

# Deep Generative Models



Zhiping (Patricia) Xiao  
University of California, Los Angeles

2021 Winter

## Introduction

## Deep Generative Models

Early Forms of DGMs

DGMs Training Procedure

Modern DGMs

A Unified View of DGMs

## Applications

Generative Adversarial Networks (GANs)

Normalizing Flow (NF)

Integrating Domain Knowledge into Deep Learning

# Introduction

---



This lecture is about a unifying theoretical perspective of DGMs. The reason why we are interested:

- ▶ Trending: most popular research topic nowadays (CVPR, ICML, NeurIPS, etc.)
- ▶ Promising: style transfer/fusion, music/image/text generations, etc.

Generative vs. Discriminative models:

- ▶  $\mathbb{P}(X, Y)$  vs.  $\mathbb{P}(Y|X)$ ;
- ▶ Estimate distribution (G) instead of just boundaries (D);
- ▶ etc.

Deep: multiple layers of hidden variables.

Prof. Eric Xing's lecture 12 & 13.<sup>1</sup>

- ▶ Lecture scribe: 12 & 13
- ▶ Lecture slides: 12 & 13
- ▶ Lecture record: 12 & 13

Prerequisites:

- ▶ Variational Inferences (Lecture 7&8 in Prof. Xing's lecture, in our reading group presented by Yewen.)

<sup>1</sup>[www.cs.cmu.edu/~epxing/Class/10708-20/lectures.html](http://www.cs.cmu.edu/~epxing/Class/10708-20/lectures.html)

# Deep Generative Models

---



A kind of Hierarchical Bayesian Model. We estimate the hidden values and the parameters to approximate the observations.

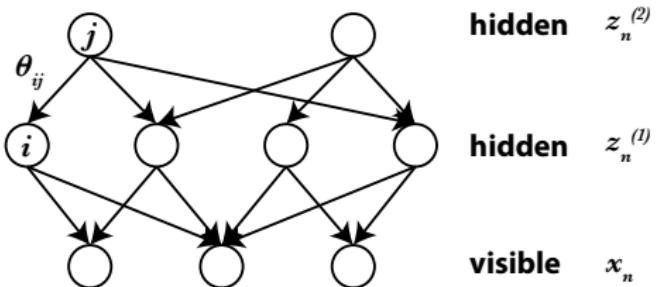


Figure: SBMs: lower layers are connected to upper layers via sigmoid functions. Use  $\theta_k$  to denote all parameters connected to  $x_{k,n}$ ;  $\theta_i$  for that connected to  $z_{i,n}^{(1)}$  from  $z_{*,n}^{(2)}$ . Then:

$$p(x_{k,n} = 1 | \theta_k, z_n^{(1)}) = \sigma(\theta_k^T z_n^{(1)}); p(z_{i,n}^{(1)} = 1 | \theta_i, z_n^{(2)}) = \sigma(\theta_i^T z_n^{(2)}).$$

A dual, alternative process that unifies **inference** and **generative** process:

- ▶ run **generative** model on input  $p_\theta(\mathbf{X})$ , and also run **inference** model on hidden values  $p_\phi(\mathbf{h})$ .
- ▶ Use the **process** instead of a global math expression to define the model.
- ▶ inference and generative models may or may not be related.

$$\mathbf{X}_n = G_\theta(\mathbf{X}_{n-1}), \quad \mathbf{X}_{n-1} = F_\phi(\mathbf{X}_n)$$

Defines a **training procedure**. Not “model” in a rigorous way.

- ▶ Using alternative loss-functions. Containing an **encoder** network and a **predictor** network.
- ▶ Use the **training procedure** instead of a global math expression to define the model.

Suppose the latent representation (code) is  $\mathbf{y} \in \mathbb{R}^m$ ,  $y_i \in [0, 1]$ , then the predictor minimizes the prediction error on  $\mathbf{y}$ , while the encoder maximizes the prediction error (e.g. mean square error).

Restricted Boltzmann machines (RBMs) [Smolensky, 1986]

- ▶ Equivalent to an infinitely-deep sigmoid network.

Deep belief networks (DBNs) [Hinton et al., 2006]

- ▶ Inference in DBNs is problematic due to “*explaining away*” (e.g. one observation  $A$ , two potential causes  $B$  and  $C$ , symptom  $A$  makes both  $B$  and  $C$  become more likely, but once you pick a cause, then the other’s probability goes back down <sup>5</sup>);
- ▶ Hybrid graphical model, some layers directed, some layers undirected.

Deep Boltzmann Machines (DBMs) [Salakhutdinov & Hinton, 2009]

- ▶ Undirected model.

Variational autoencoders (VAEs) [Kingma & Welling, 2014] / Neural Variational Inference and Learning (NVIL) [Mnih & Gregor, 2014]

- ▶ The first modern actively-used DGMs.
- ▶ Old ideas (generative model  $p_\theta(\mathbf{x}|\mathbf{z})$  and inference model  $q_\phi(\mathbf{z}|\mathbf{x})$ ) but excellent executions, produce very nice results.
- ▶ Still, the two models can be very different.
- ▶ Trained in a variational way.

Generative adversarial networks (GANs) [Goodfellow et al., 2014]

- ▶ Defining a procedure, again, not really a “model”.  
Alternatively train  $G_\theta$  and  $D_\phi$ .

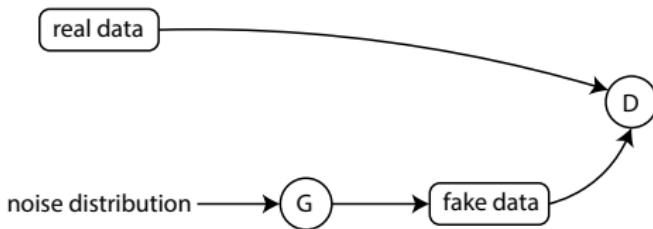


Figure: GAN:

$$\min_G \max_D \mathbb{E}_{\mathbf{x}_{real}} (\log(D(\mathbf{x}_{real}))) + \mathbb{E}_{\mathbf{x}_{fake}} (\log(1 - D(G(\mathbf{x}_{fake}))))$$

And countless ideas following them. We have a zoo of such models.

<sup>5</sup>Reference slides on “explaining away”.

## Posterior Distribution / Inference model

- ▶ Variational approximation
- ▶ Recognition model
- ▶ Inference network (if parameterized as neural networks)
- ▶ Recognition network (if parameterized as neural networks)
- ▶ (Probabilistic) encoder

## “The Model” (prior + conditional, or joint) / Generative model

- ▶ The (data) likelihood model
- ▶ Generative network (if parameterized as neural networks)
- ▶ Generator
- ▶ (Probabilistic) decoder

Training of early forms of DGMs typically uses EM framework.

