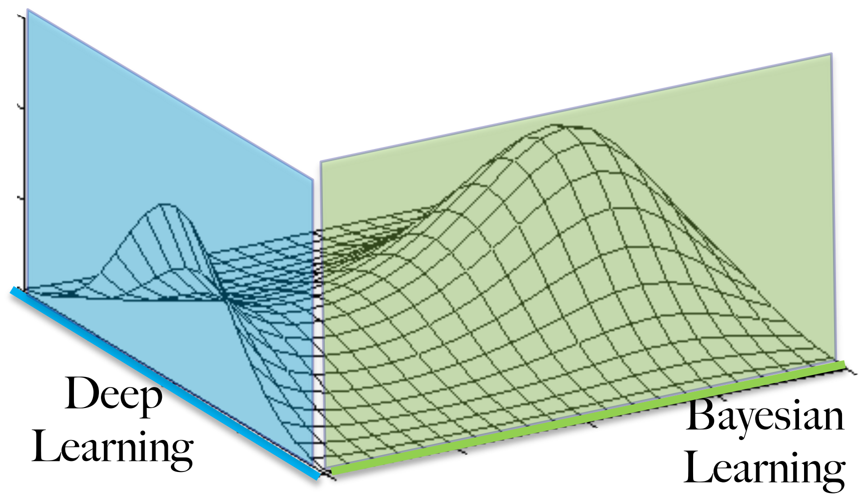


Point estimate (e.g., SGD)



Full distribution (e.g., MCMC)

Towards **Better** Representations



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.

Increasing Flexibility ⟸ Increasing Complexity

Seek further understanding?

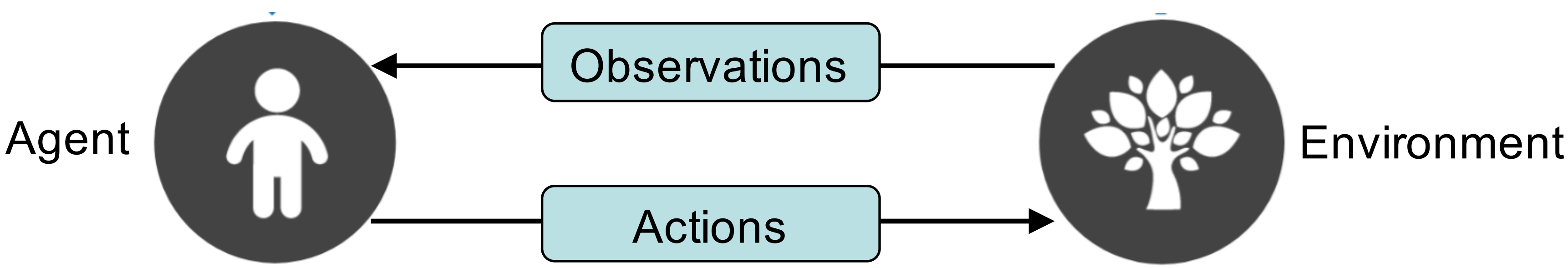
- *Intrinsic Dimension of Objective Landscape*

Bayesian Deep Learning

Problem Setup

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

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



- For testing input, Bayesian predictive distribution

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

Figures adapted from Teh's talk at NIPS 2017

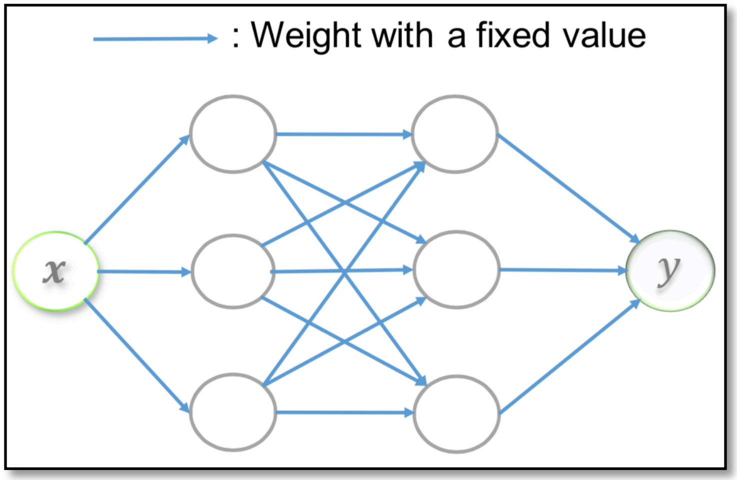
The Pitfall of Stochastic Optimization

- In optimization, the single ``best" point on training is used

$$\boldsymbol{\theta}_{MAP} = \text{argmax}_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}|\mathcal{D}) = \text{argmax}_{\boldsymbol{\theta}} U(\boldsymbol{\theta})$$

$$U(\boldsymbol{\theta}) \triangleq \underbrace{-\sum_{i=1}^{N} \log p(\boldsymbol{d}_i|\boldsymbol{\theta})}_{\text{loss function}} \underbrace{- \log p(\boldsymbol{\theta})}_{\text{regularizer}}$$

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



- The MAP approximates this expectation as

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

Parameter uncertainty is ignored

Large-scale Bayesian Learning

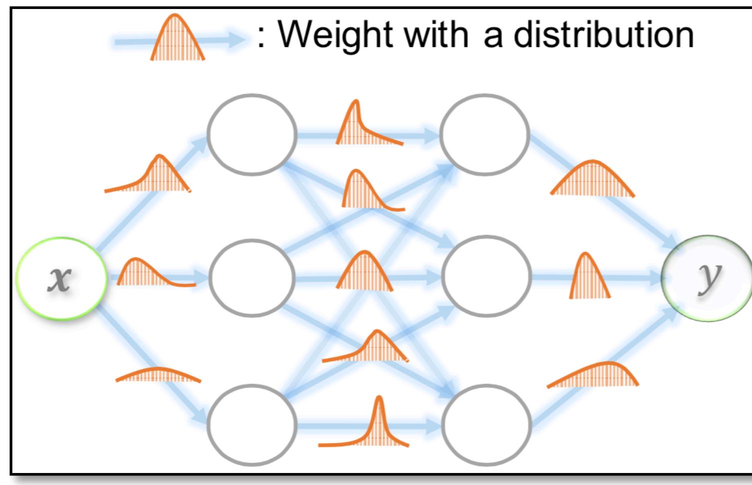
- In Bayesian, the full posterior distribution after observing training set is used

SG-MCMC
SG-VI
S-EP

?

Requirements:

- Accurate approximation
- Scalability to large datasets



- Samples are used for prediction

$$p(y|x,\mathcal{D}) = \mathbb{E}_{p(\boldsymbol{\theta}|\mathcal{D})}[p(y|x,\boldsymbol{\theta})] \approx \frac{1}{T}\sum_{t=1}^{T} p(y|x,\boldsymbol{\theta}_t)$$

SGLD *vs.* SGD

- Stochastic Gradient Langevin Dynamics (SGLD) draws samples:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \epsilon_t \tilde{\boldsymbol{f}}_t + \boxed{\sqrt{2\epsilon_t}\boldsymbol{\xi}_t}.$$

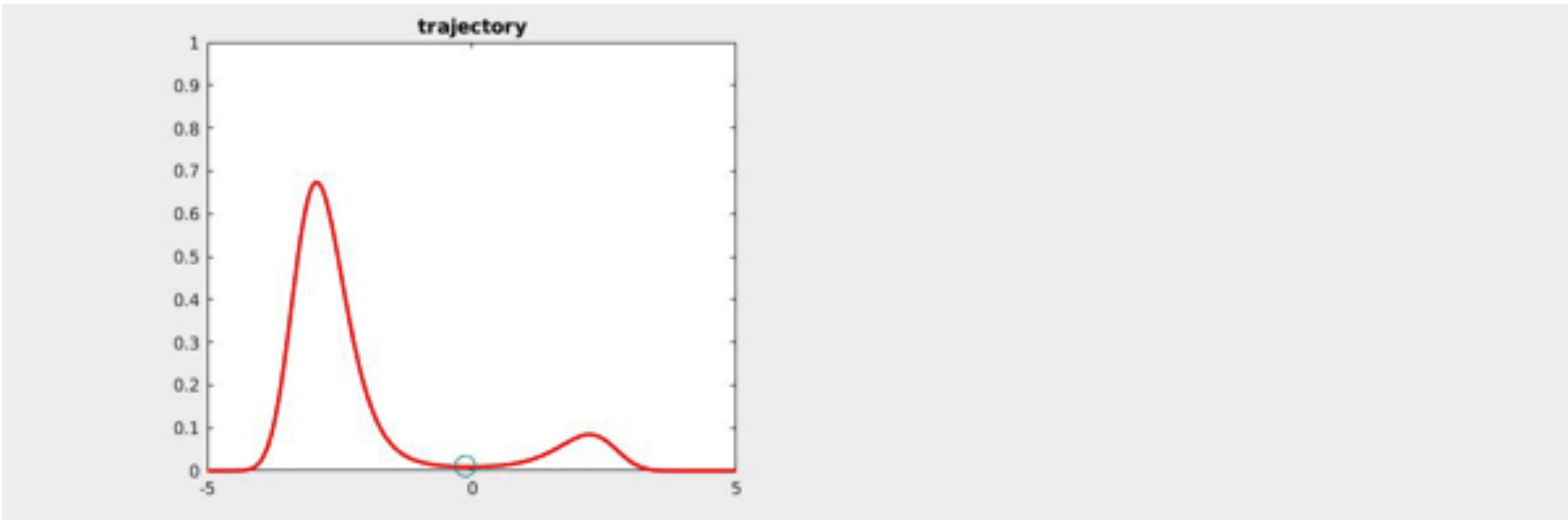
where

  - Step size: ϵ_t
  - Stochastic gradient: $\tilde{\boldsymbol{f}}_t = \nabla \tilde{U}_t(\boldsymbol{\theta})$
  - Gaussian noise: $\boldsymbol{\xi}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

