

## Regression: Caveats

Causal interpretation of regressions requires many assumptions

Threats to validity include:

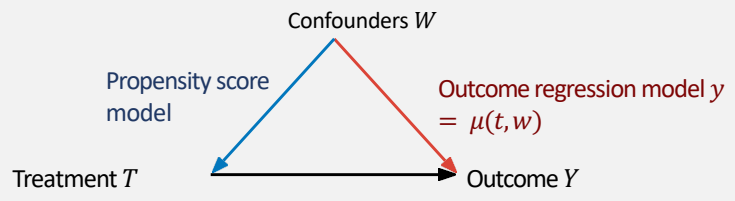
- **Modeling assumptions** : e.g., what if we use a linear model and causal relationship is non-linear
- **Multicollinearity**: if covariates are correlated, we can't get accurate coefficients
- **Omitted variables**: Omission of confounders can invalidate findings

## Doubly robust methods: Best of both worlds?

- Both propensity score weighting and regression models require correctly specified models
  - E.g., if propensity score or regression is modeled as a linear combination, but is non-linear, then it is not correctly specified
- Doubly robust methods combine “best of” propensity score and regression methods
- If either propensity score or regression is correctly specified, then doubly robust method yields unbiased estimates

## Augmented IPW or Doubly Robust Estimator

- Doubly Robust Estimator (DR) or Augmented IPW
- Combines the propensity score weighting with the outcome regression



J. Robins, A. Rotnitzky, and L. Zhao. "Estimation of regression coefficients when some regressors are not always observed." *Journal of the American statistical Association* 89.427 (1994): 846-866.

PennState  
Institute for Computational  
and Data Sciences

Center for Artificial Intelligence Foundations and Scientific Applications  
 Artificial Intelligence Research Laboratory

CTSI  
Clinical and Translational  
Science Institute

## Doubly robust methods revisited

```

      graph TD
        W((W)) --> T((T))
        W((W)) --> Y((Y))
        T((T)) --> Y((Y))
      
```

**Augmented IPW**

- Model both  $\mu(t, w)$  and  $e(w)$
- Consistent if either  $\hat{\mu}(t, w)$  or  $\hat{e}(w)$  is consistent
- Theoretically converges at a faster rate than COM or IPW

$$\hat{\tau} = \frac{1}{n} \sum_{i=1}^n \left( \hat{\mu}(t_i, w_i) + \left( \frac{t_i}{\hat{e}(w_i)} + \frac{(1-t_i)}{1-\hat{e}(w_i)} \right) (y_i - \hat{\mu}(t_i, w_i)) \right)$$

Outcome modeling

IPW weighting

Residual bias

PennState  
College of Engineering  
Science and Technology

Principles of Causal Inference

Vasant G Honavar

## Doubly Robust Models: Caveat

- If either propensity score or regression is correctly specified, then doubly robust model is unbiased.
- Seems like doubly robust method should be strictly better (less biased) than either propensity score weighting or regression
- But, if **both** propensity score or regression are **slightly** incorrect, then doubly robust estimator may become *very* biased

## Doubly Robust Methods

**Intuition** Combine propensity score weighting and regression models to provide unbiased estimate when either propensity score or regression is correctly specified

**Keep in mind** Fundamental assumptions (ignorability, etc.) must still hold.  
If both models are slightly incorrect, doubly robust estimator can be more biased

## Covariate balancing propensity score (CBPS)

- Since propensity score plays a dual role - the probability of being treated and covariate balancing score, CBPS estimates the propensity score by solving:

$$\mathbb{E}_{\beta} \left[ \frac{t_i f(w_i)}{e(w_i; \beta)} - \frac{(1-t_i) f(w_i)}{1-e(w_i; \beta)} \right] = 0$$

- where
  - $f(w)$  is a user-specified function of  $w$  e.g.,  $f(w) = w$  balances the distribution of covariates with respect to their first moments etc.
  - $e(w_i; \beta)$  is an estimate of propensity score parameterized by  $\beta$
  - CBPS is robust to mild mis-specification of the propensity score

K. Imai and M. Ratkovic. 2014 Covariate balancing propensity score. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 76(1):243–263

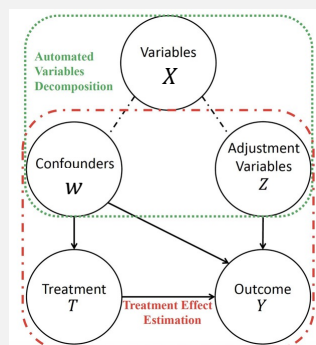
## Sample and Covariate Re-weighting

### Data-Driven Variable Decomposition (D<sup>2</sup>VD)

- **Assumption:** Observed variables can be decomposed into confounders ( $W$ ), adjustment variables ( $Z$ ) and the irrelevant variables
- D<sup>2</sup>VD distinguishes the confounders and adjustment variables, while eliminating the irrelevant variables.

$$\hat{\tau} = \mathbb{E} \left[ (Y - \varphi(z)) \frac{T - e(W)}{e(W)(1 - e(W))} \right]$$

- $\varphi(z)$  is a latent representation optimized to compensate for the effect of confounders on  $Y$



Kuang, Kun, et al. "Treatment effect estimation with data-driven variable decomposition." AAAI'17.



## Differentiated confounder balancing (DCB)

- DCB selects confounders and adjusts the weights of both confounders and samples to balance the distribution of samples in the treated and untreated groups
- Bypasses the estimation of propensity scores

Kuang, K., Cui, P., Li, B., Jiang, M., Wang, Y., Wu, F. and Yang, S., 2019. Treatment effect estimation via differentiated confounder balancing and regression. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 14(1), pp.1-25.

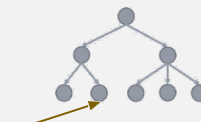
## Tree-based Methods

- Classification And Regression Trees (CART)
  - Recursively partition the data space based on GINI index or entropy
  - Fit a simple prediction model for each partition

Leaf specific effect:

$$\mu(w, x | \Pi) \equiv \mathbb{E} \left[ Y_i(w) \mid X_i \in \ell(x | \Pi) \right]$$

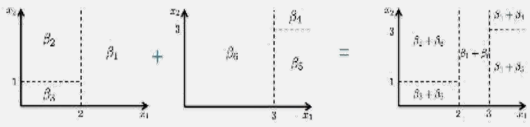
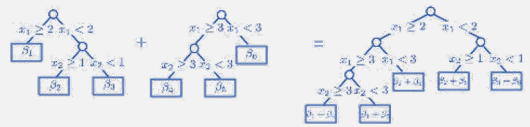
A specific leaf node



$$\tau(x | \Pi) \equiv \mu(1, x | \Pi) - \mu(0, x | \Pi)$$

## Tree-based Methods

- Bayesian Additive Regression Trees (BART)
  - A Bayesian ensemble of trees model
  - Nonparametric Bayesian regression



$$Y = g(x; T_1, M_1) + g(x; T_2, M_2) + \dots + g(x; T_m, M_m) + \varepsilon$$

$$\varepsilon \sim N(0, \sigma^2)$$

Hill, Jennifer L. "Bayesian nonparametric modeling for causal inference." *Journal of Computational and Graphical Statistics* 20.1 (2011): 217-240.

