

# Density Ratio Estimators in Variational Bayesian Machine Learning

Lammy

Department of Mathematics and Statistics  
UNSW

Statistics Honours, 2018

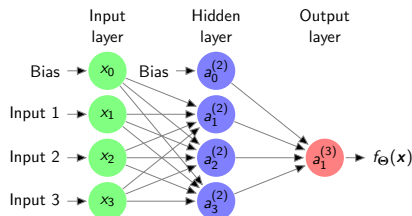
- 1 Background Info
  - Neural Networks
  - (Amortized) Variational Inference
  - Density Ratio Estimation
- 2 Early Experimentation
  - Initial Inference Experiment
  - Inference Experiment - Activation Function
- 3 Theory Break
- 4 Experiments
  - Inference Experiment
- 5 Further Estimator Loss Function Analysis
- 6 Generation Experiment

- 1 Background Info
  - Neural Networks
  - (Amortized) Variational Inference
  - Density Ratio Estimation
- 2 Early Experimentation
  - Initial Inference Experiment
  - Inference Experiment - Activation Function
- 3 Theory Break
- 4 Experiments
  - Inference Experiment
- 5 Further Estimator Loss Function Analysis
- 6 Generation Experiment

# Neural Networks

## Overall Structure

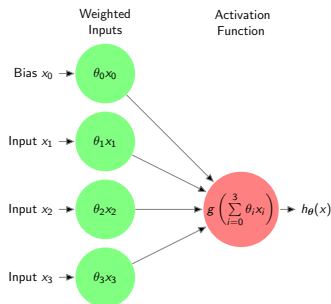
- Mathematical model based off human brain.
- Letting  $f^*$  be some function in  $\mathbb{R}$ , goal of neural network is to approximate  $f^*$  using a mapping with parameters  $\Theta$  from input  $\mathbf{x}$  to output  $\mathbf{y}$ :  $\mathbf{y} = \mathbf{f}_{\Theta}(\mathbf{x})$ .
- Universal Approximation Theorem states a neural network can approximate any function if it is complex enough.
- Consists of layers of nodes:



# Neural Networks

## Individual Node Structure

- Each node is a generalised linear model of preceding layer output.
- Weights are randomly initialised from normal or uniform distribution.



# Neural Networks

## Activation Functions

- Used to map node output to certain space.
- Input layer has no activation function (Linear).
- Hidden layer typically has Rectified Linear Unit (ReLU) activation function:  $g(x) = \max\{0, x\}$ .
- Choice of activation function for output layer depends on space of function being approximated:
  - Linear activation function outputs in  $\mathbb{R}$ .
  - ReLU activation function in  $[0, \infty)$  ideal for non-negative quantities e.g. time or price.
  - Sigmoid activation function  $g(x) = (1 + \exp(-x))^{-1}$  in  $(0, 1)$  ideal for probabilities e.g. classification.

- Weights and biases trained such that loss function is minimized:
  - Mean Squared Error:  $\min_{\Theta} \frac{1}{2} \|\mathbf{y} - \mathbf{f}_{\Theta}(\mathbf{x})\|_2^2$  suited for regression.
  - Bernoulli Loss:  $\min_{\Theta} -y \log(\mathbf{f}_{\Theta}(x)) - (1 - y) \log(1 - \mathbf{f}_{\Theta}(x))$  suited for classification.
- Back-propagation finds partial derivatives of loss function with respect to weights and biases by propagating error backwards through network.
- Gradient descent uses these partial derivatives to optimize network.
- In our programs we use 'Adam' optimizer, a gradient descent algorithm which adjusts training rate based on moments of convergence rate.

# Outline

## 1 Background Info

- Neural Networks
- (Amortized) Variational Inference
- Density Ratio Estimation

## 2 Early Experimentation

- Initial Inference Experiment
- Inference Experiment - Activation Function

## 3 Theory Break

## 4 Experiments

- Inference Experiment

## 5 Further Estimator Loss Function Analysis

## 6 Generation Experiment



# (Amortized) Variational Inference

## Bayesian Inference

- We recap the fundamental problem in Bayesian statistics, which is to evaluate, or estimate posterior densities  $p(z|x)$  so that we may perform inference on unknown parameters.

$$p(z|x) = \frac{p(z, x)}{p(x)} = \frac{p(z)p(x|z)}{\int_{\mathcal{Z}} p(z, x) dz}$$

- Problems arise when  $\int_{\mathcal{Z}} p(z, x) dz$  is computationally intractable.
- Typical MCMC methods resolve this problem by sampling from Markov chain that converges to the stationary distribution  $p(z, x)$ .
- This is extremely slow when dealing with large datasets or high dimensional data.
- Variational Inference is a solution.