- SGLD is the SG-MCMC analog to SGD

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \epsilon_t \tilde{\boldsymbol{f}}_t$$

Sampling Procedure of SGLD



Sampling Dynamics

Approximated Histogram

Stochastic Gradient MCMC vs Optimization

Algorithms	**SG-MCMC**	**Optimization**
Basic	SGLD	SGD
Precondition	pSGLD	Adam/RMSprop/Adagrad
Momentum	SGHMC	SGD with momentum
Thermostat	SGNHT	Santa

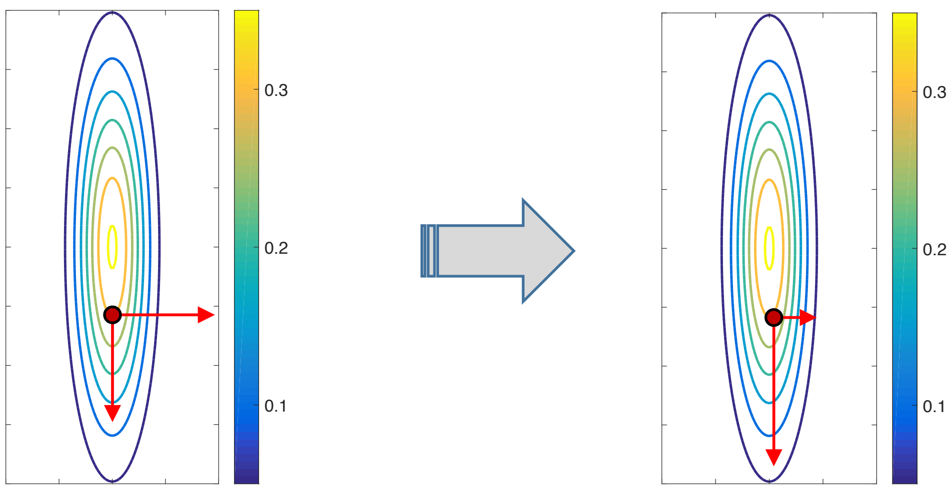
C Li, C Chen, D Carlson, L Carin. **AAAI** 2016. Oral Presentation
Preconditioned Stochastic Gradient Langevin Dynamics for Deep Neural Networks

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

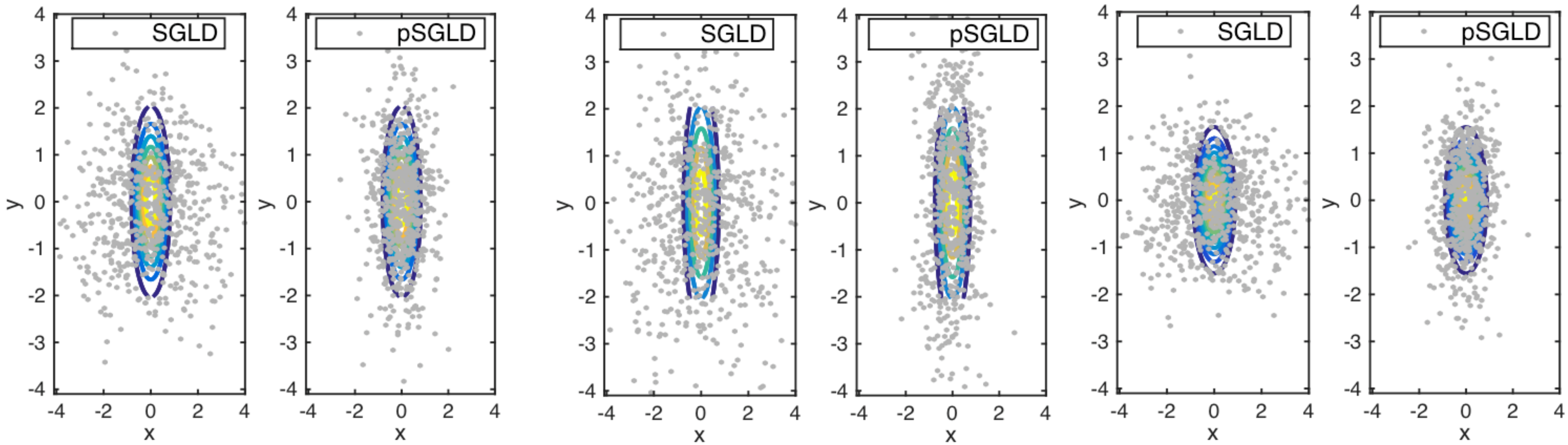
- Preconditioner: approximate geometry information.
- Preconditioner constructed as diagonal matrix.
- Adjust the step size, according the local geometry.



Any preconditioning optimization algorithms (eg, 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 $a = 0.5, \epsilon = 0.3$

Applications to Deep Neural Nets

- **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, Binary dropout $\approx$ Gaussian dropout

By combining dropConnect and Gaussian corruption, the update rule:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\xi}_0 \odot \boldsymbol{\theta}_t - \frac{\epsilon}{2}\tilde{\boldsymbol{f}}_t = \boldsymbol{\theta}_t - \frac{\epsilon}{2}\tilde{\boldsymbol{f}}_t + \boldsymbol{\xi}_0' \ ,$$

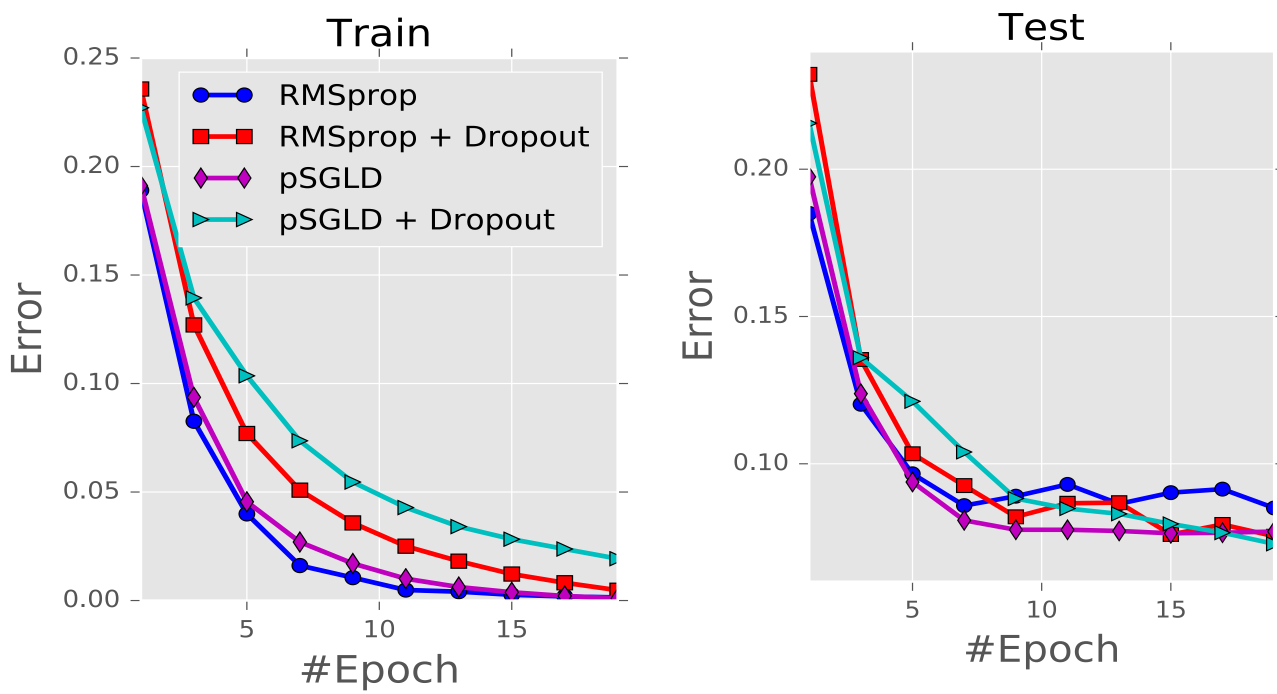
where $\boldsymbol{\xi}_0' \sim \mathcal{N}\left(0, \frac{p}{(1-p)}\mathrm{diag}(\boldsymbol{\theta}_t^2)\right)$