## Advantages of BART

- Easy to implement
- Little need for parameter tuning
- Posterior provides uncertainty of the estimation
- Can handle large number of covariates or confounders
- Can handle discrete as well as continuous treatment variables and missing data

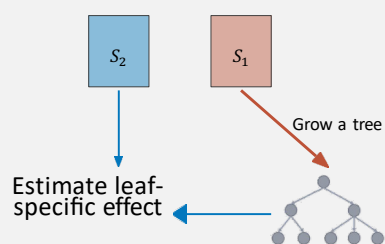
## Tree-based Methods: Causal Forests

- An ensemble of trees
- A variant of Breiman's random forest algorithm
- Trees and forests help find matches adaptively
- Extended to multiple treatments

S. Wager, and S. Athey. "Estimation and inference of heterogeneous treatment effects using random forests." *Journal of the American Statistical Association* 113.523 (2018): 1226-1242.

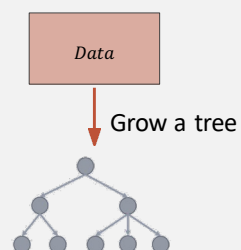
## Tree-based Methods: Causal Forests


- Estimate causal effect using two data sets
  - One data set used to build a tree
  - Second data set used to estimate leaf-specific causal effects
- Advantage:
  - Can estimate CATE
  - Consistency of CATE estimates
  - Nice Asymptotic properties



## Causal forests for propensity estimation

- Use the entire data set to grow the tree
- Estimate propensity at each leaf
- Ensembles possible
- Each leaf node at least  $k$  observations




 **PennState**  
Institute for Computational  
and Data Sciences

**Center for Artificial Intelligence Foundations and Scientific Applications**  
**Artificial Intelligence Research Laboratory**

**CTSI** Clinical and Translational  
Science Institute

# Deep learning for causal effect estimation

 **PennState**  
College of Information  
Science and Technology

Principles of Causal Inference

Vasant G Honavar

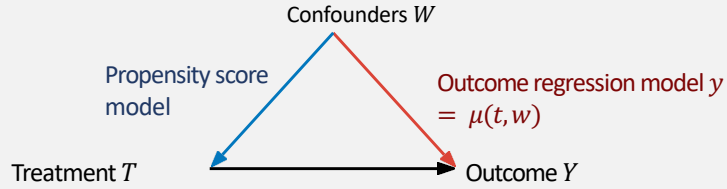


## Doubly robust methods revisited

- Both propensity score weighting and regression models require correctly specified models
  - E.g., if propensity score or regression is modeled as a linear combination, but is non-linear, then it is not correctly specified
- Doubly robust methods combine “best of” propensity score and regression methods
- If either propensity score or regression is correctly specified, then doubly robust method yields unbiased estimates

## Augmented IPW or Doubly Robust Estimator

- Doubly Robust Estimator (DR) or Augmented IPW
- Combines the propensity score weighting with the outcome regression



J. Robins, A. Rotnitzky, and L. Zhao. "Estimation of regression coefficients when some regressors are not always observed." *Journal of the American statistical Association* 89.427 (1994): 846-866.

**PennState**  
Institute for Computational  
and Data Sciences

**Center for Artificial Intelligence Foundations and Scientific Applications**  
Artificial Intelligence Research Laboratory

**CTSI**  
Clinical and Translational  
Science Institute

## Doubly robust methods revisited

**Augmented IPW**

- Model both  $\mu(t, w)$  and  $e(w)$
- Consistent if either  $\hat{\mu}(t, w)$  or  $\hat{e}(w)$  is consistent
- Theoretically converges at a faster rate than COM or IPW

$$\hat{t} = \frac{1}{n} \sum_{i=1}^n \left( \hat{\mu}(t_i, w_i) + \left( \frac{t_i}{\hat{e}(w_i)} + \frac{(1-t_i)}{1-\hat{e}(w_i)} \right) (y_i - \hat{\mu}(t_i, w_i)) \right)$$

↑

Outcome modeling

↑

IPW weighting

↑

Residual bias

```

graph TD
    W((W)) --> T((T))
    W((W)) --> Y((Y))
    T((T)) --> Y((Y))
        
```

- If either propensity score or regression is correctly specified, then doubly robust model is unbiased

**PennState**  
Center for Artificial Intelligence  
Foundations and Scientific Applications

Principles of Causal Inference

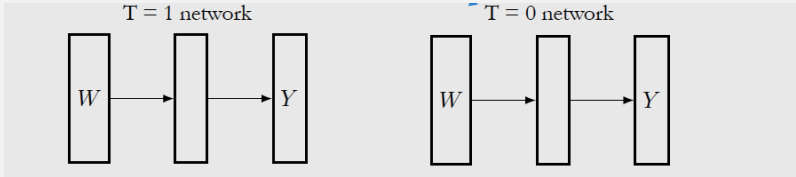
Vasant G Honavar

## Deep learning for causal effect estimation

- Deep outcome modeling
- Covariate balancing through representation learning
- Generative modeling
- Adversarial learning

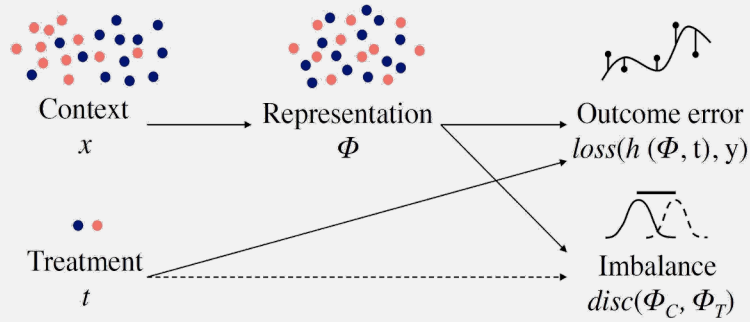
### Deep outcome modeling

- We train two deep neural networks
  - one to predict treated outcome
  - One to predict the control outcome



- After training, we use the trained model to predict outcomes  $\hat{Y}_i(1)$  and  $\hat{Y}_i(0)$  or (conditional outcomes)
- Estimate ATE or CATE by averaging the predicted outcomes (or conditional outcomes)

## Balanced representation learning

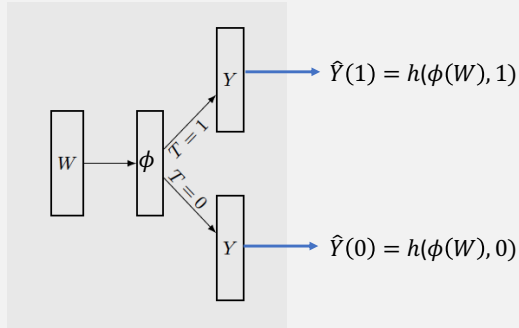


**Key idea:** A deep neural network or kernel function can map the covariates into a representation space where the treated and control distributions are nearly identical

<sup>1</sup> Shalit, U., Johansson, F.D. and Sontag, D., 2017, Estimating individual treatment effect: generalization bounds and algorithms. In *International Conference on Machine Learning* (pp. 3076-3085). PMLR.