# (Amortized) Variational Inference

## Introduction

- Amortized variational inference approximates  $p(z|x)$  with a different density  $q(z|x)$ , often represented as a neural network with parameters  $\phi$ .
- Minimize (reverse) KL Divergence between the two distributions. Since  $p(z|x)$  changes with different  $x$ , take expectation with respect to dataset  $q^*(x)$ :

$$q_{\phi}^*(z|x) = \arg \min_{q(z|x) \in \mathcal{Q}} \mathbb{E}_{q^*(x)} [KL(q_{\phi}(z|x) || p(z|x))].$$

- Reverse KL Divergence is the expected logarithmic difference between two distributions P and Q with respect to Q:

$$KL(q(z|x) || p(z|x)) = \mathbb{E}_{q(z|x)} \left[ \log \left( \frac{q(z|x)}{p(z|x)} \right) \right]$$

- Problem: We cannot calculate  $\mathbb{E}_{q^*(x)} [KL(q_{\phi}(z|x) || p(z|x))]$  as we don't know  $p(z|x)$ .

# (Amortized) Variational Inference

## NELBO Derivation

- Solution: Apply Bayes' law to  $p(z|x)$  and move out intractable  $\log p(x)$  term.

$$\mathbb{E}_{q^*(x)}[KL(q_\phi(z|x)||p(z|x))] = \mathbb{E}_{q^*(x)q_\phi(z|x)}[\log q(z) - \log p(x|z) - \log p(z) + \log p(x)]$$

$$\mathbb{E}_{q^*(x)}[KL(q_\phi(z|x)||p(z|x)) - \log p(x)] = -\mathbb{E}_{q^*(x)q(z)}[\log p(x|z)] + \mathbb{E}_{q^*(x)}KL[q_\phi(z|x)||p(z)]$$

- Since  $p(x)$  is constant, we have

$$\begin{aligned} q_\phi^*(z|x) &= \arg \min_{q_\phi(z|x) \in \mathcal{Q}} KL(q_\phi(z|x)||p(z|x)) \\ &= \arg \min_{q_\phi(z|x) \in \mathcal{Q}} -\mathbb{E}_{q^*(x)q(z)}[\log p(x|z)] + \mathbb{E}_{q^*(x)}KL[q_\phi(z|x)||p(z)] \end{aligned}$$

- We refer to  $-\mathbb{E}_{q^*(x)q(z)}[\log p(x|z)] + \mathbb{E}_{q^*(x)}KL[q_\phi(z|x)||p(z)]$  as the negative evidence lower bound  $NELBO(q)$  because it attains a minimum at  $\mathbb{E}_{q^*(x)}[-\log p(x)]$ .

# (Amortized) Variational Inference

## Optimization Problem

- Since  $q_\phi(z|x)$  is represented as a neural network, optimisation involves minimizing the loss function  $NELBO(q)$  with respect to  $\phi$ :

$$\min_{\phi} -\mathbb{E}_{q_\phi(z|x)q^*(x)}[\log p(x|z)] + \mathbb{E}_{q^*(x)}[KL(q_\phi(z|x)||p(z))].$$

- Problem: Neural networks are deterministic, but distributions are probabilistic.

# (Amortized) Variational Inference

## Stochastic Neural Networks

Two main solutions:

- 1 Instead of neural network outputting  $z$ , it outputs  $\mu$  and  $\sigma^2$ , and we have  $z \sim \mathcal{N}(\mu, \sigma^2 I_{M \times M})$ .
- 2 Add random noise  $\epsilon \sim \pi(\epsilon)$  as additional input alongside  $x$ . Typically  $\pi(\epsilon) = \mathcal{N}(0, I_{n \times n})$ .

With the first solution, we know  $q_\phi(z|x)$  and can easily evaluate  $KL(q_\phi(z|x) || p(z))$ .

In the second solution,  $q_\phi(z|x)$  is implicit, so we must resort to density ratio estimation techniques to estimate  $\frac{q_\phi(z|x)}{p(z)}$ . In this thesis, we restrict ourselves to the second solution.

# (Amortized) Variational Inference

## Implicit Distributions

- Recall the optimization problem

$$\min_{\phi} -\mathbb{E}_{q_{\phi}(z|x)q^*(x)}[\log p(x|z)] + \mathbb{E}_{q^*(x)}[KL(q_{\phi}(z|x)||p(z))].$$

- Since we use density ratio estimation to evaluate  $KL(q_{\phi}(z|x)||p(z))$ , the prior  $p(z)$  can also be implicit. This optimization problem is known as the “prior-contrastive” formulation.
- If the likelihood  $p(x|z)$  is implicit, then our optimization problem is

$$\min_{\phi} KL(q(z, x)||p(z, x)).$$

We call this the “joint-contrastive” formulation.