- 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

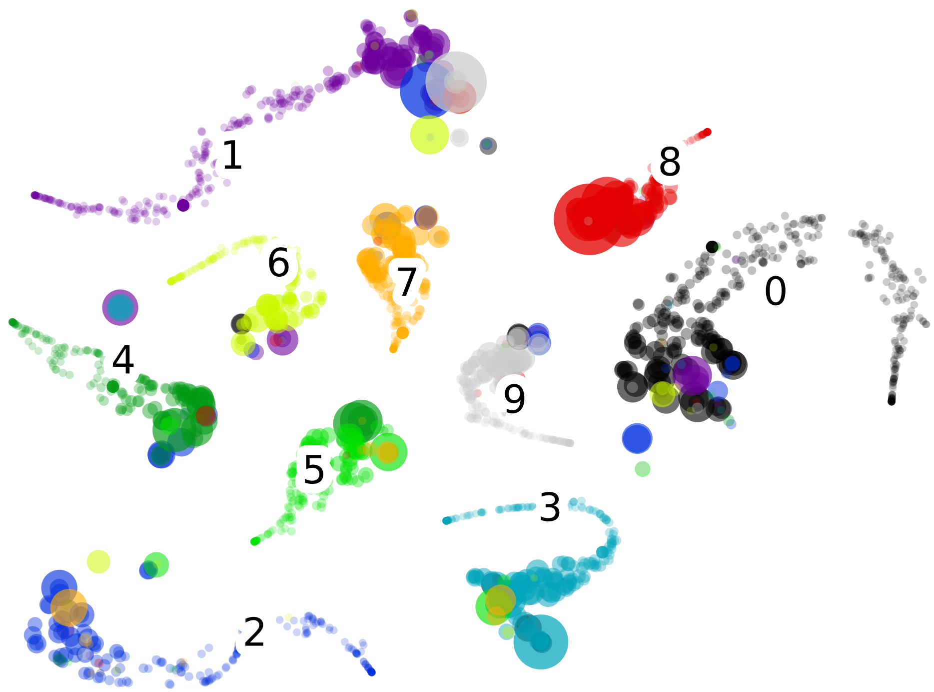
- *Optimization converges faster on training, but overfit*
- *Uncertainty learned in training prevent over-fitting on testing*



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*



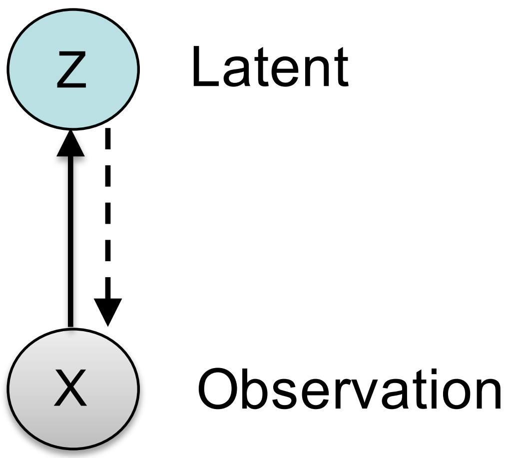
t-SNE embedding of prediction mean and std

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

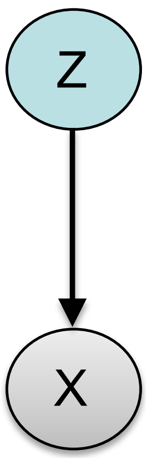
Deep Generative Models

T1: Latent Variable Inference



Variational Autoencoders (VAE)

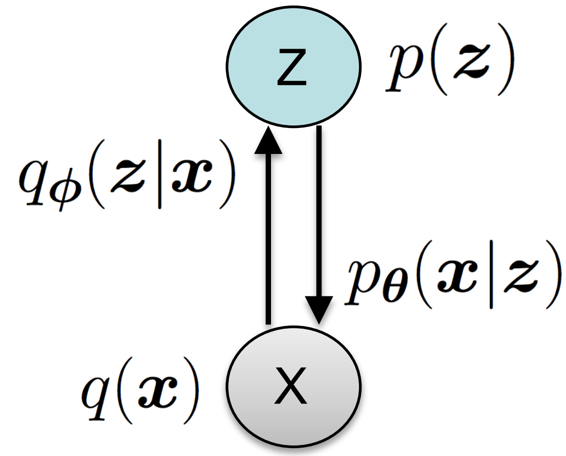
T2: Sample generation



Generative Adversarial Networks (GAN)

Adversarial Learning for Distribution Matching

Adversarially Learned Inference (ALI)



$p_\theta(\boldsymbol{x}) = \int p_\theta(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})\mathrm{d}\boldsymbol{z}$

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

$$\boxed{\text{Joint distribution matching: } p(\boldsymbol{x}, \boldsymbol{z}) = q(\boldsymbol{x}, \boldsymbol{z})}$$

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

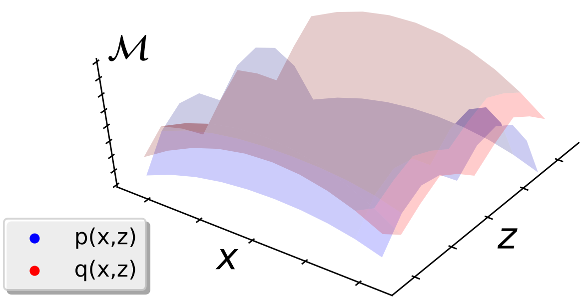
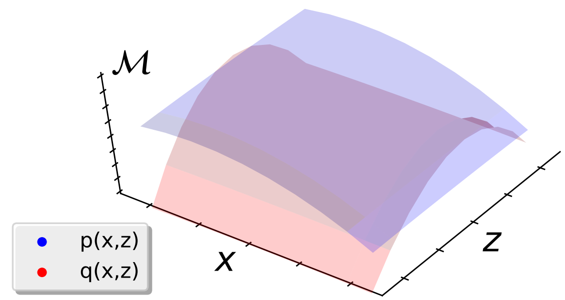
$$\boxed{\text{Marginal distribution matching: } p(\boldsymbol{x}) = q(\boldsymbol{x})}$$

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

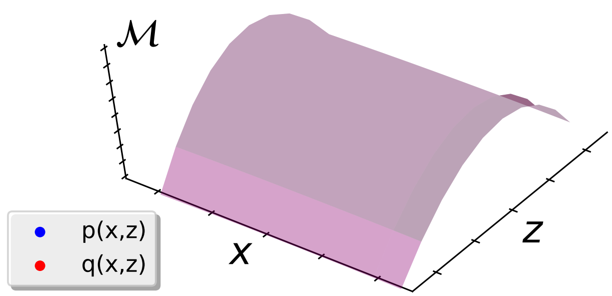
$$\tilde{\boldsymbol{x}} = g_\theta(\boldsymbol{z}, \boldsymbol{\epsilon}), \ \boldsymbol{z} \sim p(\boldsymbol{z}), \ \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}), \ \text{and}$$
$$\tilde{\boldsymbol{z}} = g_\phi(\boldsymbol{x}, \boldsymbol{\zeta}), \ \boldsymbol{x} \sim q(\boldsymbol{x}),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

Non-identifiable Issues

- Joint distribution matching as shape matching of two probability measures

. . .  . . .  . . .

- The matched joint distribution can still have arbitrary shape

. . .  . . .

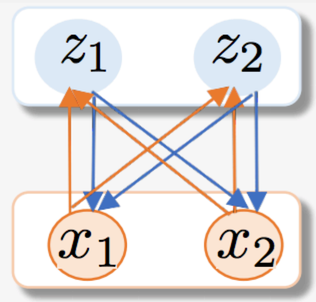
Issues: The correlation between x and z is not specified.