## Balanced representation learning for causal effect estimation

- Target agnostic regression network (TARNet)



Intuition:

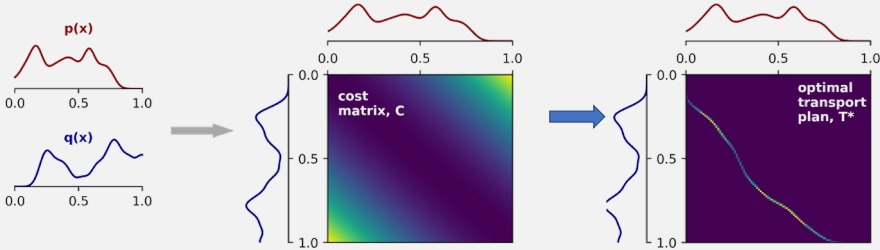
- $\phi$  is trained to simultaneously predict both  $\hat{Y}_i(1)$  and  $\hat{Y}_i(0)$  or their conditional counterparts
- Can we do better?

$$\operatorname{argmin}_{h, \phi} \frac{1}{n} \operatorname{MSE}(Y_i(T_i), h(\phi(w_i), T_i)) + \lambda R(h)$$

<sup>1</sup>Shalit, U., Johansson, F.D. and Sontag, D., 2017, Estimating individual treatment effect: generalization bounds and algorithms. In *International Conference on Machine Learning* (pp. 3076-3085). PMLR.

### Integral probability metrics for representation balancing

- Wasserstein distance or earthmover distance
- Measures the amount of probability mass that needs to be “transported” to transform one distribution into another



Wasserstein distance = cost of the optimal transport plan



## Integral probability metrics for representation balancing

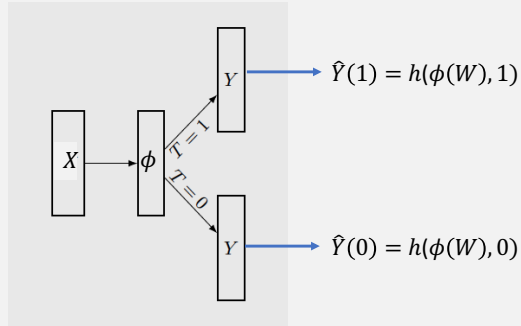
- Maximum mean discrepancy distance
- Normed distance between the means of two densities say,  $p$  and  $q$  after a kernel function  $\phi$  has mapped them to a representation space

$$MMD(p, q, F) = \sup_{f \in F} \|\mathbb{E}_p[f(x)] - \mathbb{E}_q[f(y)]\|^2$$

- When  $p = q$ ,  $MMD(p, q, F)$  is close to 0

## Representation balancing for causal effect estimation

- Counterfactual regression network (CFRNet)



- Intuition: Minimize an integral probability metric between the treated and untreated distributions in the representation space

$$\operatorname{argmin}_{h, \phi, IPM} \frac{1}{n} \operatorname{MSE}(Y_i(T_i), h(\phi(x_i), T_i)) + \alpha IPM(\phi(x|T=1), \phi(x|T=0)) + \lambda R(h)$$

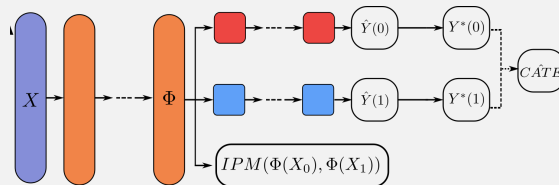
<sup>1</sup>Shalit, U., Johansson, F.D. and Sontag, D., 2017, Estimating individual treatment effect: generalization bounds and algorithms. In *International Conference on Machine Learning* (pp. 3076-3085). PMLR.

## Representation balancing for causal effect estimation

- Counterfactual regression network (CFRNet)

$$\operatorname{argmin}_{h, \phi, IPM} \frac{1}{n} \operatorname{MSE}(Y_i(T_i), h(\phi(x_i), T_i)) + \alpha IPM(\phi(X|T=1), \phi(x|T=0)) + \lambda R(h)$$

- The mean squared error of estimated causal effect is bounded by the sum of the factual loss, the counterfactual loss, and the variance of the (conditional) outcome
- The regularizer  $R(h)$ , typically,  $L_2$  penalty on the weights controls the complexity of the model for outcome prediction)

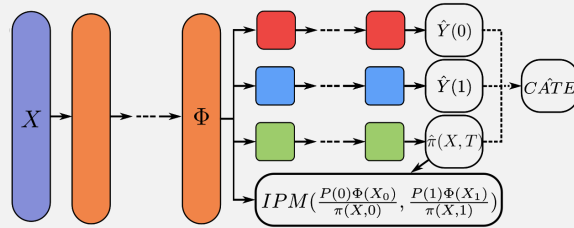


<sup>1</sup>Shalit, U., Johansson, F.D. and Sontag, D., 2017, Estimating individual treatment effect: generalization bounds and algorithms. In *International Conference on Machine Learning* (pp. 3076-3085). PMLR.

## Representation balancing for causal effect estimation

### Weighted CFRNet: CFRNet with consistency guarantees

- Incorporates inverse propensity weighting into CFRNet



$$\frac{\pi(X, T)}{P(T)} = e(X)$$

<sup>1</sup> Johansson, F.D., Shalit, U., Kallus, N. and Sontag, D., 2022. Generalization Bounds and Representation Learning for Estimation of Potential Outcomes and Causal Effects. *Journal of Machine Learning Research*, 23(166), pp.1-50.

## Representation balancing for causal effect estimation

### Weighted CFRNet: CFRNet with consistency guarantees

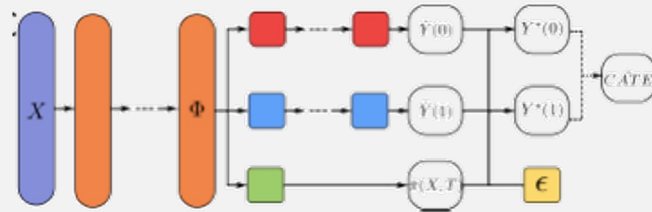
- Incorporates inverse propensity weighting into CFRNet

$$\begin{aligned}
 \arg \min_{h, \Phi, IPM, \pi, \lambda_h, \lambda_\pi} & \frac{1}{N} \sum_{i=1}^N \underbrace{\frac{\hat{P}(T_i)}{\pi(\Phi(X_i), T_i)}}_{IPW} \cdot \underbrace{MSE(Y_i, h(\Phi(X_i), T_i))}_{\text{Outcome Loss}} + \lambda_h \underbrace{\mathcal{R}(h)}_{L_2 \text{ Outcome}} + \\
 & \alpha \cdot \underbrace{IPM\left(\frac{\hat{P}(1)}{\pi(\Phi(X, 1))} \cdot \Phi(X|T=1), \frac{\hat{P}(0)}{\pi(\Phi(X, 0))} \cdot \Phi(X|T=0)\right)}_{\text{Distance between IPW weighted T \& C covar. distributions}} + \lambda_\pi \underbrace{\frac{\|\pi\|_2}{N}}_{L_2 \text{VAR}(\pi)}
 \end{aligned}$$

<sup>1</sup> Johansson, F.D., Shalit, U., Kallus, N. and Sontag, D., 2022. Generalization Bounds and Representation Learning for Estimation of Potential Outcomes and Causal Effects. *Journal of Machine Learning Research*, 23(166), pp.1-50.