- ▶ via sampling / data augmentation: directly infer hidden variable, given observations  $p(\mathbf{z}|\mathbf{x})$

$$\mathbf{z} = \{\mathbf{z}_1, \mathbf{z}_2\}$$

$$\mathbf{z}_1^{new} \sim p(\mathbf{z}_1|\mathbf{z}_2, \mathbf{x})$$

$$\mathbf{z}_2^{new} \sim p(\mathbf{z}_2|\mathbf{z}_1^{new}, \mathbf{x})$$

- ▶ variational inference: generator parameters  $\theta$ , variational inference model parameters  $\phi$ , optimizing an variational lower bound:

$$\log(p(\mathbf{x})) \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{x}|\mathbf{z}))] + \text{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) := \mathcal{L}(\theta, \phi; \mathbf{x})$$

$$\max_{\theta, \phi} \mathcal{L}(\theta, \phi; \mathbf{x})$$

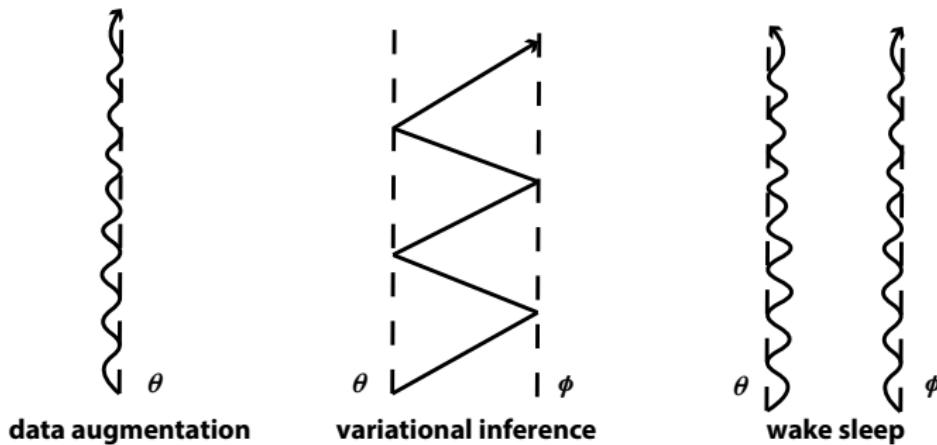


Figure: Illustration of the training methods' differences.

- ▶ wake sleep: the **loss**-functions become **different**

$$\text{Wake: } \min_{\theta} \mathbb{E}_{q_{\phi}(\mathbf{z}|\mathbf{x})} [\log(p_{\theta}(\mathbf{x}|\mathbf{z}))]$$

$$\text{Sleep: } \min_{\phi} \mathbb{E}_{p_{\theta}(\mathbf{x}|\mathbf{z})} [\log(q_{\phi}(\mathbf{z}|\mathbf{x}))]$$

Consider a generative model  $p_\theta(\mathbf{x}|\mathbf{z})$ , and prior  $p(\mathbf{z})$ ; we have joint distribution:

$$p_\theta(\mathbf{x}, \mathbf{z}) = p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})$$

Assume variational distribution  $q_\phi(\mathbf{z}|\mathbf{x})$ ;

Objective: Maximize lower bound for log likelihood.

$$\log(p(\mathbf{x})) \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{x}|\mathbf{z}))] + \text{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) := \mathcal{L}(\theta, \phi; \mathbf{x})$$

where KL refers to KL Divergence ( $p, q$  are distributions):<sup>6</sup>

$$\text{KL}(q||p) = \sum_x q(x) \log \frac{q(x)}{p(x)}$$

There are **multiple ways** of expressing its objectives.

<sup>6</sup> references of KL.

Maximizing the variational lower bound:

$$\begin{aligned}\mathcal{L}(\theta, \phi; \mathbf{x}) &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{x}|\mathbf{z}))] + \text{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) \\ &= \log p(\mathbf{x}) - \text{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x}))\end{aligned}$$

“E-Step”: <sup>7</sup>

$$\max_{\phi} \mathcal{L}(\theta, \phi; \mathbf{x})$$

“M-Step”:

$$\max_{\theta} \mathcal{L}(\theta, \phi; \mathbf{x})$$

Equivalently: minimize free energy.

$$\mathcal{F}(\theta, \phi; \mathbf{x}) = -\log p(\mathbf{x}) + \text{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x}))$$

<sup>7</sup>To call it EM is misleading but there is a correspondence.

Maximize data log-likelihood with two steps of loss relaxation.

Wake phase:  $\min_{\theta} \mathbb{E}_{q_{\phi}(\mathbf{z}|\mathbf{x})} [\log(p_{\theta}(\mathbf{x}|\mathbf{z}))]$

- ▶ Maximize the variational lower bound of log-likelihood, or minimizing free energy (original goal)

$$\mathcal{F}(\theta, \phi; \mathbf{x}) = -\log p(\mathbf{x}) + \text{KL}(q_{\phi}(\mathbf{z}|\mathbf{x}) || p_{\theta}(\mathbf{z}|\mathbf{x}))$$

- ▶ Correspond to variational **M step**).
- ▶ Get samples from  $q_{\phi}(\mathbf{z}|\mathbf{x})$  through inference on hidden variables.

Sleep phase:  $\min_{\phi} \mathbb{E}_{p_{\theta}(\mathbf{x}|\mathbf{z})}[\log(q_{\phi}(\mathbf{z}|\mathbf{x}))]$ , simplified as  
 $\min_{\phi} \mathbb{E}_{p_{\theta}(\mathbf{x}, \mathbf{z})}[\log(q_{\phi}(\mathbf{z}|\mathbf{x}))]$  (for the ease of optimization).

$$\nabla_{\phi} \mathcal{F}(\theta, \phi; \mathbf{x}) = \dots + \nabla_{\phi} \mathbb{E}_{q_{\phi}(\mathbf{z}|\mathbf{x})}[\log(p_{\theta}(\mathbf{z}|\mathbf{x}))] + \dots$$

includes the high variance term  $\log p_{\theta}$ , but could be estimated with the log-derivative trick:

$$\begin{aligned}\nabla_{\phi} \mathbb{E}_{q_{\phi}}[\log p_{\theta}] &= \int \nabla_{\phi} q_{\phi} \log p_{\theta} = \int q_{\phi} \log p_{\theta} \nabla_{\phi} \log q_{\phi} \\ &= \mathbb{E}_{q_{\phi}}[\log p_{\theta} \nabla_{\phi} \log q_{\phi}]\end{aligned}$$

estimated by Monte Carlo ( $\log p_{\theta}(\mathbf{x}, \mathbf{z}_i)$  can be arbitrarily large):

$$\nabla_{\phi} \mathbb{E}_{q_{\phi}}[\log p_{\theta}] \approx \mathbb{E}_{\mathbf{z}_i \sim q_{\phi}}[\log p_{\theta}(\mathbf{x}, \mathbf{z}_i) \nabla_{\phi} q_{\phi}(\mathbf{z}_i|\mathbf{x})]$$