Non-identifiable Issues

Problem Illustration

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

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



	z_1	z_2
x_1	$\delta/2$	$(1-\delta)/2$
x_2	$(1-\delta)/2$	$\delta/2$

(a)

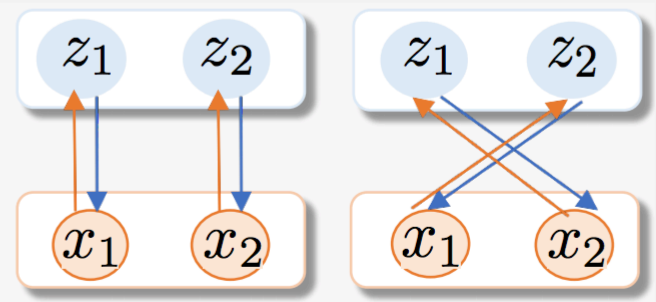
Many applications require meaningful mappings.

1. In unsupervised learning, the inferred latent code can reconstruct its $\boldsymbol{x}$ itself with high probability. $\delta \to 1$ or $\delta \to 0$

Non-identifiable Issues

Problem Illustration

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



	z_1	z_2
x_1	1/2	0
x_2	0	1/2

(b)

	z_1	z_2
x_1	0	1/2
x_2	1/2	0

(c)

- In (b) $\delta = 1$ or (c) $\delta = 0$, only one o the solutions will be meaningful in supervised learning.

Many applications require meaningful mappings.

❷ In supervised learning, the task-specified correspondence between samples imposes restrictions on the mappings.

ALICE

Adversarially Learned Inference with Conditional Entropy (ALICE)

$$\min_{\boldsymbol{\theta},\boldsymbol{\phi}} \max_{\boldsymbol{\omega}} \underbrace{\mathcal{L}_{\text{ALICE}}(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\omega})}_{\text{Our ALICE Objective}} = \underbrace{\mathcal{L}_{\text{ALI}}(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\omega})}_{\text{ALI Objective}} + \underbrace{\mathcal{L}_{\text{CE}}(\boldsymbol{\theta}, \boldsymbol{\phi})}_{\text{CE Regularizer}} . \tag{1}$$

- *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.* ℓ_k-norm
2. **Implicit cycle-consistency** Adversarially learned "perfect" reconstruction

$$\min_{\boldsymbol{\theta},\boldsymbol{\phi}} \max_{\boldsymbol{\eta}} \ \mathcal{L}^{\mathrm{A}}_{\mathrm{Cycle}}(\boldsymbol{\theta},\boldsymbol{\phi},\boldsymbol{\eta}) = \mathbb{E}_{\boldsymbol{x}\sim q(\boldsymbol{x})}[\log \sigma(f_{\boldsymbol{\eta}}(\boldsymbol{x},\boldsymbol{x}))] + \mathbb{E}_{\hat{\boldsymbol{x}}\sim p_{\boldsymbol{\theta}}(\hat{\boldsymbol{x}}|\boldsymbol{z}),\boldsymbol{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \log(1-\sigma(f_{\boldsymbol{\eta}}(\boldsymbol{x},\hat{\boldsymbol{x}})))]. \tag{2}$$

$$\underbrace{H^{q_\phi}(\boldsymbol{x}|\boldsymbol{z})}_{\text{Conditional entropy}} + \underbrace{\mathbb{E}_{q_\phi(\boldsymbol{z})}[\mathrm{KL}(q_\phi(\boldsymbol{x}|\boldsymbol{z})\|p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}))]}_{\text{Conditional distribution matching}} = \underbrace{-\mathbb{E}_{q_\phi(\boldsymbol{x},\boldsymbol{z})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})]}_{\text{Cycle consistency}}$$

Comments

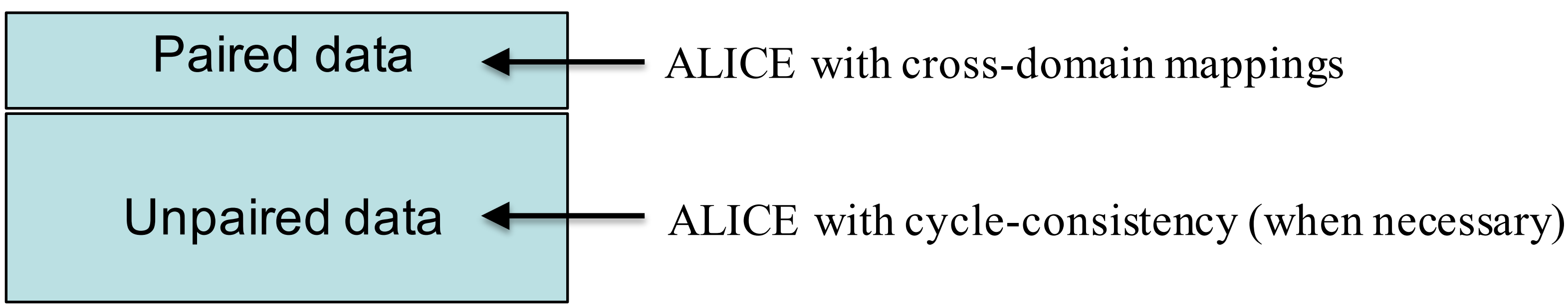
- *Explicit method is easy to train, but could generate "blurred" samples*
- *Implicit method is difficult to train, but potentially more "realistic" samples*

Semi-supervised Learning

In semi-supervised learning, the pairwise information is leveraged to approximate CE:

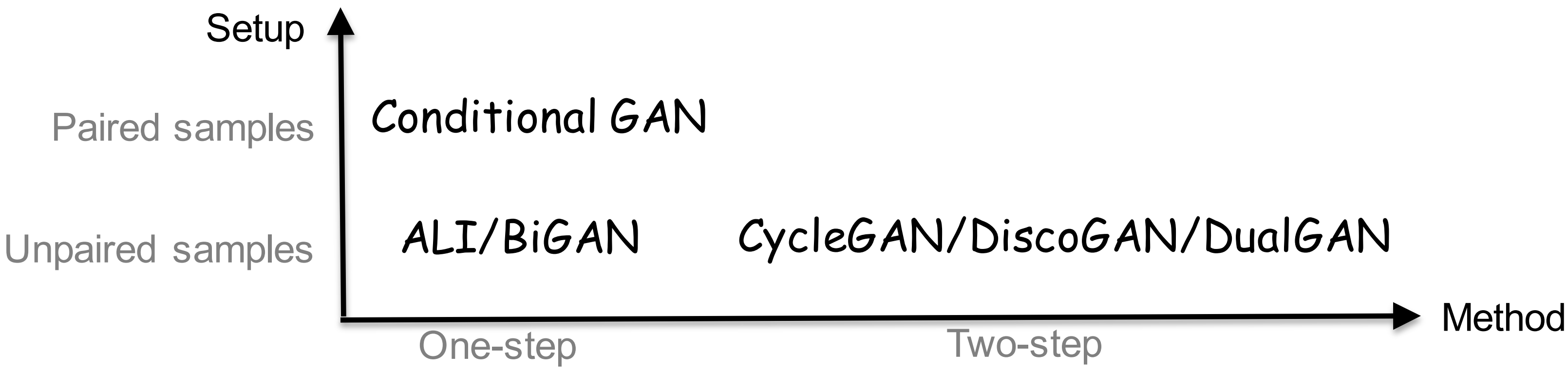
1. **Explicit mapping** Prescribed the forms, ℓ_k-norm or standard supervised losses
2. **Implicit mapping** Implicit mapping via conditional GAN

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\chi}} \; \mathcal{L}^{\mathrm{A}}_{\mathrm{Map}}(\boldsymbol{\theta}, \boldsymbol{\chi}) = \mathbb{E}_{\boldsymbol{x},\boldsymbol{z}\sim\tilde{\pi}(\boldsymbol{x},\boldsymbol{z})}[\log \sigma(f_{\boldsymbol{\chi}}(\boldsymbol{x}, \boldsymbol{z})) \\ + \mathbb{E}_{\hat{\boldsymbol{x}}\sim p_{\boldsymbol{\theta}}(\hat{\boldsymbol{x}}|\boldsymbol{z})} \log(1 - \sigma(f_{\boldsymbol{\chi}}(\hat{\boldsymbol{x}}, \boldsymbol{z})))]. \tag{3}$$



A Unified Perspective

Joint distribution matching



A Unified Perspective

Joint distribution matching

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

CycleGAN is easier to train, as it decomposes the joint distribution matching objective (as in ALI) into four subproblems.

$$\underbrace{H^{q_\phi}(\boldsymbol{x}|\boldsymbol{z})}_{\text{Conditional entropy}} + \underbrace{\mathbb{E}_{q_\phi(\boldsymbol{z})}[\mathrm{KL}(q_\phi(\boldsymbol{x}|\boldsymbol{z})\|p_\theta(\boldsymbol{x}|\boldsymbol{z}))]}_{\text{Conditional distribution matching}} = \underbrace{-\mathbb{E}_{q_\phi(\boldsymbol{x},\boldsymbol{z})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})]}_{\text{Cycle consistency}}$$