## Representation balancing for causal effect estimation

Extensions with inverse propensity weighting: doubly robust estimates  
 DragonNet



- Instead of adding an IPM loss, we predict propensity scores and add a nudge parameter

$$\arg \min_{\Phi, \pi, h} \underbrace{MSE(Y, h(\Phi(X), T))}_{\text{Outcome Loss}} + \alpha \underbrace{BCE(T, \pi(\Phi(X), T))}_{\pi \text{ Loss}} + \lambda \underbrace{\mathcal{R}(h)}_{L_2}$$

<sup>1</sup>Shi, C., Blei, D.M., and Veitch, V., 2019, December. Adapting neural networks for the estimation of treatment effects. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems* (pp. 2507-2517)..

## Semiparametric Theory for Causal Effect Estimation

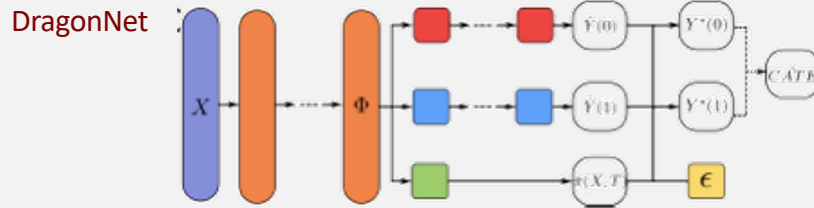
- Causal inference from observational data is about estimating a target parameter  $T(P) = ATE$  of distribution  $P$  of treatment effects
- Because we lack counterfactuals, we do not know the true distribution  $P$  of treatment effects
- We do know (can estimate) some constraints on  $P$ , e.g., the treatment mechanism
- We can encode these constraints into a likelihood that parametrically defines a family  $\mathcal{P}$  of approximate data informed distributions of  $P$
- Within  $\mathcal{P}$ , there exists a sample-inferred distribution  $\tilde{P}$  that we can use in place of  $P$  in  $T(P)$
- How to choose an optimal  $\tilde{P}$ ?
- We can use maximum likelihood estimate, but...

## Semiparametric Theory for Causal Effect Estimation

- How to choose an optimal  $\tilde{P}$ ?
- We can use maximum likelihood estimate, but...
- The likelihood may contain 'nuisance' terms that we do not care to estimate accurately
- MLE in such setting may yield a biased estimate of the target  $T(P)$  because it tries obtain accurate estimates of the nuisance terms as well
- We can sharpen the focus of the likelihood on  $T(P)$  using a nudge parameter  $\epsilon$  along an efficient influence curve (EIC)
- The resulting ATE estimate is asymptotically unbiased, efficient, and has confidence intervals with (asymptotically) correct coverage



## Representation balancing for causal effect estimation



- Semi parametric estimation yields

$$ATE = \frac{1}{N} \sum_{i=1}^N \left[ \underbrace{\left( \frac{T}{\pi(\mathbf{x}_i 1)} - \frac{1-T}{\pi(\mathbf{x}_i 0)} \right)}_{\text{Treatment Modeling}} \times \underbrace{(Y - h(\mathbf{x}_i T))}_{\text{Residual Confounding}} \right] + \underbrace{[h(\mathbf{x}_i 1) - h(\mathbf{x}_i 0)]}_{\text{Outcome Modeling}}$$

Adjustment

<sup>1</sup>Shi, C., Blei, D.M., and Veitch, V., 2019, December. Adapting neural networks for the estimation of treatment effects. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems* (pp. 2507-2517)..

## Targeted Maximum Likelihood Estimation (TMLE)

- TMLE iteratively uses a nuisance parameter  $\epsilon$  to nudge the outcome models towards sharper estimates of the ATE
  1. Fit  $h$  by predicting outcomes (e.g., using TARNet) and minimizing  $MSE(Y, h(X, T))$
  2. Fit  $\pi$  by predicting treatment (e.g., using logistic regression) and  $BCE(T, \pi(X, T))$
  3. Plug-in  $h$  and  $\pi$  functions to fit  $\epsilon$  and estimate  $h^*(X, T)$  where

$$\underbrace{h^*(X, T)}_{Y^*} = \underbrace{h(X, T)}_Y + \underbrace{\left( \frac{T}{\pi(X, T)} - \frac{1-T}{\pi(X, 1-T)} \right)}_{\text{Treatment Modeling Adjustment}} \times \underbrace{\epsilon}_{\text{Nudge}}$$

by minimizing  $MSE(Y, h^*(X, T))$ .

4. Plug-in  $h^*(X, T)$  to estimate ATE:  $ATE_{TMLE} = \frac{1}{N} \sum_{i=1}^N \underbrace{h^*(X_i, 1)}_{Y_i^*(1)} - \underbrace{h^*(X_i, 0)}_{Y_i^*(0)}$

## From TMLE to targeted regularization in DragonNet

- Targeted regularization adapts TMLE to a neural network loss function

1. (a) Use Dragonnet to predict  $h(\Phi(X), T)$  and  $\pi(\Phi(X), T)$ .

(b) Calculate the standard ML loss for the network using a hyperparameter  $\alpha$ :


$$\arg \min_{\Phi, \pi, h} \underbrace{MSE(Y, h(\Phi(X), T))}_{\text{Outcome Loss}} + \alpha \underbrace{BCE(T, \pi(\Phi(X), T))}_{\pi \text{ Loss}} + \lambda \underbrace{R_2(h)}_{L_2}$$

2. (a) Compute  $h^*(\Phi(X), T)$  as above


$$\underbrace{h^*(\Phi(X), T)}_{Y^*} = \underbrace{h(\Phi(X), T)}_F + \underbrace{\left( \frac{1-T}{\pi(\Phi(X), T)} - \frac{1-T}{\pi(\Phi(X), 1-T)} \right)}_{\text{Treatment Modeling Adjustment}} \times \underbrace{e}_{\text{"nudges"}}$$

(b) Calculate regularization loss:  $MSE(Y, h^*(\Phi(X), T))$


3. Combine and minimize the losses from 1 and 2 using a hyperparameter  $\beta$

 PennState  
Institute for Computational  
and Data Sciences

Center for Artificial Intelligence Foundations and Scientific Applications  
Artificial Intelligence Research Laboratory

 CTSI  
Clinical and Translational  
Science Institute

# Adversarial training of representations for causal effect estimation

 PennState  
College of Information  
Science and Technology

Principles of Causal Inference

Vasant G Honavar

PennState Institute for Computational and Data Sciences | Center for Artificial Intelligence Foundations and Scientific Applications Artificial Intelligence Research Laboratory | CTSI Clinical and Translational Science Institute

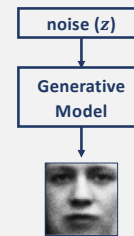
# Adversarial Networks

The diagram illustrates the architecture of an Adversarial Network. It features three main components: a **Generative Model**, a **Real world** source, and a **Discriminative Model**. The **Generative Model** (blue box) outputs a synthetic face image. The **Real world** (red box) provides a natural face image. Both images are fed into the **Discriminative Model** (black box). The Discriminative Model's task is to determine if the input is 'real or fake?', as indicated by the output box above it. A dashed arrow shows the flow of data from both image sources into the Discriminative Model.