# (Amortized) Variational Inference

## Autoencoders

- An autoencoder consists of an encoder  $q_\phi(z|x)$  that ‘compresses’ data  $x$  into lower dimensional latent  $z$ , and a decoder  $p_\theta(x|z)$  that generates data  $x$  from  $z$ .
- This is the same as our Bayesian inference problem except the likelihood is a neural network parametrised with  $\theta$ .
- To generate new data, sample  $z \sim p(z)$  and pass through decoder.
- We typically know  $p_\theta(x|z)$  in explicit form e.g. if  $x$  is grey-scale image data, then  $p_\theta(x|z)$  is a Bernoulli distribution with parameters given by sigmoid output of the neural network.
- Therefore have optimization problem
$$\min_{\theta, \phi} -\mathbb{E}_{q_\phi(z|x)q^*(x)}[\log p_\theta(x|z)] + \mathbb{E}_{q^*(x)}[KL(q_\phi(z|x)||p(z))].$$
- If  $p_\theta(x|z)$  is implicit then we have the optimization problems  $\min_\phi KL(q(z, x)||p(z, x))$  and  $\min_\theta KL(p(z, x)||q(z, x))$ .

# Outline

## 1 Background Info

- Neural Networks
- (Amortized) Variational Inference
- **Density Ratio Estimation**

## 2 Early Experimentation

- Initial Inference Experiment
- Inference Experiment - Activation Function

## 3 Theory Break

## 4 Experiments

- Inference Experiment

## 5 Further Estimator Loss Function Analysis

## 6 Generation Experiment



# Density Ratio Estimation

## Class Probability Estimation

We want to estimate  $\frac{q(u)}{p(u)}$ .

- 1 Label  $n$  samples from  $p(u)$ :  $U_p = \{u_1^{(p)}, \dots, u_m^{(p)}\}$  with  $y = 0$ .
- 2 Label  $n$  samples from  $q(u)$ :  $U_q = \{u_1^{(q)}, \dots, u_m^{(q)}\}$  with  $y = 1$ .
- 3 Applying Bayes' theorem, we have an expression for the density ratio.

$$\frac{q(u)}{p(u)} = \frac{P(u|y=1)}{P(u|y=0)} = \frac{P(y=1|u)}{P(y=0|u)} \text{ as } P(y=1) = P(y=0)$$

- 4 Define discriminator function as neural network parametrised by  $\alpha$ :  
 $D_\alpha(u) \simeq P(y=1|u)$ , so that  $\frac{q(u)}{p(u)} \simeq \frac{D_\alpha(u)}{1-D_\alpha(u)}$ .
- 5 Train discriminator with Bernoulli loss:  
 $\min_\alpha -\mathbb{E}_{q(u)}[\log D_\alpha(u)] - \mathbb{E}_{p(u)}[\log(1 - D_\alpha(u))]$ .
- 6 Optimal discriminator is  $D_\alpha^*(u) = \frac{q(u)}{q(u)+p(u)}$ .

# Density Ratio Estimation

## Class Probability Estimation

Prior-Contrastive Application:

$$\min_{\alpha} -\mathbb{E}_{q^*(x)q_{\phi}(z|x)}[\log D_{\alpha}(z, x)] - \mathbb{E}_{q^*(x)p_{\theta}(z)}[\log(1 - D_{\alpha}(z, x))]$$

$$\min_{\phi} -\mathbb{E}_{q^*(x)q_{\phi}(z|x)}[\log p(x|z)] + \mathbb{E}_{q^*(x)q_{\phi}(z|x)} \left[ \log \frac{D_{\alpha}(z, x)}{1 - D_{\alpha}(z, x)} \right]$$

Joint-Contrastive Application:

$$\min_{\alpha} -\mathbb{E}_{q^*(x)q_{\phi}(z|x)}[\log D_{\alpha}(z, x)] - \mathbb{E}_{p(z)p(x|z)}[\log(1 - D_{\alpha}(z, x))]$$

$$\min_{\phi} \mathbb{E}_{q^*(x)q_{\phi}(z|x)} \log \frac{D_{\alpha}(z, x)}{1 - D_{\alpha}(z, x)}$$

Program alternates between several optimisation steps of discriminator and one optimisation step of posterior.