- ▶ Minimize a different objective (reversed KLD) wrt  $\phi$  to ease the optimization:

$$\mathcal{F}'(\theta, \phi; \mathbf{x}) = -\log p(\mathbf{x}) + \text{KL}(p_\theta(\mathbf{z}|\mathbf{x}) || q_\phi(\mathbf{z}|\mathbf{x}))$$

- ▶ Correspond to the variational **E step**.
- ▶ Why changing objective: original objective is suffering from high variance caused by the gradient of the original KL term, and therefore it is generally intractable.
- ▶ “Dreaming” up samples from  $p_\theta(\mathbf{x}|\mathbf{z})$  through top-down pass.
- ▶ Doing something “wrong” but not “too wrong”.

## Variational Inference

- ▶ Distribution  $q_\phi(\mathbf{z}|\mathbf{x})$
- ▶ M Step:  
 $\min_\theta \text{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x}))$ 
  - ▶  $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\nabla_\theta \log(p_\theta(\mathbf{x}|\mathbf{z}))]$
- ▶ E Step:  
 $\min_\phi \text{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x}))$ 
  - ▶  $\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{x}|\mathbf{z}))]$
  - ▶ High variance, need variance-reduce in practice
  - ▶ Learning with real samples of  $\mathbf{x}$ .
- ▶ Single objective, guaranteed to converge

## Wake Sleep

- ▶ Inference model  $q_\phi(\mathbf{z}|\mathbf{x})$
- ▶ Wake:  
 $\min_\theta \text{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x}))$ 
  - ▶  $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\nabla_\theta \log(p_\theta(\mathbf{x}|\mathbf{z}))]$
- ▶ Sleep:  
 $\min_\phi \text{KL}(p_\theta(\mathbf{z}|\mathbf{x})||q_\phi(\mathbf{z}|\mathbf{x}))$ 
  - ▶  $\mathbb{E}_{p_\theta(\mathbf{z}, \mathbf{x})}[\nabla_\phi \log(q_\phi(\mathbf{z}, \mathbf{x}))]$
  - ▶ Low variance
  - ▶ Learning with generated samples of  $\mathbf{x}$ .
- ▶ Two objective, not guaranteed to converge

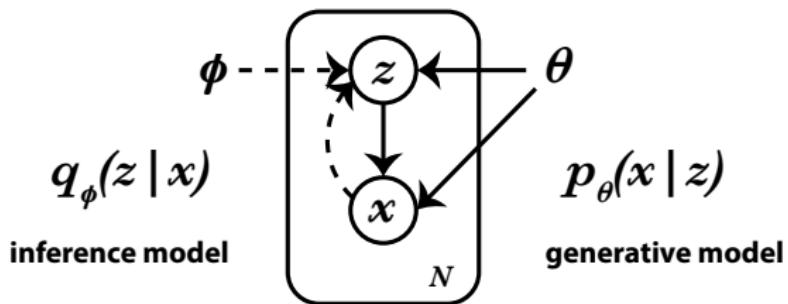


Figure: Key Ideas: Inference model and Generative model. Prior  $p(\mathbf{z})$ , joint distribution  $p_\theta(\mathbf{x}, \mathbf{z}) = p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})$ . Use variational inference with an inference model, enjoy similar applicability with wake-sleep algorithm. [Kingma & Welling, 2014]

Variational lower bound:

$$\mathcal{L}(\theta, \phi; \mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}, \mathbf{z})] - \text{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))$$

Recall that for a variational inference model we suffer from large variance in sleep / E phase:

$$\nabla_\phi \mathcal{F}(\theta, \phi; \mathbf{x}) = \dots + \nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{z}|\mathbf{x}))] + \dots$$

This time the variance is reduced via **reparameterization trick**. (Alternatives: use control variates as in reinforcement learning.)

Reparameterization trick in gradient estimation of the inference model:

1. Assume a trivial noise distribution (e.g. standard Gaussian):  $\epsilon \sim p(\epsilon)$
2. Do a deterministic transformation:

$$\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x}) \iff \mathbf{z} = g_\phi(\epsilon, \mathbf{x})$$

3. Reparameterized expression e.g.:

$$\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} [\log(p_\theta(\mathbf{x}, \mathbf{z}))] = \mathbb{E}_{\epsilon \sim p(\epsilon)} [\nabla_\phi \log p_\theta(\mathbf{x}, \mathbf{z}_\phi(\epsilon))]$$

has empirically lower variance of the gradient estimate.



Figure: Celebrity faces generated (Radford 2015). VAEs tend to generate **blurred** images due to the mode covering behavior.

Mode-covering behavior has something to do with the KL divergence: reference.

Defining a procedure involving a generator  $G_\theta$  and a discriminator  $D_\phi$ .

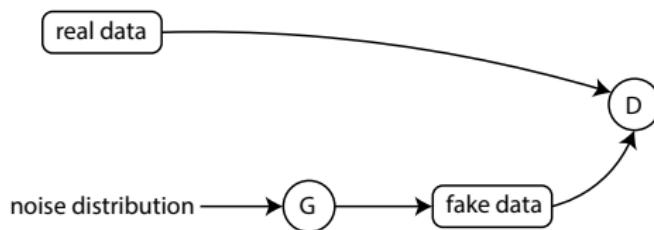


Figure: [Goodfellow et al., 2014] GAN:  
 $\min_G \max_D \mathbb{E}_{\mathbf{x}_{real}} (\log(D(\mathbf{x}_{real}))) + \mathbb{E}_{\mathbf{x}_{fake}} (\log(1 - D(G(\mathbf{x}_{fake}))))$

**D** (discriminator, output 1 for real data and 0 for fake data) is trained first and then **G** in each iteration. While training **D** we minimize:

$$\ell_D = -\mathbb{E}_{\mathbf{x}_{real}}(\log(D(\mathbf{x}_{real}))) - \mathbb{E}_{\mathbf{x}_{fake}}(\log(1 - D(G(\mathbf{x}_{fake}))))$$

and while training **G** we minimize:

$$\ell_G = \log(1 - D(G(\mathbf{x}_{fake})))$$

where  $\mathbf{x}_{real}$  are sampled from real data  $\mathbf{x}_{real} \sim p_{data}(\mathbf{x})$  and  $\mathbf{x}_{fake}$  is sampled from a noise distribution  $\mathbf{x}_{fake} \sim p_{noise}(\mathbf{x})$ .