Proof:

CycleGAN $\left\{\begin{array}{l}\text{GAN Loss} \Longrightarrow \text{Marginal Matching} \\ \text{Cycle Loss} \Longrightarrow \text{Conditional Matching}\end{array}\right\} \Longrightarrow$ Joint Matching ■

A Unified Perspective

Joint distribution matching

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

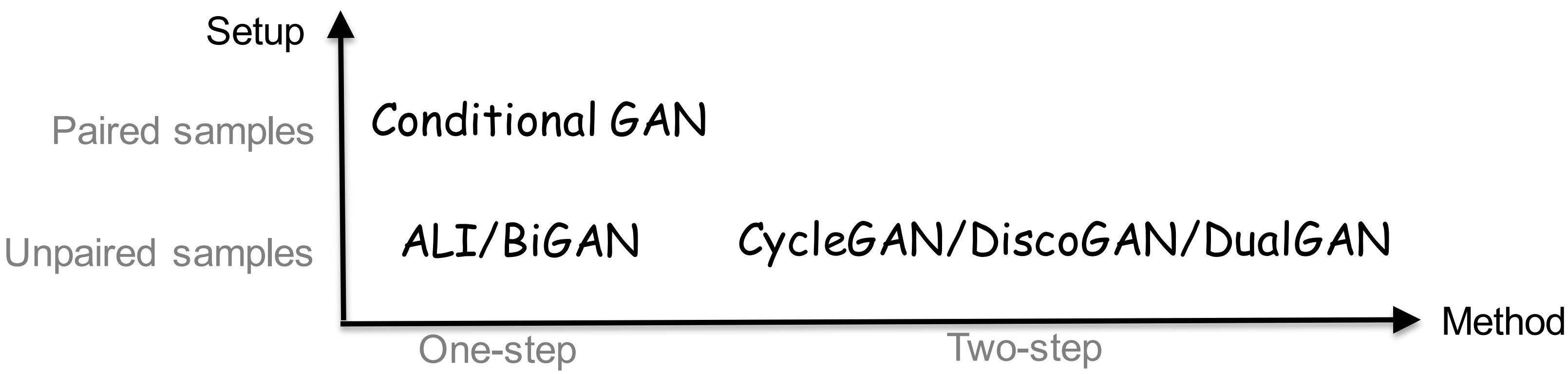
When the optimum in (3) is achieved, $\tilde{\pi}(\boldsymbol{x}, \boldsymbol{z}) = p_{\boldsymbol{\theta}^*}(\boldsymbol{x}, \boldsymbol{z}) = q_{\boldsymbol{\phi}^*}(\boldsymbol{x}, \boldsymbol{z})$.
One can leverage the empirically-defined distributions $\tilde{\pi}(\boldsymbol{x}, \boldsymbol{z})$ implied by paired data, to resolve the ambiguity issues in unsupervised bivariate GANs.

Proof:

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\chi}} \; \mathcal{L}^{\mathrm{A}}_{\mathrm{Map}}(\boldsymbol{\theta}, \boldsymbol{\chi}) = \mathbb{E}_{\boldsymbol{x}, \boldsymbol{z} \sim \tilde{\pi}(\boldsymbol{x}, \boldsymbol{z})} [\log \sigma(f_{\boldsymbol{\chi}}(\boldsymbol{x}, \boldsymbol{z})) + \mathbb{E}_{\hat{\boldsymbol{x}} \sim p_{\boldsymbol{\theta}}(\hat{\boldsymbol{x}}|\boldsymbol{z})} \log(1 - \sigma(f_{\boldsymbol{\chi}}(\hat{\boldsymbol{x}}, \boldsymbol{z})))] \Longrightarrow \tilde{\pi}(\boldsymbol{x}, \boldsymbol{z}) = p_{\boldsymbol{\theta}^*}(\boldsymbol{x}, \boldsymbol{z}) \quad \blacksquare$$

A Unified Perspective

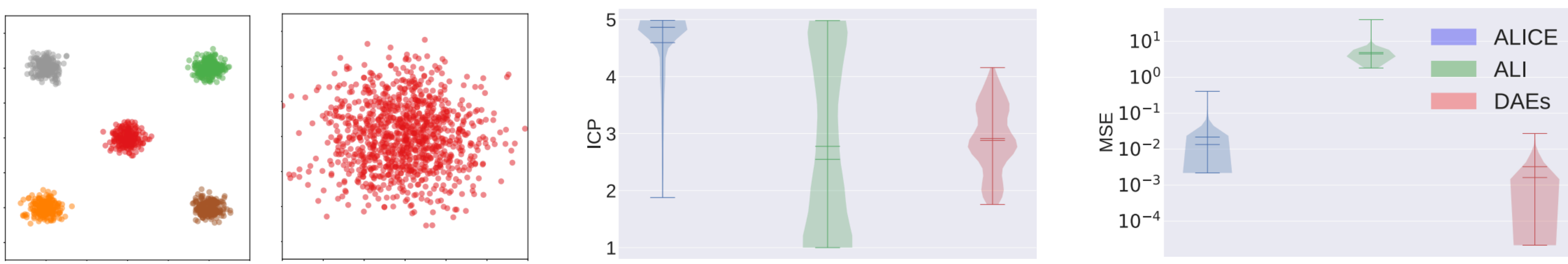
Joint distribution matching



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



(a) True $\boldsymbol{x}$ (b) True $\boldsymbol{z}$ (c) Inception Score (d) MSE

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

Results: Unsupervised Learning

Sensitivity to hyper-parameters λ

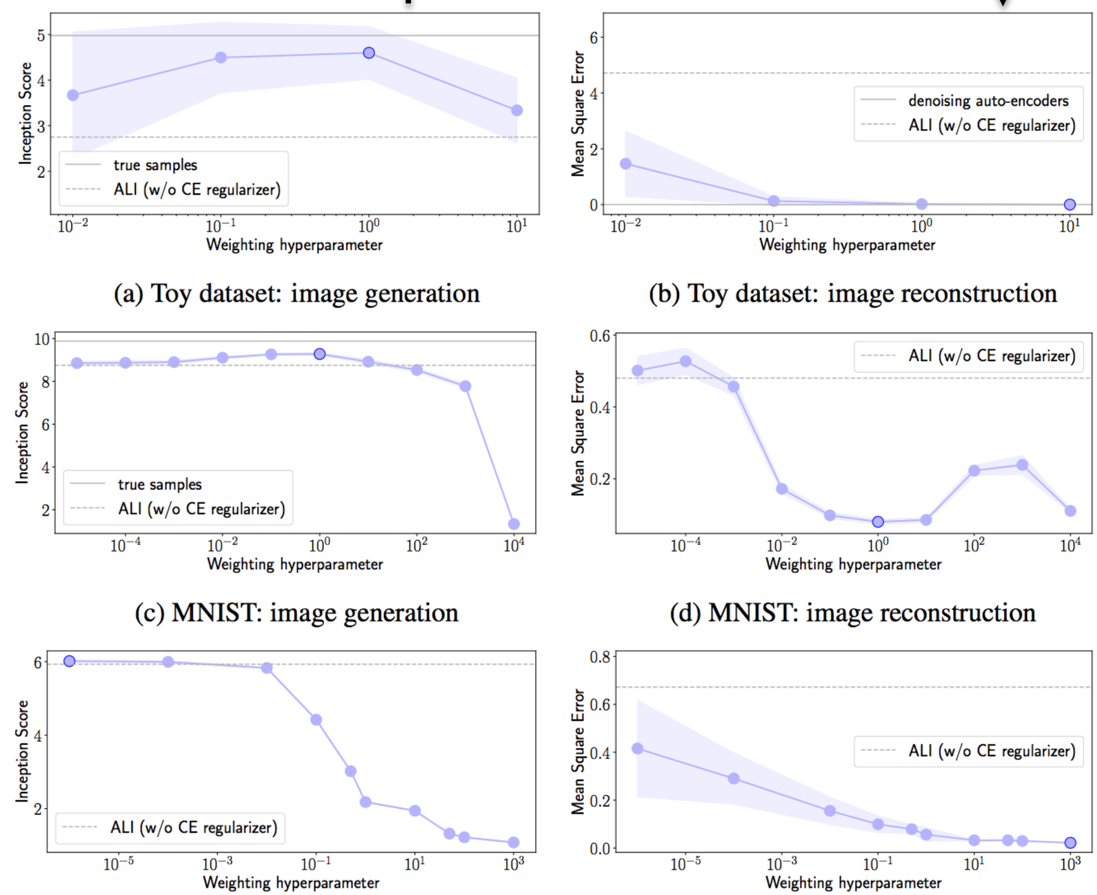
$$\mathcal{L}_{\text{ALICE}} = \mathcal{L}_{\text{ALI}} + \lambda\, \mathcal{L}_{\text{CE}}$$