# Density Ratio Estimation

## Divergence Minimisation

### Theorem

*If  $f$  is a convex function with derivative  $f'$  and convex conjugate  $f^*$ , and  $\mathcal{R}$  is a class of functions with codomains equal to the domain of  $f'$ , then we have the lower bound for the  $f$ -divergence between distributions  $p(u)$  and  $q(u)$ :*

$$D_f[p(u)||q(u)] \geq \sup_{r \in \mathcal{R}} \{ \mathbb{E}_{q(u)}[f'(r(u))] - \mathbb{E}_{p(u)}[f^*(f'(r(u)))] \},$$

*with equality when  $r(u) = q(u)/p(u)$ .*

For the reverse KL divergence,  $f(u) = u \log u$  so we have

$$KL[q(u)||p(u)] \geq \sup_{r \in \mathcal{R}} \{ \mathbb{E}_{q(u)}[1 + \log r(u)] - \mathbb{E}_{p(u)}[r(u)] \}$$

# Density Ratio Estimation

## Divergence Minimisation

- Let our ratio estimator be a neural network parametrised by  $\alpha$ :  
 $r_\alpha(u) \simeq \frac{q(u)}{p(u)}$ .
- For a fixed  $q(u)$ ,  $KL[q(u)||p(u)]$  is constant, so we can maximise the lower bound with respect to  $\alpha$  until it reaches equality, which is when  $r_\alpha(u) = \frac{q(u)}{p(u)}$ . The optimisation problem for this is

$$\min_{\alpha} -\mathbb{E}_{q(u)}[\log r_\alpha(u)] + \mathbb{E}_{p(u)}[r_\alpha(u)].$$

- Obviously our optimal ratio estimator is  $r_\alpha^*(u) = \frac{q(u)}{p(u)}$ .

# Density Ratio Estimation

## Divergence Minimisation

Prior-Contrastive Application:

$$\min_{\alpha} -\mathbb{E}_{q^*(x)q_{\phi}(z|x)}[\log r_{\alpha}(z, x)] + \mathbb{E}_{q^*(x)p(z)}[r_{\alpha}(z, x)]$$

$$\min_{\phi} -\mathbb{E}_{q^*(x)q_{\phi}(z|x)}[\log p(x|z)] + \mathbb{E}_{q^*(x)q_{\phi}(z|x)}[\log r_{\alpha}(z, x)]$$

Joint-Contrastive Application:

$$\min_{\alpha} -\mathbb{E}_{q^*(x)q_{\phi}(z|x)}[\log r_{\alpha}(z, x)] + \mathbb{E}_{p(z)p(x|z)}[r_{\alpha}(z, x)]$$

$$\min_{\phi} \mathbb{E}_{q^*(x)q_{\phi}(z|x)}[\log r_{\alpha}(z, x)]$$

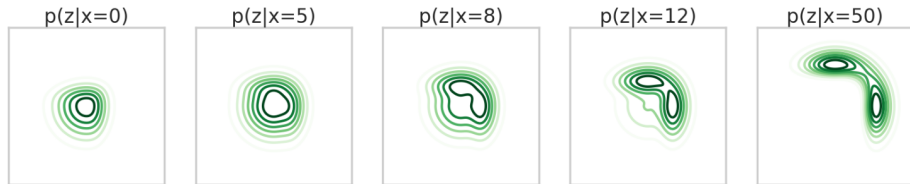
- 1 Background Info
  - Neural Networks
  - (Amortized) Variational Inference
  - Density Ratio Estimation
- 2 Early Experimentation
  - Initial Inference Experiment
  - Inference Experiment - Activation Function
- 3 Theory Break
- 4 Experiments
  - Inference Experiment
- 5 Further Estimator Loss Function Analysis
- 6 Generation Experiment

# Initial Inference Experiment

## Experiment Outline

$$p(z_1, z_2) \sim \mathcal{N}(0, \sigma^2 I_{2 \times 2})$$

$$p(x|z) \sim \text{EXP}(3 + \max(0, z_1)^3 + \max(0, z_2)^3)$$



- Posterior is flexible and bimodal.
- Use implicit form of variational posterior, inputting  $\epsilon \sim \mathcal{N}(0, I_{3 \times 3})$  alongside  $x$  in neural network.

# Inference Experiment

## Experiment Outline

### Class Probability Estimation vs Divergence Minimisation

- Learning rate of 0.0001 for prior-contrastive and 0.00001 for joint-contrastive.
- 5000 training steps of estimator for initialisation, then alternation between 100 estimator steps and 1 posterior step for 10000 posterior iterations in prior-contrastive and 40000 posterior iterations in joint-contrastive.
- Sigmoid output activation function for class probability estimation and ReLU output for divergence minimisation.
- Only trained on data values  $x = 0, 5, 8, 12, 50$ , with 200 samples of each taken per training batch (training batch size is 1000).
- Both algorithms have identical neural network structure and runtime.
- Small constant of  $c = 10^{-18}$  added to log function input to prevent  $\log 0$ .