PennState University of Artificial Intelligence, Science and Technology | Principles of Causal Inference | Vasant G Honavar

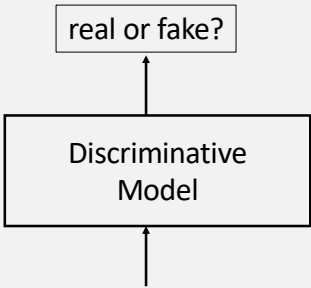
## Generative Model

- How to generate different samples from the model?
  - Input random noise
- Generative model as a neural network
  - computes  $x = G(z|\theta)$
  - differentiable
  - does not have to be invertible
  - $z$  typically is very high dimensional (higher than  $x$ )



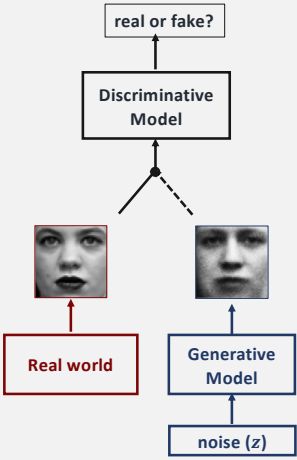
## Discriminative Model

- Discriminative model is a critic
  - A good critic can tell real from fake data
- Discriminative model as a neural net
  - differentiable
  - computes  $D(x)$ , with value 1 if real, 0 if fake



## Training Procedure: Basic Idea

- $G$  tries to fool  $D$
- $D$  tries not to be fooled
- Models are trained simultaneously
  - As  $G$  gets better,  $D$  has a more challenging task
  - As  $D$  gets better,  $G$  has a more challenging task
- Ultimately, we don't care about the  $D$ 
  - Its role is to force  $G$  to work harder





**PennState**  
Institute for Computational  
and Data Sciences

Center for Artificial Intelligence Foundations and Scientific Applications  
Artificial Intelligence Research Laboratory

**CTSI**  
Clinical and Translational  
Science Institute

## Loss Functions

- Loss function for D
  - Maximize the likelihood that model says 'real' to samples from the world and 'fake' to generated samples
  - $\mathcal{L}_D = -\mathbb{E}_{x \sim \text{world}} \ln D(x) - \mathbb{E}_z \ln (1 - D(G(z)))$
- What should the loss function be for G?
  - $\mathcal{L}_G = -\mathcal{L}_D$
- But because first term doesn't matter for G (why?)
  - $\mathcal{L}_G = \mathbb{E}_z \ln (1 - D(G(z)))$
- Known as a **minimax procedure**

**PennState**  
Institute for Computational  
and Data Sciences

Principles of Causal Inference

Vasant G Honavar

## Loss function: Discriminator

- **Discriminator** needs to:
  - Correctly classify **real** data:

$$\max_D \mathbb{E}_{x \sim p_{data}(x)} [\log D(x)] \quad D(x) \rightarrow 1$$

- Correctly classify **wrong** data:

$$\max_D \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))] \quad D(G(z)) \rightarrow 0$$

- The discriminator is an **adaptive loss function**.

## Loss function: Generator

- **Generator** needs to **fool** the discriminator:

- Generate samples similar to the real ones:

$$\min_G \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))] \quad D(G(z)) \rightarrow 1$$

- Non saturating objective (Goodfellow et al., 2014):

$$\min_G \mathbb{E}_{z \sim p_z(z)} [-\log(D(G(z)))]$$

**PennState**  
Institute for Computational  
and Data Sciences

Center for Artificial Intelligence Foundations and Scientific Applications  
Artificial Intelligence Research Laboratory

**CTSI**  
Clinical and Translational  
Science Institute

## Training Procedure

- Train both models simultaneously via stochastic gradient descent using mini-batches consisting of
  - some generated samples
  - some real-world samples
- Training of  $D$  is straightforward
- Error for  $G$  comes via back propagation through  $D$ 
  - Freeze  $D$  weights and propagate  $\mathcal{L}_G$  through  $D$  to determine  $\partial \mathcal{L}_G / \partial x$
- $D$  can be trained without altering  $G$ , and vice versa
  - May want multiple training epochs of just  $D$  so it can stay ahead
  - May want multiple training epochs of just  $G$  because it has a harder task

```

graph BT
    noise[noise (z)] --> GM[Generative Model]
    GM --> gen_img[Generated Image]
    real_world[Real world] --> DM[Discriminative Model]
    gen_img --> DM
    DM --> output[real or fake?]
  
```

**PennState**  
Institute for Computational  
and Data Sciences

Principles of Causal Inference

Vasant G Honavar

## Generative adversarial networks for ITE estimation


### Idea:

- Factual outcome  $\rightarrow$  observed labels
- Counterfactual outcome  $\rightarrow$  missing labels
- Missing labels generated using GANs


### Approach: A combination of two GANs

- Counterfactual GAN:
  - Input: the data with missing labels
  - Goal: estimate the counterfactual outcome
  - Output: the complete data
- ITE GAN
  - Input: the complete data from counterfactual block
  - Output: causal effect estimate

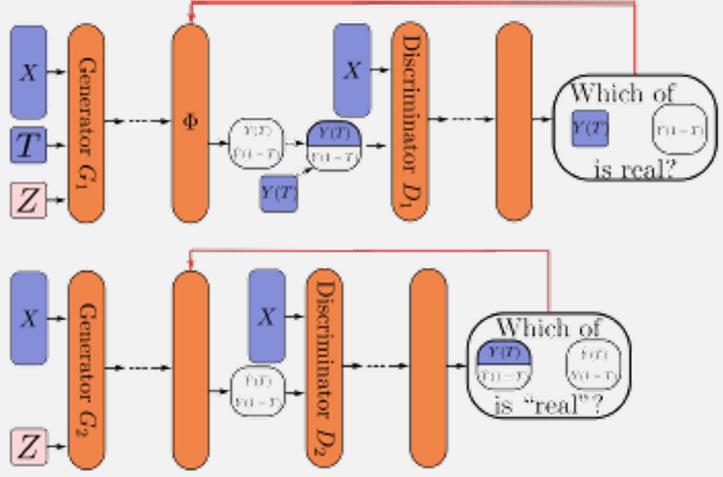
Yoon, Jinsung, James Jordan, and Mihaela van der Schaar. "GANITE: Estimation of individualized treatment effects using generative adversarial nets.