Learning goal is to achieve equilibrium of the game, optimal state:

- ▶ Generated distribution is identical to the real distribution.
- ▶  $D(\mathbf{x}) = \frac{1}{2}$



Figure: Generated bedrooms (Radford et al., 2016). GANs tend to generate **sharp** images but very **narrow** (focusing on a few areas e.g. the bed).

Analogy: from *Alchemy* to *modern Chemistry*.

- ▶ Basic elements are concluded in a unified way;
- ▶ Rules are found accordingly;
- ▶ No need to try countless times until getting some “Hail Mary results” with luck.

Paper: On Unifying Deep Generative Models [Z Hu, Z YANG, R Salakhutdinov, E Xing]

- ▶ GANs: achieve an equilibrium between generator and discriminator
- ▶ VAEs: maximize lower bound of the data likelihood

Is there a way of making DGMs expressions a little bit similar with each other?

- ▶ GAN objective in variational-EM format;
- ▶ VAE's new formulation (and comparision to GAN);
- ▶ Linking GAN and VAE to Wake-Sleep.

To model a distribution:

$\mathbf{x} \sim p_\theta(\mathbf{x}|y) \iff \mathbf{x} = G_\theta(\mathbf{z}), \mathbf{z} \sim p(\mathbf{z}|y=0)$  where

$$p_\theta(\mathbf{x}|y) = \begin{cases} p_{g_\theta}(\mathbf{x}) & y = 0 \\ p_{\text{data}}(\mathbf{x}) & y = 1 \end{cases}$$

Conventional formulation ( $\mathbf{z} \sim p_{noise}(\mathbf{z})$ ):

$$\min_{\theta} \max_{\phi} \mathbb{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} [\log(D_{\phi}(\mathbf{x}))] + \mathbb{E}_{\mathbf{x} \sim G_{\theta}(\mathbf{z})} [\log(1 - D_{\phi}(\mathbf{x}))]$$
$$\left\{ \begin{array}{l} \max_{\phi} \mathcal{L}_{\phi} = \mathbb{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} [\log(D_{\phi}(\mathbf{x}))] + \mathbb{E}_{\mathbf{x} \sim G_{\theta}(\mathbf{z})} [\log(1 - D_{\phi}(\mathbf{x}))] \\ \max_{\theta} \mathcal{L}_{\theta} = \mathbb{E}_{\mathbf{x} \sim G_{\theta}(\mathbf{z})} [\log(D_{\phi}(\mathbf{x}))] \end{array} \right.$$

The new form:

$$\left\{ \begin{array}{l} \max_{\phi} \mathcal{L}_{\phi} = \mathbb{E}_{p_{\theta}(\mathbf{x}|y)p(y)} [\log(q_{\phi}(y|\mathbf{x}))] \\ \max_{\theta} \mathcal{L}_{\theta} = \mathbb{E}_{p_{\theta}(\mathbf{x}|y)p(y)} [\log(q_{\phi}(1-y|\mathbf{x}))] \end{array} \right.$$

where  $q_{\phi}(1-y|\mathbf{x})$  can also be denoted as  $q_{\phi}^r(y|\mathbf{x})$ .

## Variational EM

$$\begin{aligned}\mathcal{L}(\theta, \phi; \mathbf{x}) = & \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{x}|\mathbf{z}))] \\ & + KL(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))\end{aligned}$$

$$\max_{\theta} \mathcal{L}(\theta, \phi; \mathbf{x}), \quad \max_{\phi} \mathcal{L}(\theta, \phi; \mathbf{x})$$

## GANs

$$\max_{\phi} \mathcal{L}_\phi = \mathbb{E}_{p_\theta(\mathbf{x}|y)p(y)}[\log(q_\phi(y|\mathbf{x}))]$$

$$\max_{\theta} \mathcal{L}_\theta = \mathbb{E}_{p_\theta(\mathbf{x}|y)p(y)}[\log(q_\phi(1 - y|\mathbf{x}))]$$

- ▶ Single objective
- ▶ Generative model:  $p_\theta(\mathbf{x}|\mathbf{z})$
- ▶ Inference model:  $q_\phi(\mathbf{z}|\mathbf{x})$
- ▶ The reconstruction term  $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{x}|\mathbf{z}))]$  is similar to GANs' objectives.

- ▶ Two objectives
- ▶ Interpret  $q_\phi(y|\mathbf{x})$  as the generative model
- ▶ Interpret  $p_\theta(\mathbf{x}|y)$  as the inference model
- ▶ Doesn't exist prior regularization of  $p(\mathbf{z})$ .

Recall that in maximizing the variational lower bound:

$$\begin{aligned}\mathcal{L}(\theta, \phi; \mathbf{x}) &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log(p_\theta(\mathbf{x}|\mathbf{z}))] + KL(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) \\ &= \log p(\mathbf{x}) - KL(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x}))\end{aligned}$$

That is, we minimized the KLD from the inference model to the posterior:

$$-\log p(\mathbf{x}) + KL(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x}))$$

Starting from an initial point  $(\theta_0, \phi_0)$ , let  $p(y)$  be a uniform prior distribution, and

$$\begin{aligned} p_{\theta=\theta_0}(\mathbf{x}) &= \mathbb{E}_{p(y)}[p_{\theta=\theta_0}(\mathbf{x}|y)] \\ q^r(\mathbf{x}|y) &\propto q^r(y|\mathbf{x})p_{\theta=\theta_0}(\mathbf{x}) \end{aligned}$$

Lemma (update rule for  $\theta$ )<sup>8</sup>

$$\begin{aligned} &\nabla_{\theta} \mathbb{E}_{p_{\theta}(\mathbf{x}|y)p(y)} [\log(q_{\phi=\phi_0}^r(y|\mathbf{x}))] \Big|_{\theta=\theta_0} \\ &= \nabla_{\theta} (\mathbb{E}_{p(y)} [\text{KL}(p_{\theta}(\mathbf{x}|y)||q^r(\mathbf{x}|y)) - \text{JSD}(p_{\theta}(\mathbf{x}|y=0)||p_{\theta}(\mathbf{x}|y=1))] ) \Big|_{\theta=\theta_0} \end{aligned}$$

<sup>8</sup>JSD = Jensen-Shannon divergence, KL = KL divergence.