# Initial Inference Experiment

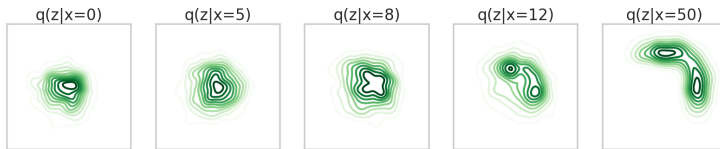
## Results

- Estimator loss and estimated NELBO recorded every iteration.
- Gaussian kernel density estimator  $\hat{q}_{\phi}(z|x)$  used to estimate posterior density for  $x = 0, 5, 8, 12, 50$ , and average KL divergence between these 5 values was recorded every 100 iterations.
- Experiment repeated 30 times and the 3 arrays of recorded values were averaged.

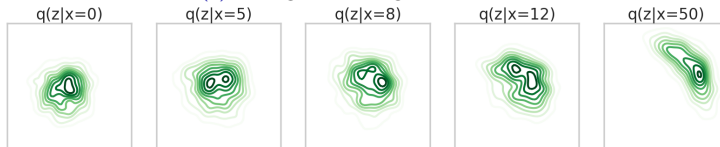
Algorithm	Mean KL Divergence	Standard Deviation
PC Divergence Minimisation	1.3807	0.0391
PC Class Probability Estimation	1.3267	0.0041
JC Divergence Minimisation	1.6954	0.4337
JC Class Probability Estimation	1.3925	0.0669