 PennState  
 Institute for Computational and Data Sciences

Center for Artificial Intelligence Foundations and Scientific Applications  
 Artificial Intelligence Research Laboratory

 CTSI  
 Clinical and Translational Science Institute

## Generative adversarial networks for ITE estimation (GANITE)



The diagram illustrates the GANITE architecture, which consists of two parallel generative adversarial networks (GANs) used for ITE estimation.


**Top GAN (Generator  $G_1$  and Discriminator  $D_1$ ):**

- Generator  $G_1$ :** Takes inputs  $X$  (blue box) and  $T$  (blue box) and produces a distribution  $\Phi$  (orange vertical bar).
- Discriminator  $D_1$ :** Takes inputs  $X$  (blue box) and  $Z$  (pink box) and produces a distribution  $\Psi$  (orange vertical bar).
- Decision:** A box asks "Which of  $Y(T)$  (blue box) or  $Y(1-T)$  (pink box) is real?".

**Bottom GAN (Generator  $G_2$  and Discriminator  $D_2$ ):**

- Generator  $G_2$ :** Takes inputs  $X$  (blue box) and  $Z$  (pink box) and produces a distribution  $\Phi$  (orange vertical bar).
- Discriminator  $D_2$ :** Takes inputs  $X$  (blue box) and  $Z$  (pink box) and produces a distribution  $\Psi$  (orange vertical bar).
- Decision:** A box asks "Which of  $Y(T)$  (blue box) or  $Y(1-T)$  (pink box) is 'real'?".

Red arrows indicate the flow of information from the distributions  $\Phi$  and  $\Psi$  to the decision boxes.

 PennState  
 Center for Artificial Intelligence Foundations and Scientific Applications  
 Artificial Intelligence Research Laboratory

Principles of Causal Inference

Vasant G Honavar

## Counterfactual GAN

1. Taking  $X, T$ , and generative noise  $Z$  as input, generator  $G_1$  generates both potential outcomes  $\{Y(T), \hat{Y}(1-T)\}$ . A factual loss  $MSE(Y(T), \hat{Y}(T))$  is applied.
2. Create a new vector  $C = \{Y(T), \hat{Y}(1-T)\}$  by combining the observed potential outcome and the counterfactual predicted by  $G_1$ .
3. Taking  $X$  and  $C$  as inputs, the discriminator rates each value in  $C$  for the probability that it is the observed outcome using the categorical cross entropy:

$$\mathcal{L}(D) = CCE(\underbrace{\{P(C_0 = Y(T))\}}_{\text{Prob first idx is real}}, \underbrace{\{P(C_1 = Y(T))\}}_{\text{Prob sec idx is real}}, \underbrace{\{C_0 = Y(T)\}}_{\text{1 if idx 0 is real}}, \underbrace{\{C_1 = Y(T)\}}_{\text{1 if idx 1 is real}})$$

4. This loss is then fed back to  $G_1$  such that the total loss for the generator is now

$$\arg \min_{G_1} = MSE(Y(T), \hat{Y}(T)) - \lambda \mathcal{L}(D_1)$$

## ITE GAN

1. Taking only  $X$  and generative noise  $Z$  as input,  $G_2$  generates a new potential outcome vector  $\mathbf{R} = \{\hat{Y}(T), \hat{Y}(1-T)\}$ .  $G_2$  receives an MSE loss to minimize the difference between its predictions and the "complete dataset"  $\mathbf{C}$ :  $MSE(\mathbf{C}, \mathbf{R})$ .
2. Discriminator  $D_2$  takes  $X$ ,  $\mathbf{C}$ , and  $\mathbf{R}$  as inputs and estimates a probability that  $\mathbf{C}$  is the "complete" dataset, and that  $\mathbf{R}$  is the "complete dataset":

$$\mathcal{L}(D) = CCE(\{ \underbrace{P(\mathbf{C} = \mathbf{C})}_{\text{Prob C is "CD"}}, \underbrace{P(\mathbf{R} = \mathbf{C})}_{\text{Prob R is "CD"}} \}, \{ \underbrace{\mathbf{C} = \mathbf{C}}_{\substack{1 \text{ if idx } 0 \text{ is } \mathbf{C} \\ 0 \text{ if idx } 1 \text{ is } \mathbf{C}}}, \underbrace{\mathbf{C}_1 = Y(T)}_{\substack{1 \text{ if idx } 1 \text{ is } \mathbf{C} \\ 0 \text{ if idx } 0 \text{ is } \mathbf{C}}} \})$$

3. This loss is then fed back to the generator  $G_2$  such that the total loss for the generator is now

$$\arg \min_{G_2} = MSE(\mathbf{C}, \mathbf{R}) - \lambda \mathcal{L}(D_2)$$



PennState Institute for Computational and Data Sciences | Center for Artificial Intelligence Foundations and Scientific Applications Artificial Intelligence Research Laboratory | CTSI Clinical and Translational Science Institute

## Variational autoencoders for Causal Effect Estimation

- **CEVAE**
  - Assumption: many **proxies** of the hidden confounder(s) are available
  - Estimate of a latent-variable model using VAE
    - Discover the hidden confounders
    - Infer how the hidden confounders affect the treatment and outcome
  - Advantages:
    - Weaker assumptions about
      - the data generating process and
      - the structure of the hidden confounders

```
graph BT; z((z)) --> x((x)); z --> y((y)); z --> t((t)); t --> y
```

## Account for hidden confounders

- CEVAE : Relaxing the unconfoundedness assumption.

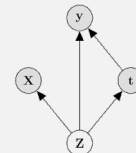


Figure 1

**Theorem 1.** If we recover  $p(\mathbf{Z}, \mathbf{X}, \mathbf{t}, \mathbf{y})$  then we recover the ITE under the causal model in Figure 1.

$$ITE(x) := \mathbb{E}[y|\mathbf{X} = x, do(\mathbf{t} = 1)] - \mathbb{E}[y|\mathbf{X} = x, do(\mathbf{t} = 0)]$$

$$p(\mathbf{y}|\mathbf{X}, do(\mathbf{t} = 1)) = \int_{\mathbf{Z}} p(\mathbf{y}|\mathbf{X}, do(\mathbf{t} = 1), \mathbf{Z}) p(\mathbf{Z}|\mathbf{X}, do(\mathbf{t} = 1)) d\mathbf{Z} \stackrel{(i)}{=} \int_{\mathbf{Z}} p(\mathbf{y}|\mathbf{X}, \mathbf{t} = 1, \mathbf{Z}) p(\mathbf{Z}|\mathbf{X}) d\mathbf{Z},$$

Note: (i) denote applications of the rules of do-calculus

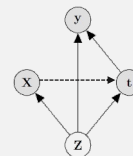
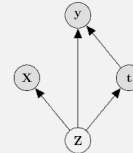
Louizos, Christos, et al. "Causal effect inference with deep latent-variable models." *Advances in Neural Information Processing Systems*. 2017.