Generation ↑ Reconstruction ↓

Toy dataset

MNIST

CIFAR-10



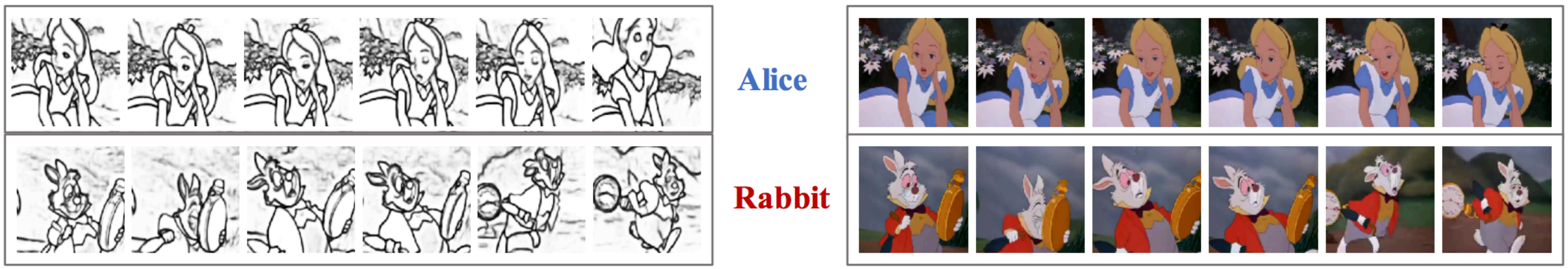
(a) Toy dataset: image generation (b) Toy dataset: image reconstruction

(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

Motivation

Deep/Bayesian learning achieves better representations via increasing model complexity

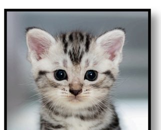
Goodness of fit

Dataset
(x, y)

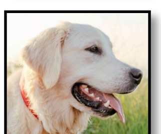
Choose architectures

Neural Nets

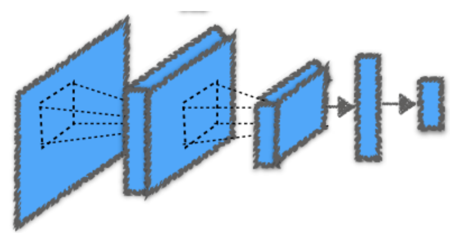
Increase model parameters for better fitting



Cat



Dog



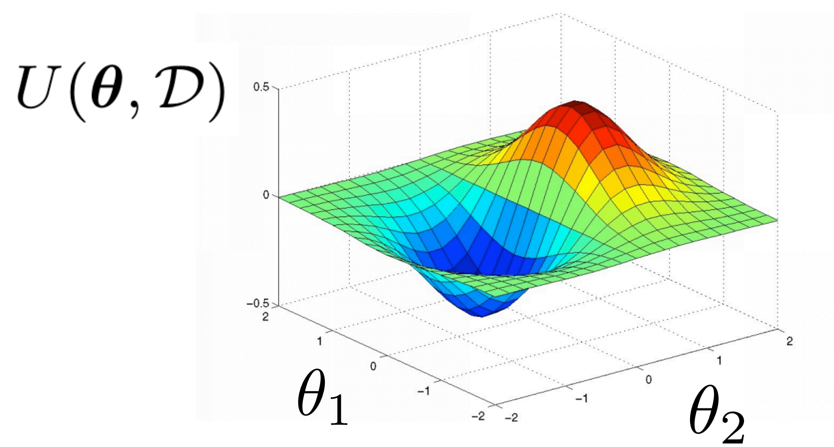
How many parameters are really needed?

Objective Landscapes

- Objective: $U(\boldsymbol{\theta}, \mathcal{D}) \triangleq \underbrace{-\sum_{i=1}^{N} \log p(\boldsymbol{d}_i|\boldsymbol{\theta})}_{\text{loss function}} \underbrace{- \log p(\boldsymbol{\theta})}_{\text{regularizer}}$

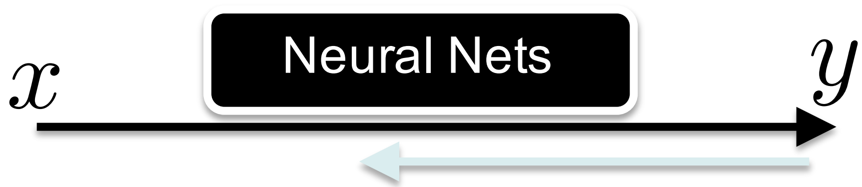
  - *Datasets:* $\mathcal{D} = \{\boldsymbol{d}_i\}_{i=1}^{N}$ $\boldsymbol{d}_i \triangleq (x_i, y_i)$
  - *Neural Nets:* $\theta \in \mathbb{R}^D$

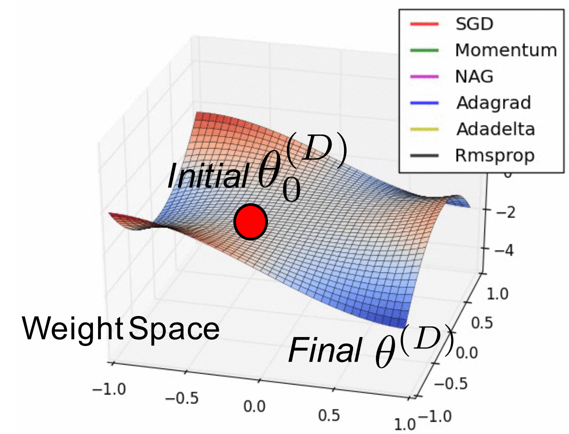
One Example of **Objective Landscapes**



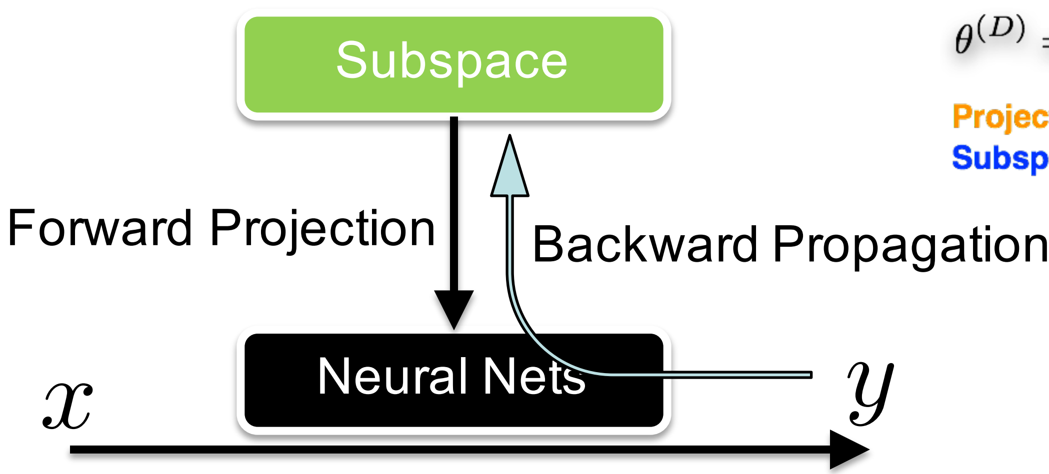
Direct vs. Subspace Training

Direct Training: Training in the Original Weight Space



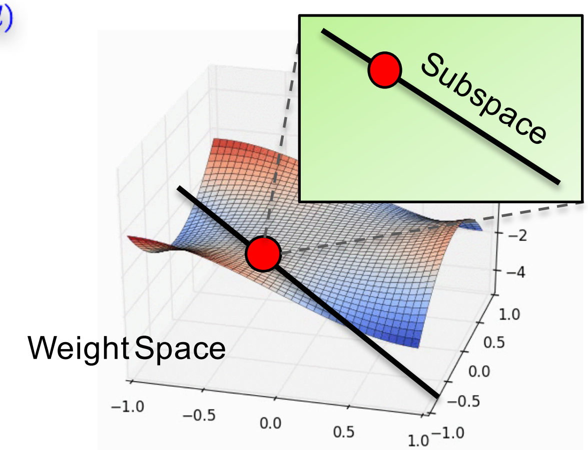


Subspace Training: Training in a Low Dimensional Orthogonal Space



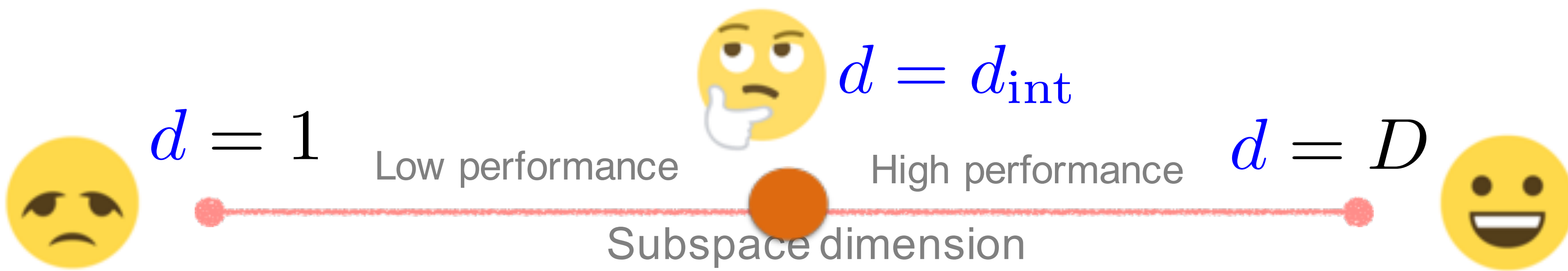
$$\theta^{(D)} = \theta_0^{(D)} + P\theta^{(d)}$$