Lemma (update rule for  $\theta$ )

$$\begin{aligned} & \nabla_{\theta} \mathbb{E}_{p_{\theta}(\mathbf{x}|y)p(y)} [\log(q_{\phi=\phi_0}^r(y|\mathbf{x}))] \Big|_{\theta=\theta_0} \\ &= \nabla_{\theta} (\mathbb{E}_{p(y)} [\text{KL}(p_{\theta}(\mathbf{x}|y) || q^r(\mathbf{x}|y))] - \text{JSD}(p_{\theta}(\mathbf{x}|y=0) || p_{\theta}(\mathbf{x}|y=1))) \Big|_{\theta=\theta_0} \end{aligned}$$

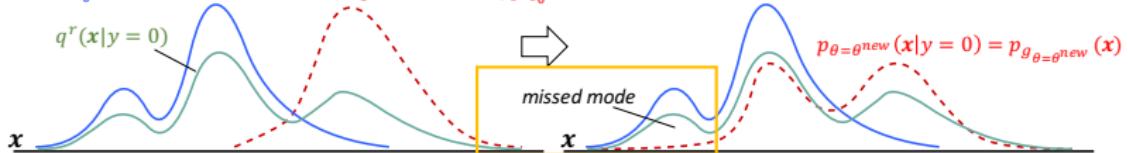
Connection to variational inference:

- ▶ See  $\mathbf{x}$  as latent variables,  $y$  as visible;
- ▶  $p_{\theta=\theta_0}(\mathbf{x})$  as prior distribution,  $q^r(\mathbf{x}|y)$  as posterior distribution,  $p_{\theta}(\mathbf{x}|y)$  as variational distribution.



## GANs: minimizing KLD

$$p_{\theta=\theta_0}(x|y=1) = p_{data}(x) \quad p_{\theta=\theta_0}(x|y=0) = p_{g_{\theta=\theta_0}}(x)$$



- Missing mode phenomena of GANs
  - Asymmetry of KLD
  - Concentrates  $p_{\theta}(x|y=0)$  to large modes of  $q^r(x|y=0)$   
⇒  $p_{g_{\theta}}(x)$  misses modes of  $p_{data}(x)$
- Symmetry of JSD
  - Does not affect the behavior of mode missing

$$\text{KL}\left(p_{g_{\theta}}(x)||q^r(x|y=0)\right) = \int p_{g_{\theta}}(x) \log \frac{p_{g_{\theta}}(x)}{q^r(x|y=0)} dx$$

- Large positive contribution to the KLD in the regions of  $x$  space where  $q^r(x|y=0)$  is small, unless  $p_{g_{\theta}}(x)$  is also small
- ⇒  $p_{g_{\theta}}(x)$  tends to avoid regions where  $q^r(x|y=0)$  is small



Figure: Blue: prior distribution of real data, Green: posterior distribution of the synthetic data, Red: variational distribution of synthetic data.

<sup>9</sup>You won't pop up unless you have adequate samples.

VAE: maximizing the variational lower bound:

$$\mathcal{L}(\theta, \phi; \mathbf{x}) = \mathbb{E}_{p_{data}(\mathbf{x})} [\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} [\log p_\theta(\mathbf{x}, \mathbf{z})] - \text{KL}(q_\phi(\mathbf{z}|\mathbf{x}) || p(\mathbf{z}))]$$

To align VAE with GAN, we introduce the real/fake indicator  $y$  and adversarial discriminator.

Beside  $\mathbf{x}$  the code / observation / example /etc., and  $\mathbf{z}$  the hidden state / latent representation, we introduce  $y$ , together with a perfect discriminator  $q_*(y|\mathbf{x})$ .

$$q_*(y = 1|\mathbf{x}) = 1 \text{ if } \mathbf{x} \text{ is real}$$

$$q_*(y = 0|\mathbf{x}) = 1 \text{ if } \mathbf{x} \text{ is generated}$$

and also a generative distribution: <sup>10</sup>

$$p_\theta(\mathbf{x}|\mathbf{z}, y) = \begin{cases} p_\theta(\mathbf{x}|\mathbf{z}) & y = 0 \\ p_{data}(\mathbf{x}) & y = 1 \end{cases}$$

<sup>10</sup>This format is similar to InfoGAN.

Also, let the posterior  $p_\theta(\mathbf{z}, y|\mathbf{x}) \propto p_\theta(\mathbf{z}, y|\mathbf{x})p(\mathbf{z}|y)p(y)$ :<sup>11</sup>

Lemma (New Objective of VAE at  $(\theta_0, \phi_0)$  )

$$\begin{aligned}\mathcal{L}(\theta, \phi; \mathbf{x}) &= \mathbb{E}_{p_{data}(\mathbf{x})} [\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} [\log p_\theta(\mathbf{x}, \mathbf{z})] - \text{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))] \\ &= 2\mathbb{E}_{p_{\theta_0}(\mathbf{x})} [\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}, y)q_*^r(y|\mathbf{x})} [\log p_\theta(\mathbf{x}|\mathbf{z}, y)] \\ &\quad - \text{KL}(q_\phi(\mathbf{z}|\mathbf{x}, y)q_*^r(y|\mathbf{x})||p(\mathbf{z}|y)p(y))] \\ &= 2\mathbb{E}_{p_{\theta_0}(\mathbf{x})} [-\text{KL}(q_\phi(\mathbf{z}|\mathbf{x}, y)q_*^r(y|\mathbf{x})||p(\mathbf{z}, y|\mathbf{x}))]\end{aligned}$$

<sup>11</sup> $p(\cdot)$  are fixed priors.

The KLD to minimize for VAE is:

$$\text{KL}(q_\phi(\mathbf{z}|\mathbf{x}, y) q^r(y|\mathbf{x}) || p(\mathbf{z}, y|\mathbf{x}))$$

Recall that in the new form of GAN, the KLD to minimize:

$$\text{KL}(p_\theta(\mathbf{x}|y) || q^r(\mathbf{x}|y))$$

There is a major difference here: GANs KL term does  $\min_\theta(P_\theta||Q)$  and VAEs does  $\min_\theta(Q||P_\theta)$ .<sup>12</sup>

- ▶ GANs:  $\min_\theta(P_\theta||Q)$  tends to missing mode, ignoring regions with small values of  $p_{data}$ ;
- ▶ VAEs:  $\min_\theta(Q||P_\theta)$  tends to cover regions with small values of  $p_{data}$ .