# Initial Inference Experiment

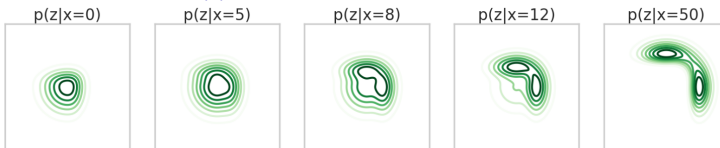
## Example Posterior Outputs



(a) Average KL Divergence of 1.3288



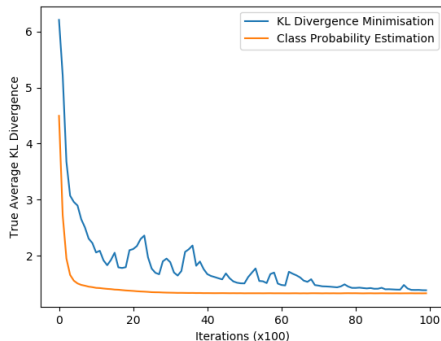
(b) Average KL Divergence of 1.3963



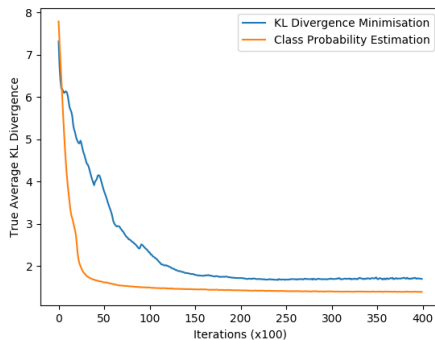
(c) True Posterior Plot

# Initial Inference Experiment

## KL Divergence Plots



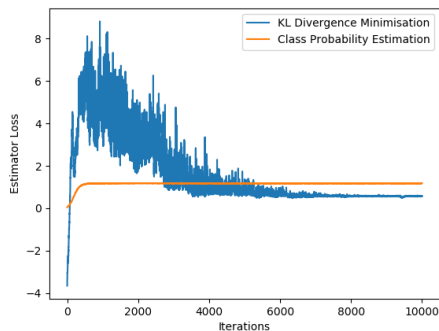
(a) Prior-Contrastive



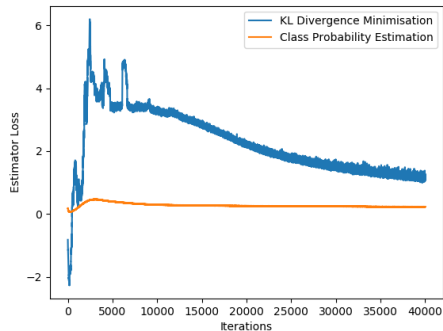
(b) Joint-Contrastive

# Initial Inference Experiment

## Estimator Losses



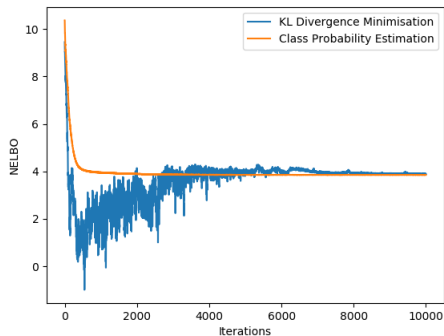
(a) Prior-Contrastive



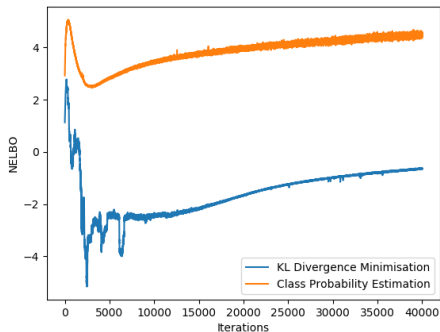
(b) Joint-Contrastive

# Initial Inference Experiment

## NELBOs



(a) Prior-Contrastive



(b) Joint-Contrastive

# Initial Inference Experiment

## Failures

- Roughly 40% of the Divergence Minimisation experiments resulted in 'failures', in which the estimator loss initialised at 41.4465 and remained constant over optimisation steps.
- Program was set to restart every time this happened.
- Analysis of estimator output showed that it was outputting ratio of 0.
- Recall ratio estimator loss of  $-\mathbb{E}_q[\log r_\alpha(z, x) + \mathbb{E}_p[r_\alpha(z, x)]]$  and our added constant term of  $c = 10^{-18}$ .
- $-\log 10^{-18} = 41.4465$
- This means neural network initialised to output negative number which is mapped to 0 by ReLU.
- Partial derivative of loss function w.r.t weights is 0 as changing weight values slightly still results in negative output before ReLU.

# Initial Inference Experiment

## Problems with ReLU

- ‘Failures’ caused from ReLU outputting in  $[0, \infty)$  despite  $\frac{q(u)}{p(u)} \in (0, \infty)$ .
- If  $q(u) < p(u)$ ,  $\frac{q(u)}{p(u)} \in (0, 1)$ , and if  $q(u) > p(u)$ ,  $\frac{q(u)}{p(u)} \in (1, \infty)$ .
- Linearity of ReLU activation results in inconsistent training, as small training steps should be taken if  $q(u) < p(u)$ , but large training steps required for  $q(u) > p(u)$ .
- First contribution of thesis: we propose exponential activation function  $g(x) = e^x$  for ratio estimator.
- This maps  $\mathbb{R}^-$  to  $(0, 1)$ , and  $\mathbb{R}^+$  to  $(1, \infty)$ .
- Training is consistent and neural network cannot output 0.

# Outline

- 1 Background Info
  - Neural Networks
  - (Amortized) Variational Inference
  - Density Ratio Estimation
- 2 Early Experimentation
  - Initial Inference Experiment
  - Inference Experiment - Activation Function
- 3 Theory Break
- 4 Experiments
  - Inference Experiment
- 5 Further Estimator Loss Function Analysis
- 6 Generation Experiment



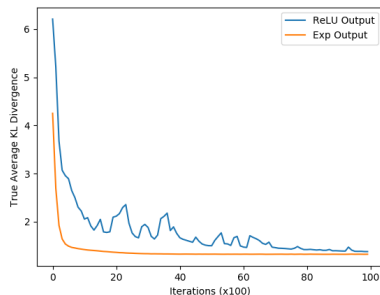
# Inference Experiment - Activation Function

Same experiment setup and parameters as previous experiment, but now we are comparing divergence minimisation with ReLU and exponential outputs for the ratio estimator.

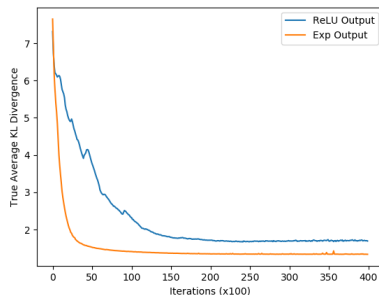
Algorithm	Mean KL Divergence	Standard Deviation
PC Divergence Minimisation - ReLU	1.3807	0.0391
PC Divergence Minimisation - Exp	1.3265	0.0045
JC Divergence Minimisation - ReLU	1.6954	0.4337
JC Divergence Minimisation - Exp	1.3397	0.0066