Projection matrix
Subspace parameters



Intrinsic dimension

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

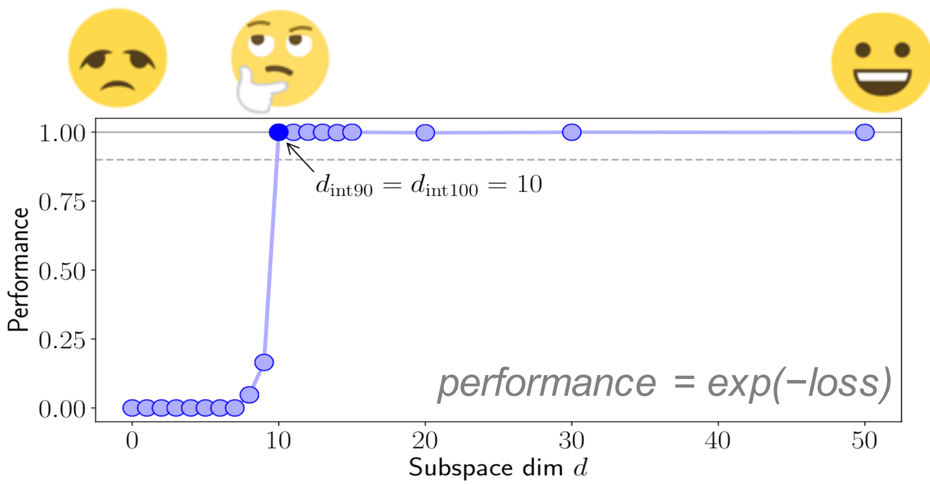
The entire space is spanned;
Any available solutions can be discovered

A Toy Problem

- A simple objective with **1000 parameters** to optimize:

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

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

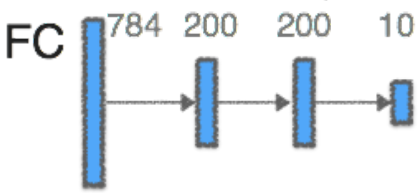


- **Generalize the concept to Neural Nets**:
  - **Datasets**: *Provide the pair-wise fitting constraints*
  - **Neural Nets Architectures**: *Provide the functional constraints*

Some networks are very compressible

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

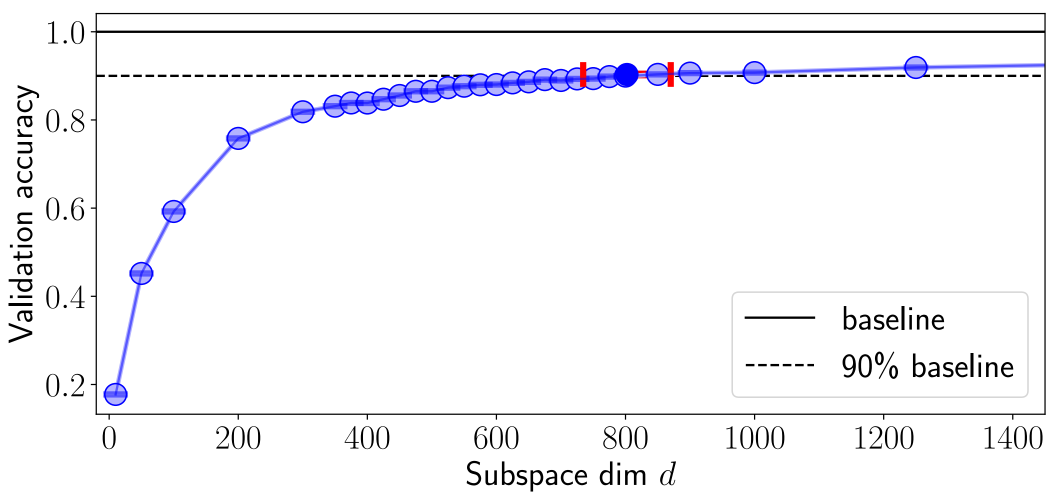
$D = 199,210$



$\mathbf{d}_{\text{int}} = \mathbf{750}$

Redundancy of the solution

$S \geq D - d_{\text{int}}$



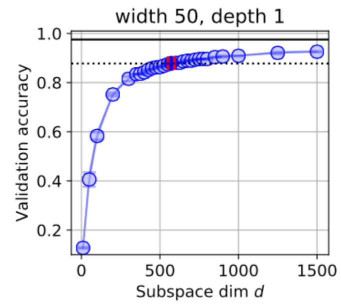
- **750 is less** than the number of input pixels (784)
  (A lot of image pixels are always black)
- **High compression rate**: 0.4%. Storage only requires 750 parameters + 1 seed
- **Highly redundant solution**: S > 199,210 - 750 = 198,460

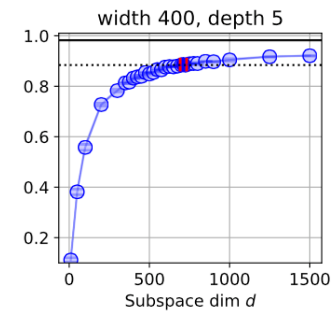
Robustness of intrinsic dimension

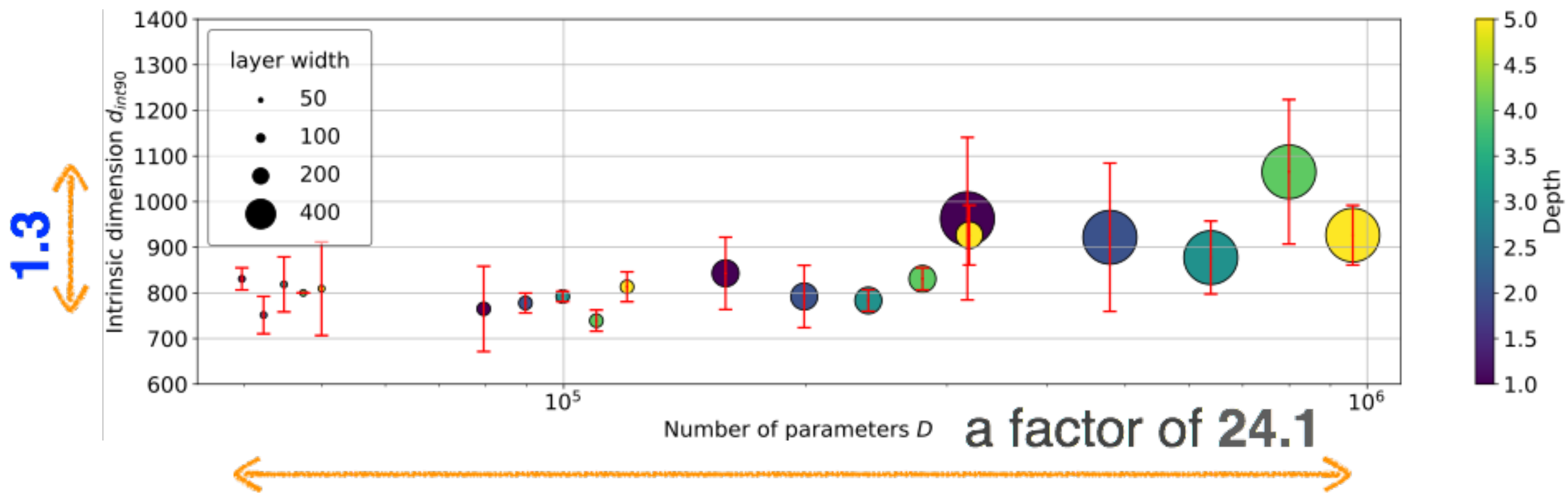
- MNIST with 20 different FC's

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

Intrinsic Dimensions as Quantitative Metrics

- Objective: $U(\boldsymbol{\theta}, \mathcal{D})$

  - Fitness of Network Architectures

    Fixing the dataset, d_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_int as 9K, 2.9K and 1K, respectively.