<sup>12</sup>KLD Asymmetry inspires combination of GANs and VAEs.

Recall Wake Sleep: two loss-functions are used.

$$\text{Wake: } \min_{\theta} \mathbb{E}_{q_{\phi}(\mathbf{z}|\mathbf{x})p_{data}(\mathbf{x})} [\log(p_{\theta}(\mathbf{x}|\mathbf{z}))]$$

$$\text{Sleep: } \min_{\phi} \mathbb{E}_{p_{\theta}(\mathbf{x}|\mathbf{z})p(\mathbf{z})} [\log(q_{\phi}(\mathbf{z}|\mathbf{x}))]$$

Recall VAEs objective to minimize:

$$\mathcal{L}(\theta, \phi; \mathbf{x}) = \mathbb{E}_{p_{data}(\mathbf{x})} [\mathbb{E}_{q_{\phi}(\mathbf{z}|\mathbf{x})} [\log p_{\theta}(\mathbf{x}, \mathbf{z})] - \text{KL}(q_{\phi}(\mathbf{z}|\mathbf{x}) || p(\mathbf{z}))]$$

VAE only needs the wake phase, does not need the sleep phase, and thus doesn't need the reverse-KLD trick. Stick to minimizing the wake phase KLD w.r.t.  $\theta, \phi$ .

Wake Sleep: two loss-functions are used.

$$\text{Wake: } \min_{\theta} \mathbb{E}_{q_{\phi}(\mathbf{z}|\mathbf{x})p_{data}(\mathbf{x})} [\log(p_{\theta}(\mathbf{x}|\mathbf{z}))]$$

$$\text{Sleep: } \min_{\phi} \mathbb{E}_{p_{\theta}(\mathbf{x}|\mathbf{z})p(\mathbf{z})} [\log(q_{\phi}(\mathbf{z}|\mathbf{x}))]$$

Recall GANs objective:

$$\max_{\phi} \mathcal{L}_{\phi} = \mathbb{E}_{p_{\theta}(\mathbf{x}|y)p(y)} [\log(q_{\phi}(y|\mathbf{x}))]$$

$$\max_{\theta} \mathcal{L}_{\theta} = \mathbb{E}_{p_{\theta}(\mathbf{x}|y)p(y)} [\log(q_{\phi}^r(y|\mathbf{x}))]$$

GAN is directly extending sleep phase, only difference is  $q_{\theta} \rightarrow q_{\theta}^r$ . Stick to minimizing the sleep-phase KLD.

DGMs have a long history, and is a big family.

Unification of the different DGMs is possible & useful.

- ▶ GANs and VAEs are essentially minimizing KLD in opposite directions and extend two phases of classic wake sleep algorithm, respectively;
- ▶ The general formulation is useful for analyzing a broad class of existing DGM models, and can inspire new models and algorithms.

# Applications

---



- ▶ It is trending because of its outstanding performance.
- ▶ Vanilla GAN [Goodfellow et al., 2014] objective:

$$\min_{\theta} \text{JSD}(P_{\text{data}} || P_{g_{\theta}})$$

- ▶ Note: this expression is symbolic, not executable.
- ▶ Unifying version [Hu et al. 2017] objective:

$$\min_{\theta} \text{KL}(P_{\theta} || Q)$$

Shortcoming of KLD: for  $\text{KL}(P||Q)$  if  $P$  and  $Q$  have neglectable overlap, then the KLD is degenerated, meaningless. Sometimes it becomes undefined or infinite, messing up the loss.<sup>13</sup>

In practice: if our data is a low-dimensional manifold of a high dimensional space, there can be a **negligible** intersection between the model's manifold and the true data manifold.

<sup>13</sup>Similar shortcoming applies to JSD.

Shortcoming of KLD: for  $\text{KL}(P||Q)$  if  $P$  and  $Q$  have neglectable overlap, then the KLD is degenerated, meaningless. Sometimes it becomes undefined or infinite, messing up the loss.<sup>13</sup>

In practice: if our data is a low-dimensional manifold of a high dimensional space, there can be a **negligible** intersection between the model's manifold and the true data manifold.

The loss function is re-defined via **Wasserstein Distance**.

- ▶ Well-defined in math, a.k.a Earth Mover's Distance;
- ▶ Minimum transportation cost for making pile of dirt in shape of one probability distribution to the other.

<sup>13</sup>Similar shortcoming applies to JSD.

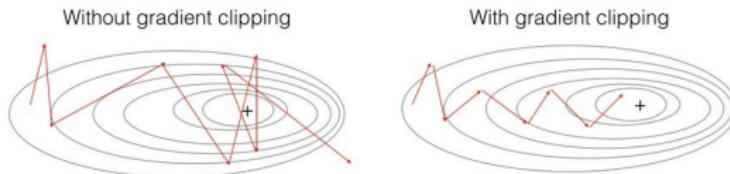


## Wasserstein GAN (WGAN)

- Objective

$$W(p_{data}, p_g) = \frac{1}{K} \sup_{\|D\|_L \leq K} \mathbb{E}_{x \sim p_{data}} [D(x)] - \mathbb{E}_{x \sim p_g} [D(x)]$$

- $\|D\|_L \leq K$  : K- Lipschitz continuous
- Use gradient-clipping to ensure  $D$  has the Lipschitz continuity



© Eric Xing (CMU, 2005-2020) 10



Figure: Lipschitz continuous: intuitively limited in how fast it can change, previously learned with convex optimization.

<sup>14</sup>[Arjovsky et al., 2017]

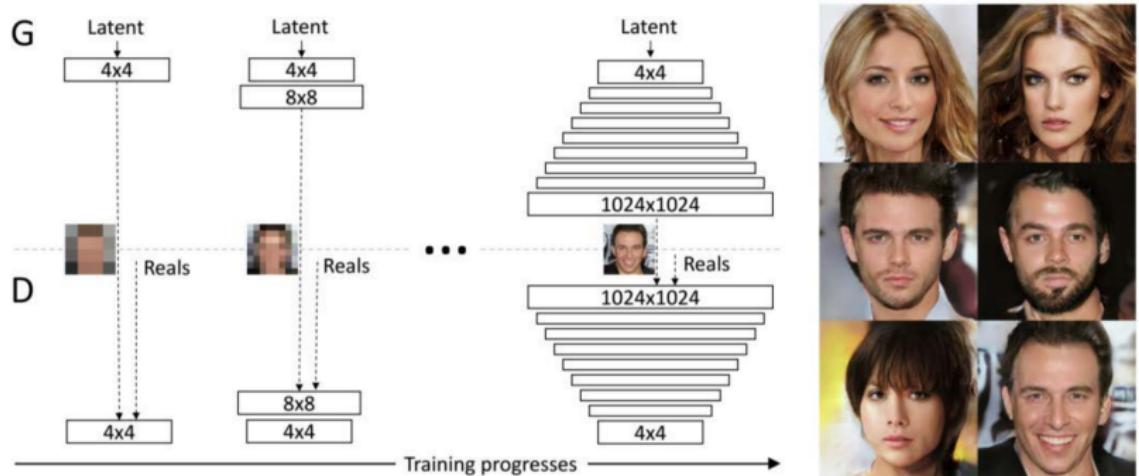


Figure: Ideas: Begin with very low-resolution images and very shallow  $G$ ,  $D$ . As training goes on, add additional layers to  $G$  &  $D$ , and use higher resolution images. By-passing the size bottleneck by not having to train the whole network at once.