## CEVAE : Relaxing the unconfoundedness assumption

$$ITE(x) := \mathbb{E}[y|X = x, do(t = 1)] - \mathbb{E}[y|X = x, do(t = 0)]$$

$$p(y|X, do(t = 1)) = \int_{\mathbf{Z}} p(y|X, do(t = 1), \mathbf{Z}) p(\mathbf{Z}|X, do(t = 1)) d\mathbf{Z} \stackrel{(i)}{=} \int_{\mathbf{Z}} p(y|X, t = 1, \mathbf{Z}) p(\mathbf{Z}|X) d\mathbf{Z},$$

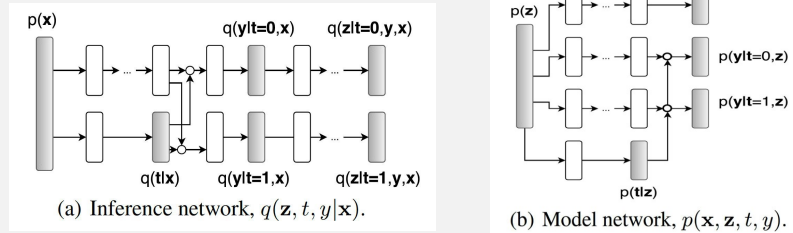
- How would the above estimate change if there is an edge from  $X$  to  $t$ ?
- **It would not**
- **Why?**
- **Because we intervene on  $t$ !**



[1] Louizos, Christos, et al. "Causal effect inference with deep latent-variable models." *Advances in Neural Information Processing Systems*. 2017.


## Variational Autoencoder for Causal Effect Estimation

CEVAE:




- White nodes: deterministic neural network transitions
- Gray nodes: sampling from the corresponding distribution
- Circles: switching based on the value of  $t$
- **Approach:** Use variational autoencoders to infer the complex non-linear relationships between  $X$  and  $(Z, t, y)$  and approximately recover  $p(Z, X, t, y)$


Louizos, Christos, et al. "Causal effect inference with deep latent-variable models." *Advances in Neural Information Processing Systems*. 2017.

 PennState  
Institute for Computational  
and Data Sciences

Center for Artificial Intelligence Foundations and Scientific Applications  
Artificial Intelligence Research Laboratory

 CTSI  
Clinical and Translational  
Science Institute

# Detour: Variational Autoencoders

 PennState  
College of Information  
Science and Technology

Principles of Causal Inference

Vasant G Honavar

PennState Institute for Computational and Data Sciences | Center for Artificial Intelligence Foundations and Scientific Applications Artificial Intelligence Research Laboratory | CTSI Clinical and Translational Science Institute

## The autoencoder (AE)

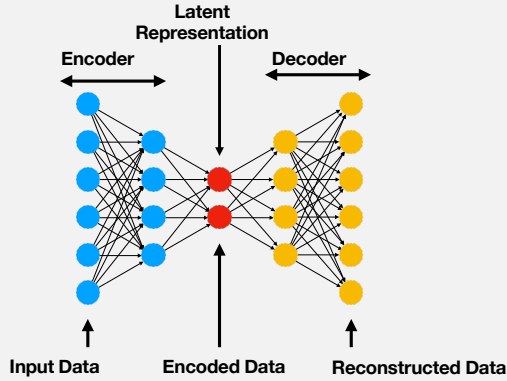
- Learn a low-dimensional representation of high-dimensional data
- AE consists of an encoder followed by a decoder

The diagram illustrates the autoencoder process. It starts with 'original data'  $X$  in a 'high-dimensional vector space', represented by a tall green vertical bar. An arrow labeled 'Encoder' points to a smaller orange box labeled 'compressed representation'  $h(x)$  in a 'low-dimensional "latent" space'. A second arrow labeled 'Decoder' points from  $h(x)$  to a tall green vertical bar labeled 'reconstructed data'  $\hat{X}$  in an 'original high-dimensional vector space'.

PennState | Principles of Causal Inference | Vasant G Honavar

# AE Architecture

- The encoder and decoder are usually neural networks
- Layers are often fully connected or convolutional (for image or time series data)

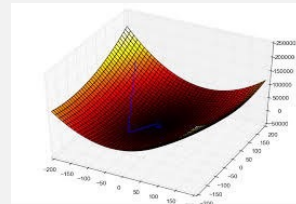


## AE learning objective

- An AE learns by minimizing the objective function

$$L = \frac{1}{N} \sum d(\mathbf{x}, \hat{\mathbf{x}})$$

- Over our set of training data where  $d$  is a “distance function”, typically squared error, and  $N$  is the number of training samples
- Update weights in the encoder  $E$  and decoder  $D$  via gradient descent





PennState  
Institute for Computational and Data Sciences

**Center for Artificial Intelligence Foundations and Scientific Applications**  
 Artificial Intelligence Research Laboratory

CTSI  
Clinical and Translational Science Institute

## From Autoencoder (AE) to Variational Autoencoder

Map a data set into a distribution

Encoder
Sample
Decoder

Input Data
Mean, Variance
Encoded Data
Reconstructed Data

PennState  
College of Engineering, Science and Technology

Principles of Causal Inference

Vasant G Honavar

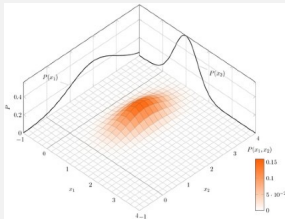
## Variational Autoencoder (VAE)

- Key idea: make both the encoder and the decoder probabilistic
- i.e., the latent variables,  $z$ , are drawn from a probability distribution depending on the input,  $X$ , and the reconstruction is chosen probabilistically from  $z$



## Modeling a data distribution

- Given data in the form of some vectors  $\mathbf{x} \in D$  in a vector space  $V$ , find a probability distribution  $p(\mathbf{x})$  over  $V$ , “peaked” only on the data  $D$



- Choose a formula  $p_{\theta}(\mathbf{x})$  for this distribution with parameters  $\theta$ , and maximize

$$\frac{1}{N} \log \prod_{\mathbf{x} \in D} p_{\theta}(\mathbf{x}) = \frac{1}{N} \sum_{\mathbf{x} \in D} \log p_{\theta}(\mathbf{x})$$

## Low-dimensionality of data

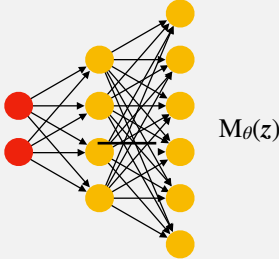
- Data usually lies on a low-dimensional submanifold of some high-dimensional ambient vector space
- We map data to the lower-dimensional subspace of interest

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

- where  $\mathbf{z}$  is a “latent variable” from the subspace. For a VAE, we assume that
  - $p_{\theta}(\mathbf{x} | \mathbf{z}) \sim$  normal with mean  $M(\mathbf{z})$  and variance  $\Sigma(\mathbf{z})^2$
  - $p(\mathbf{z}) \sim$  standard normal

### The decoder

- The VAE “decoder” is the map  $z \mapsto (M_\theta(z), \Sigma_\theta(z)^2)$



- $M_\theta(z)$  and  $\Sigma_\theta(z)^2$  are realized by neural networks

PennState  
Institute for Computational  
and Data Sciences

Center for Artificial Intelligence Foundations and Scientific Applications  
Artificial Intelligence Research Laboratory