# Inference Experiment - Activation Function

## KL Divergence Plots



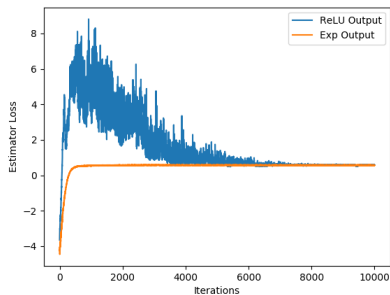
(a) Prior-Contrastive



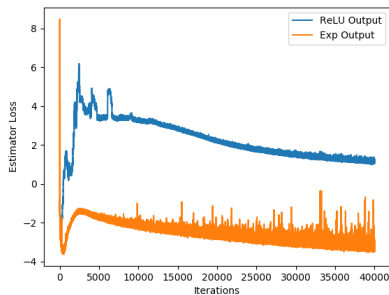
(b) Joint-Contrastive

# Inference Experiment - Activation Function

## Estimator Losses



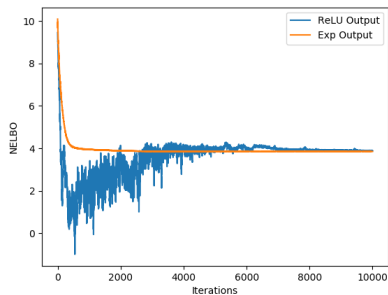
(a) Prior-Contrastive



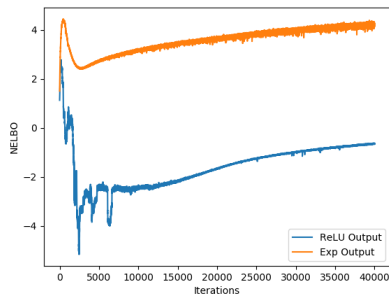
(b) Joint-Contrastive

# Inference Experiment - Activation Function

## NELBOs



(a) Prior-Contrastive



(b) Joint-Contrastive

# Theory Break

## Alternative Derivation of Class Probability Estimation

- Recall theorem behind divergence minimisation:

$$D_f[p(u)||q(u)] \geq \sup_{r \in \mathcal{R}} \{ \mathbb{E}_{q(u)}[f'(r(u))] - \mathbb{E}_{p(u)}[f^*(f'(r(u)))] \},$$

- If we let  $f(u) = u \log u - (u + 1) \log(u + 1)$  and  $D(u) = \frac{r(u)}{r(u)+1}$ , we have the lower bound

$$2JS[p(u)||q(u)] - \log 4 \geq \sup_D \{ \mathbb{E}_{q(u)}[\log D(u)] + \mathbb{E}_{p(u)}[\log(1 - D(u))] \}$$

- This is the same estimator loss as in class probability estimation:

$$\min_{\alpha} -\mathbb{E}_{q(u)}[\log D_{\alpha}(u)] - \mathbb{E}_{p(u)}[\log(1 - D_{\alpha}(u))]$$

- We call  $2JS[p(u)||q(u)] - \log 4$  the 'CPE' divergence

# Theory Break

## Analysis of Optimisation Algorithms

- When comparing class probability estimation and divergence minimisation, we are just comparing the f-divergence and estimator parametrisation (factors of Estimator Loss).
- 2 f-divergences being compared:  $KL[q(u)||p(u)]$  and  $2JS[p(u)||q(u)] - \log 4$ .
- Estimators being compared are class probability estimator  $D_\alpha(u) \simeq \frac{q(u)}{q(u)+p(u)}$  (Sigmoid Output) and direct ratio estimator  $r_\alpha(u) \simeq \frac{q(u)}{p(u)}$  (Exp Output).
- We also propose direct log ratio estimator  $T_\alpha(U) \simeq \log \frac{q(u)}{p(u)}$  (Linear Output).
- 2 problem contexts (PC+JC)  $\times$  2 f-divergences  $\times$  3 estimator parametrisations = 12 experiments.

# Outline

- 1 Background Info
  - Neural Networks
  - (Amortized) Variational Inference
  - Density Ratio Estimation
- 2 Early Experimentation
  - Initial Inference Experiment
  - Inference Experiment - Activation Function
- 3 Theory Break
- 4 Experiments
  - Inference Experiment
- 5 Further Estimator Loss Function Analysis
- 6 Generation Experiment

# Inference Experiment

## Comparing Optimal Estimators

- Same experimental setup as previous 2 experiments.
- Aim of this experiment is to verify that within each problem context, if the estimator reaches its optimal state, then similar levels of convergence will be observed regardless of the f-divergence or parametrisation.
- Low training rate with high estimator to posterior optimisation ratio (100:1) should ensure that estimator reaches near-optimal state before it is used to estimate NELBO.



# Inference Experiment

## Comparing Optimal Estimators

Algorithm	Mean KL Divergence	Standard Deviation
PC Reverse KL - $D_\alpha(z, x)$	1.3271	0.0041
PC Reverse KL - $r_\alpha(z, x)$	1.3265	0.0045
PC Reverse KL - $T_\alpha(z, x)$	1.3262	0.0041
PC CPE - $D_\alpha(z, x)$	1.3267	0.0041
PC CPE - $r_\alpha(z, x)$	1.3263	0.0035
PC CPE - $T_\alpha(z, x)$	1.3258	0.0039
JC Reverse KL - $D_\alpha(z, x)$	1.3416	0.0068
JC Reverse KL - $r_\alpha(z, x)$	1.3397	0.0066
JC Reverse KL - $T_\alpha(z, x)$	1.3446	0.0108
JC CPE - $D_\alpha(z, x)$	1.3925	0.0669
JC CPE - $r_\alpha(z, x)$	1.3915	0.0502
JC CPE - $T_\alpha(z, x)$	1.3670	0.0387

- Similar means and standard deviations within prior-contrastive methods and for joint-contrastive algorithms using reverse KL divergence.