GANs benefit dramatically from scaling.

They put efforts in scaling up GANs.

- ▶ 2 – 4 times more parameters to improve expressiveness;
- ▶ 8× larger batch size to avoid overfitting;
- ▶ Simple architecture changes that improve scalability.

<sup>16</sup>[Brock et al., 2018]

Idea: to amplify / transform an originally very simple model into something more complex / powerful; to transform a simple distribution into an arbitrarily complex one.

Method: applying a sequence of invertible transformation functions.<sup>17</sup>

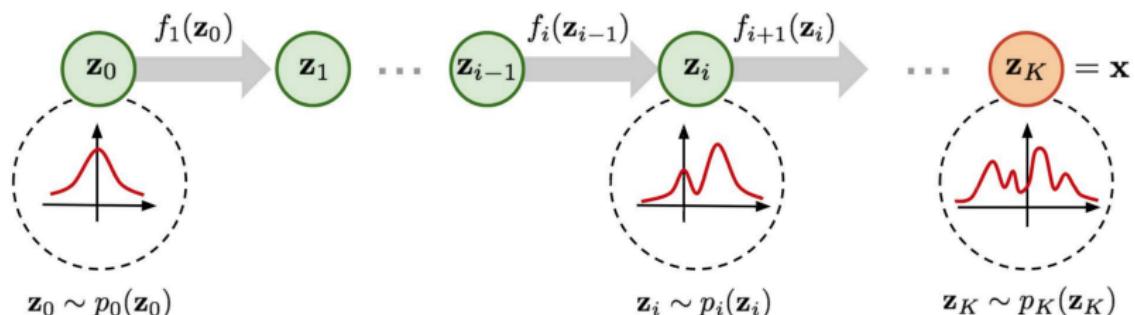


Figure: Figure from Prof. Xing's lecture slides, Figure courtesy: Lilian Weng.

<sup>17</sup>Libraries do that all the time, e.g. Uniform  $\rightarrow$  Gaussian.

Starting from  $\mathbf{z} \sim p(\mathbf{z})$ , given transformation function  $f$ , generating  $\mathbf{x} = f(\mathbf{z})$ .

- ▶ To do inference we need  $\mathbf{z} = f^{-1}(\mathbf{x})$ , thus need  $f$  to be invertible.
- ▶ To compute density we have:

$$p(\mathbf{x}) = p(\mathbf{z}) \left| \det \frac{d\mathbf{z}}{d\mathbf{x}} \right| = p(f^{-1}(\mathbf{x})) \left| \det \frac{df^{-1}}{d\mathbf{x}} \right|$$

and there are tricks of making the **Jacobian determinant**  $\det \frac{df^{-1}}{d\mathbf{x}}$  easy to compute, e.g. making  $\frac{df^{-1}}{d\mathbf{x}}$  a triangular matrix.

$$\mathbf{z}_0 \sim p(\mathbf{z}_0)$$

$$\mathbf{x} = \mathbf{z}_K = f_K \circ f_{K-1} \circ \cdots \circ f_1(\mathbf{z}_0)$$

inference:  $\mathbf{z}_i = f_i^{-1}(\mathbf{z}_{i-1})$

density:  $p(\mathbf{z}_i) = p(\mathbf{z}_{i-1}) \left| \det \frac{d\mathbf{z}_{i-1}}{d\mathbf{z}_i} \right|$

While training, we maximize the log likelihood:

$$\log p(\mathbf{x}) = \log p(\mathbf{z}_0) + \sum_{i=1}^K \log \left| \det \frac{d\mathbf{z}_{i-1}}{d\mathbf{z}_i} \right|$$

Making the **Jacobian determinant** easy to compute by choosing  $\frac{df_i^{-1}}{d\mathbf{z}_i}$  to be triangular matrix.

One step of flow in the Glow model goes through the layers:

- ▶ activation normalization;
- ▶ invertible  $1 \times 1$  convolutional;
- ▶ affine coupling.

Small building block of potentially big architectures.

Not as powerful as GAN, but cheap, easy to compute.

Motivation: Deep Learning has some disadvantages by itself.

- ▶ Heavily rely on massive labeled data;
- ▶ Uninterpretable;
- ▶ Hard to encode human intention and domain knowledge.

Human learning:

- ▶ Learn from concrete examples (similar to deep learning models)
- ▶ Learn from abstract knowledge (definitions, logic rules, etc)

Consider a statistical model  $\mathbf{x} \sim p_\theta(\mathbf{x})$ , it could be conditional model, generative model, discriminative model, etc.

Consider a constraint function  $f_\phi(\mathbf{x}) \in \mathbb{R}$ .

- ▶ The higher  $f_\phi(\mathbf{x})$  is, the better quality  $\mathbf{x}$  has w.r.t. knowledge.

Image example:

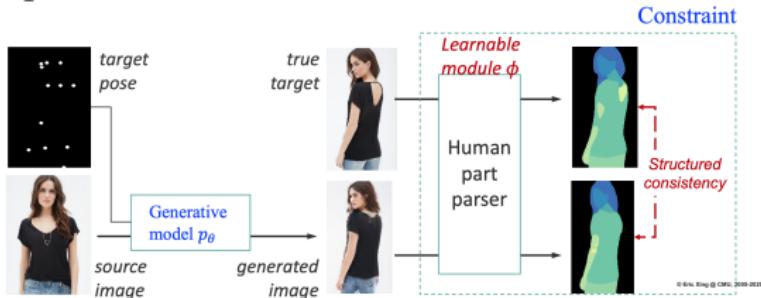


Figure: An example of using real pose as knowledge.