CTSI  
Clinical and Translational  
Science Institute

## A difficulty

- Unfortunately, it is difficult to estimate  $p_{\theta}(x)$  from its integral
 
$$p_{\theta}(x) = \int p_{\theta}(x|z)p(z)dz \approx \sum_{z_i \sim p(z)}^M p_{\theta}(x|z_i)$$

small for most  $z_i \sim p(z)$ , so we need a huge  $M$  to estimate this.
- An easier integral would be
 
$$p_{\theta}(x) = \int p_{\theta}(x|z)q(z|x)dz \approx \sum_{z_i \sim q(z|x)}^M p_{\theta}(x|z_i)$$

large for most  $z_i \sim q(z|x)$ , so we don't need large  $N$  to estimate this.

where the distributions  $q(x|z)$  and  $p_{\theta}(x|z)$  are nearly identical for each  $z$

PennState  
Institute for Computational  
and Data Sciences

Principles of Causal Inference

Vasant G Honavar

## The fix

- We choose a formula for  $q(z | x)$  that depends on parameters  $\phi$
- For a VAE, we assume that

$$q_\phi(z | x) \sim \text{normal with mean } \mu_\phi(x) \text{ and variance } \sigma_\phi(x)^2$$

and tune the parameters  $\phi$  so  $q_\phi(z | x)$  and  $p_\theta(z | x)$  are almost identical

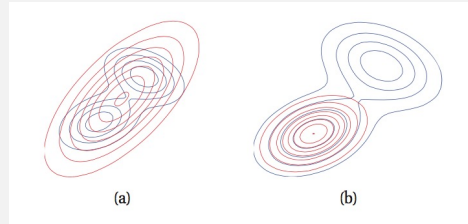
- By identical, we mean that the “KL-divergence”

$$\text{KL}(q_\phi(\cdot | x) || p_\theta(\cdot | x)) := \int q_\phi(z | x) \log \left( \frac{q_\phi(z | x)}{p_\theta(z | x)} \right) dz,$$

between these distributions, is small

## Variational Inference

- Use a simple distribution to approximate a complex distribution

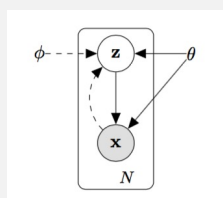


- Variational parameter:
  - Gaussian distribution:  $\mu, \sigma$
  - Gaussian mixture:  $[\mu], [\sigma], [w]$



## Theory: Variational Inference

- $X$ : data
- $Z$ : latent variable (hidden layer value)
- $\phi$ : Inference network parameter (encoder:  $q_{\phi}(z|x)$ )
- $\theta$ : generative network parameter (decoder:  $p_{\theta}(x|z)$ )



## Theory: Variational Inference

- Posterior distribution:

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

Intractable posterior!

- Goal: use variational posterior  $q_{\phi}(\mathbf{z}|\mathbf{x})$  to approximate true posterior  $p_{\theta}(\mathbf{z}|\mathbf{x})$

## Theory: Variational Inference

- Minimize KL-divergence between the variational posterior and true posterior

$$\begin{aligned}
 D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})) &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} \left[ \log \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z}|\mathbf{x})} \right] \\
 &= \log p_\theta(\mathbf{x}) - \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} \left[ \log \frac{p_\theta(\mathbf{z}, \mathbf{x})}{q_\phi(\mathbf{z}|\mathbf{x})} \right]
 \end{aligned}$$

- $\log p_\theta(\mathbf{x})$  is constant
- Minimizing  $D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x}))$  is same as maximizing  $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} \left[ \log \frac{p_\theta(\mathbf{z}, \mathbf{x})}{q_\phi(\mathbf{z}|\mathbf{x})} \right]$
- KL-divergence is non-negative

$$\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} \left[ \log \frac{p_\theta(\mathbf{x}, \mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} \right]$$

Variational lower bound of data likelihood

### Variational Lower Bound of data likelihood (ELBO)

$$\begin{aligned} \mathcal{L}_{VAE} &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} \left[ \log \frac{p_\theta(\mathbf{z}, \mathbf{x})}{q_\phi(\mathbf{z}|\mathbf{x})} \right] \\ &= \underbrace{-D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}))}_{\text{Regularization term}} + \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]}_{\text{Reconstruction term}} \end{aligned}$$

## The Reparameterization Trick

- Problem with respect to the VLB: updating  $\phi$

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

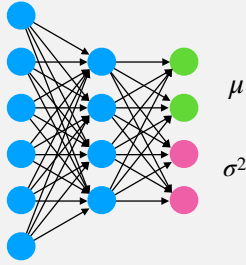
- $z \sim q_{\phi}(z|x)$  : need to differentiate through the sampling process w.r.t  $\phi$

## The Reparameterization Trick

- Solution: make the encoder deterministic
- Gaussian distribution example:
  - Previously:  $z \sim N(\mu, \sigma)$
  - Now  $z = \mu + \epsilon * \sigma, \epsilon \sim N(0, 1)$

### The encoder

- The VAE “encoder” is the map  $x \mapsto (\mu_{\phi}(x), \sigma_{\phi}(x)^2)$



- $\mu_{\phi}(x)$  and  $\sigma_{\phi}(x)^2$  are realized by neural networks

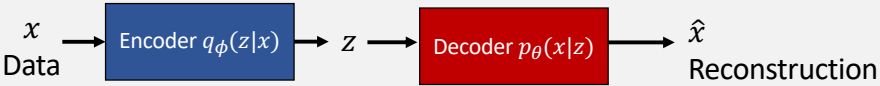
## VAE Encoder

- The encoder takes input and returns parameters for a probability density (e.g., Gaussian): i.e.  $q_\phi(z|x)$  gives the mean and co-variance matrix
- We can sample from this distribution to get random values of the lower-dimensional representation  $z$
- Implemented via a neural network: each input  $x$  gives a vector mean and diagonal covariance matrix that determine the Gaussian density  $q_\phi(z|x)$
- Parameters  $\phi$  for the neural network are learned



### VAE Decoder

- The decoder takes latent variable  $z$  and returns parameters for a distribution. E.g.,  $p_{\theta}(z|x)$  specifies the mean and variance for each element of the output
- Reconstruction  $\hat{x}$  is produced by sampling.
- Implemented via neural network, the neural network parameters  $\theta$  are learned.



## VAE generative model

- After training,  $q_\phi(z|x_i)$  is close to a standard normal,  $\mathcal{N}(0,1)$  – easy to sample.
- Using a sample of  $z$  from  $q_\phi(z|x_i)$  as input to sample from  $p_\theta(x|z)$  gives an approximate reconstruction of  $x_i$ , at least in expectation.
- If we sample any  $z$  from  $\mathcal{N}(0,1)$  and use it as input to to sample from  $p_\theta(x|z)$  then we can approximate the entire data distribution  $p(x)$ . i.e., we can generate new samples that look like the input but aren't in the input