  - Difficulty of Tasks/Datasets

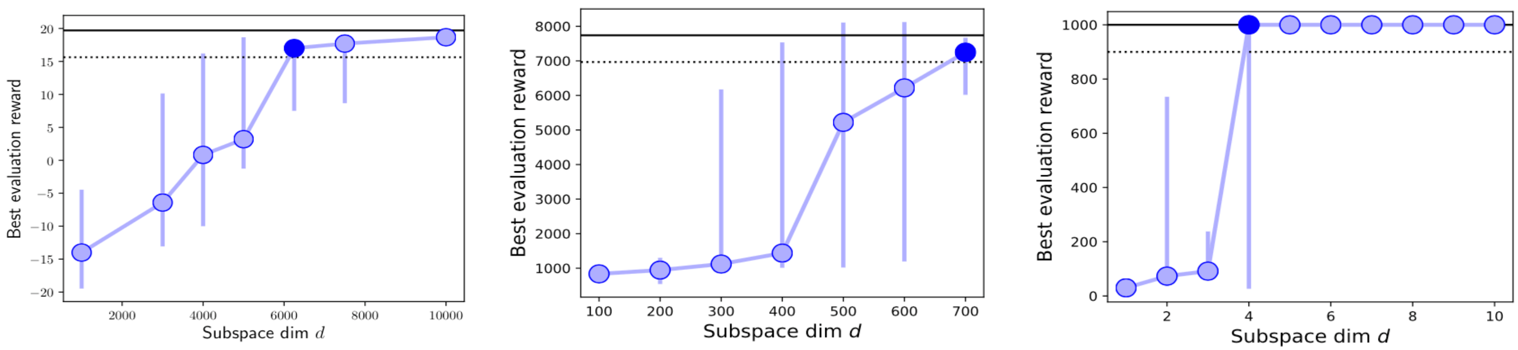
    Fixing the architecture, d_int indicates the difficulty level of specific tasks

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

    Shuffled-label MNIST: 190K; ImageNet: >800K

Policy-based RL

Evolutionary Strategies (ES)



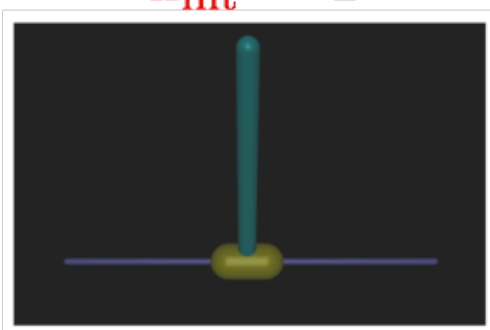
$\mathbf{d_{int} = 6,000}$ ~ CIFAR

Atari Pong

$\mathbf{d_{int} = 700}$ ~ MNIST

Humanoid

$\mathbf{d_{int} = 4}$



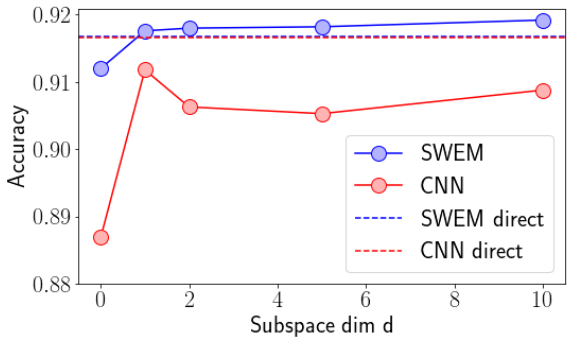
Inverted Pendulum

The low d_{int} suggests why random search and gradient-free methods work

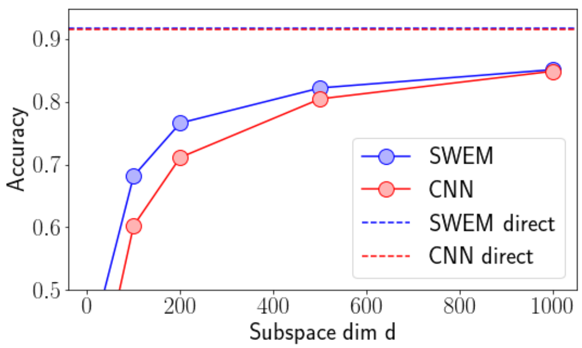
Recent Development

Applications to Guiding Model Selection

D. Shen et al. **ACL** 2018. Baseline Need More Love: On Simple Word-Embedding-Based Models (SWEM)



Training embedding

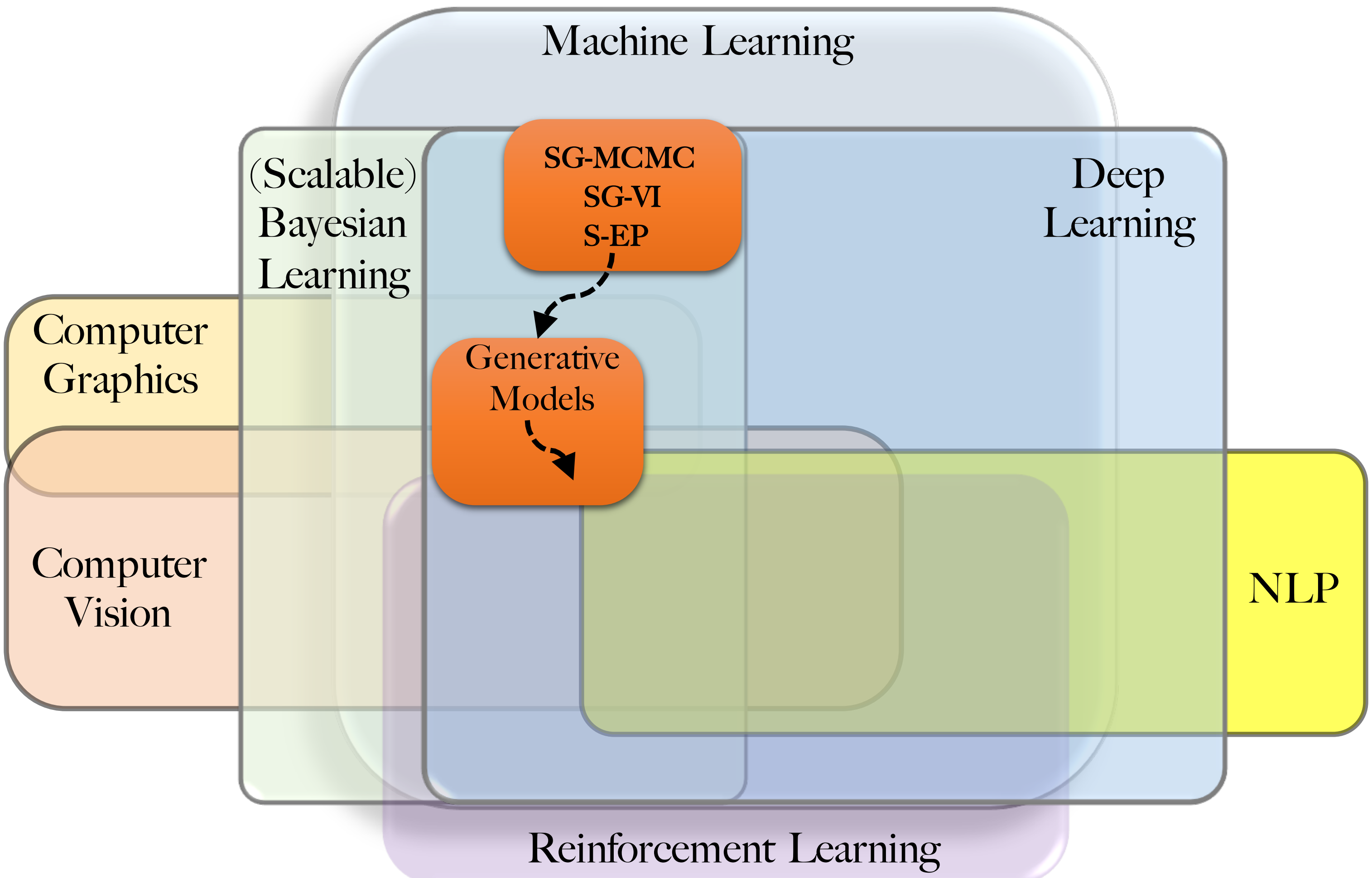


Fixing embedding

More Resource

- UBER Blog: https://eng.uber.com/intrinsic-dimension/
- Code: https://github.com/uber-research/intrinsic-dimension
- YouTube Video

Summary: Learning Trajectory



The end

Thanks!

Chunyuan Li

Duke University

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