Sentiment classification example:

- ▶ “This was a terrific movie, but the director could have done better.”
- ▶ Logical Rules: Sentence S with structure A-but-B  $\Rightarrow$  sentiment of B dominates.

One way to impose the constraint is to maximize  $\mathbb{E}_{p_\theta}[f_\phi(\mathbf{x})]$ , which means, adding a regularization term to the objective:

$$\min_{\theta} \mathcal{L}(\theta) - \alpha \mathbb{E}_{p_\theta}[f_\phi(\mathbf{x})]$$

It is **difficult** to compute  $\mathbb{E}_{p_\theta}[f_\phi(\mathbf{x})]$ . Because when we compute the derivative  $\frac{d\mathbb{E}_{p_\theta}[f_\phi(\mathbf{x})]}{d\theta}$ , we use the log probability trick (always the case when deriving expectation over distribution). In the end we'll have a term that is the log probability itself (and something else) — that term will explode, high variance, extremely unstable. (Recall: Wake-Sleep's Sleep phase.)

A variational approximation<sup>19</sup> to ease the computation of  $\mathbb{E}_{p_\theta}[f_\phi(\mathbf{x})]$  is to use  $q(\mathbf{x})$  to approximate  $p_\theta(\mathbf{x})$ .

$$\mathcal{L}(\theta, q) = \text{KL}(q(\mathbf{x}) || p_\theta(\mathbf{x})) - \lambda \mathbb{E}_q[f_\phi(\mathbf{x})]$$

It introduces variational distribution  $q$ :

- ▶ Impose constraint  $f_\phi$  on  $q$ ;
- ▶ Encourage  $q$  to stay close to  $p_\theta$ .

The objective of data-driven and knowledge-driven combination:

$$\min_{\theta, q} \mathcal{L}(\theta) - \alpha \mathcal{L}(\theta, q)$$

<sup>19</sup>Called a Posterior Regularization [Ganchev et al., 2010].

$$\min_{\theta, q} \mathcal{L}(\theta) - \alpha \mathcal{L}(\theta, q)$$

$$\mathcal{L}(\theta, q) = \text{KL}(q(\mathbf{x}) || p_\theta(\mathbf{x})) - \lambda \mathbb{E}_q[f_\phi(\mathbf{x})]$$

One way to learn via EM algorithm:

- ▶ E-Step:  $q^*(\mathbf{x}) = p_\theta(\mathbf{x}) \exp\{\lambda_\phi(\mathbf{x})\}/Z$ 
  - ▶ This approach is known as a soft constraint. Higher value of  $\lambda_\phi$ , higher probability under  $q$ .
- ▶ M-Step:  $\min_\theta \mathcal{L}(\theta) - \mathbb{E}_{q^*}[\log p_\theta(\mathbf{x})]$

Consider a supervised learning:  $p_\theta(\mathbf{y}|\mathbf{x})$  and Input-Target space  $(\mathbf{X}, \mathbf{Y})$ , with first-order logic rules:  $(r, \lambda)$

- ▶  $r(\mathbf{X}, \mathbf{Y}) \in [0, 1]$  could be soft;
- ▶  $\lambda$  is the confidence level of the rule.

Given  $l$  rules:

- ▶ E-Step:  $q^*(\mathbf{y}|\mathbf{x}) = p_\theta(\mathbf{x}) \exp \left\{ \sum_l \lambda_l r_l(\mathbf{y}|\mathbf{x}) \right\} / Z$ 
  - ▶ Current version of  $p_\theta$  with all rule constraints.
- ▶ M-Step:  $\min_\theta \mathcal{L}(\theta) - \mathbb{E}_{q^*} [\log p_\theta(\mathbf{y}|\mathbf{x})]$

Similar efforts were made by Hinton: Knowledge Distillation.

Student network:  $p_{\theta}(\mathbf{y}|\mathbf{x})$  (difficult to learn)

- ▶ Typically contains only takes labeled data.

Teacher network:  $q^*(\mathbf{y}|\mathbf{x})$

- ▶ Auxiliary, variational approximation, etc.
- ▶ Designed to be ensemble, take labeled data, but could possibly take unlabeled data as well.

Match **soft** predictions (not just 0 or 1) of the teacher network and student network.

- ▶ Train the teachers in one step, train the student to imitate the outputs of teacher network in another step.
- ▶ Will eventually get student closer to some / one / average of the teachers.

<sup>21</sup>[Hinton et al., 2015; Bucilu et al., 2006]

Teacher network is rule-regularized. Recall the previous E-Step:

$$p^*(\mathbf{y}|\mathbf{x}) = p_\theta(\mathbf{x}) \exp \left\{ \sum_l \lambda_l r_l(\mathbf{y}|\mathbf{x}) \right\} / Z$$

The results from teacher network are soft, including both  $p_\theta$  and logic rules constraints:  $\mathbf{s}_n^{(t)}$  ( $n$  is the sample index,  $t$  is the current iteration).

There is also a ground truth label:  $\mathbf{y}_n$ . Student output is  $\sigma_\theta(\mathbf{x}_n)$ .

At iteration  $t$  ( $\pi \in [0, 1]$  is a balancing parameter):

$$\theta^{(t+1)} = \arg \min_{\theta \in \Theta} \frac{1}{N} \sum_{n=1}^N (1 - \pi) \ell(\mathbf{y}_n, \sigma_\theta(\mathbf{x}_n)) + \pi \ell(\mathbf{s}_n^{(t)}, \sigma_\theta(\mathbf{x}_n))$$

<sup>22</sup>[Hu et al., 2016]

More on learning rules / constraints:

- ▶ Teacher / student network structures.
- ▶ Learn the confidence value  $\lambda_l$  of each rule. [Hu et al., 2016b]
- ▶ More generally, optimize parameters of the constraint  $f_\phi(\mathbf{x})$ . [Hu et al., 2018]
- ▶ Teachers can reach beyond the scope of logical rules. Possible to make the reward function of reinforcement learning as a type of teaching function.
  - ▶ From this perspective, reinforcement learning becomes an instance of knowledge-driven machine learning.
  - ▶ See keyword: **variational reinforcement learning**.

## Generative Adversarial Networks (GANs)

- ▶ Wasserstein GAN: new learning objectives
- ▶ Progressive GAN: new training schedule
- ▶ BigGAN: scaling up GAN models

## Normalizing Flow (NF)

- ▶ Chained transformation functions
- ▶ Exact latent inference, density evaluation, sampling

## Integrating Domain Knowledge into Deep Learning

- ▶ Domain knowledge as constraint
- ▶ Learning rules / constraints