# Inference Experiment

## Comparing Optimal Estimators

- Joint-contrastive algorithms using CPE divergence showed higher, inconsistent means and standard deviations, implying estimator did not reach optimal value as we expected.
- Repeating the experiment for this scenario but with estimator learning rate increased tenfold should lead to similar results (Results pending).
- This result implies that CPE divergence leads to slower estimator convergence.

# Inference Experiment

## Comparing Undertrained Estimators

- Aim of this experiment is to significantly reduce the amount of training the estimator undergoes between each NELBO estimation.
- The combination of f-divergence and estimator parametrisation that trains the fastest will have the highest accuracy, corresponding to the highest posterior convergence.
- Estimator training rate changed to 0.00004 and posterior training rate increased to 0.0002.
- 5000 estimator initialisation steps retained.
- Estimator to posterior iteration ratio reduced to 15:1 in prior-contrastive and 20:1 in joint-contrastive.
- Total posterior iterations reduced to 2000 in prior-contrastive and 4000 in joint-contrastive.

# Inference Experiment

## Comparing Undertrained Estimators

Algorithm	Mean KL Divergence	Standard Deviation
PC Reverse KL - $D_\alpha(z, x)$	1.3572	0.0136
PC Reverse KL - $r_\alpha(z, x)$	1.3607	0.0199
PC Reverse KL - $T_\alpha(z, x)$	1.3641	0.0141
PC CPE - $D_\alpha(z, x)$	1.3788	0.0258
PC CPE - $r_\alpha(z, x)$	1.3811	0.0365
PC CPE - $T_\alpha(z, x)$	1.3849	0.0450
JC Reverse KL - $D_\alpha(z, x)$	1.3786	0.0286
JC Reverse KL - $r_\alpha(z, x)$	1.3934	0.0410
JC Reverse KL - $T_\alpha(z, x)$	1.4133	0.0597
JC CPE - $D_\alpha(z, x)$	1.4017	0.0286
JC CPE - $r_\alpha(z, x)$	1.4086	0.0555
JC CPE - $T_\alpha(z, x)$	1.4214	0.0518

- Reverse KL divergence significantly better than CPE divergence.
- For all 4 experiment environments,  $D_\alpha(z, x) < r_\alpha(z, x) < T_\alpha(z, x)$  in terms of mean KL divergence but not by a significant amount.

# Inference Experiment

## Comparing Undertrained Estimators

- Standard deviation of class probability estimator consistently better than other two estimator parametrisations.
- f-divergence used is more significant than estimator parametrisation.
- Optimal combination is reverse KL divergence with class probability estimator.

# Further Estimator Loss Function Analysis

## Outline

- Each estimator loss function is a convex functional that reaches its minimum when estimator is optimal.
- Estimator parametrisation affects its output space and gradient of loss function with respect to the estimator.
- Choice of  $f$ -divergence affects gradient of loss function with respect to estimator.

# Further Estimator Loss Function Analysis

## Estimator Parametrisation

Need some plots xD

- Higher second derivative corresponds to faster convergence.
- Taking second functional derivative of estimator loss function with respect to estimator, we can only make one certain comparison: for the CPE divergence, the class probability estimator has a strictly higher second derivative than the direct ratio estimator.
- The density ratio changes every time the posterior is optimised, and the estimator must catch up. It can be shown that the class probability estimator has a strictly lower displacement than the direct ratio estimator, that is,  $|D_{final}^* - D_{init}^*| < |r_{final}^* - r_{init}^*|$ .

# Further Estimator Loss Function Analysis

## Choice of f-divergence

- Again observing the second functional derivatives, we can only observe that in the direct ratio estimator parametrisation, the reverse KL divergence is strictly higher than the CPE divergence.
- Nowozin's f-GAN paper also shows empirically that the reverse KL divergence is superior when it is additionally used to optimize the posterior.



# Generation Experiment

## Experiment Outline

- MNIST dataset -  $28 \times 28$  grey-scale images of handwritten digits
- Autoencoder formulation
- Not doing joint-contrastive cause unintuitive to 'pretend' we don't know likelihood function.
- Parameters are:
- Record estimator losses, NELBOs and reconstruction errors.
- Perform experiment with low dimensional latent space (2 dimensions) and high dimensional latent space (20 dimensions).

# Generation Experiment

Results - low dimensional latent space

Basically same outcome as in inference experiment: best combination is reverse KL +  $D_{\alpha}(z, x)$ .

# Generation Experiment



## Results - high dimensional latent space

- Direct ratio and direct log ratio estimators failed.
- $r_\alpha(z, x)$  outputted density ratio estimation exceeding  $\text{float64}(\text{max})$ , overflowing to  $\text{Inf}$ .
- Even though  $T_\alpha(z, x)$  outputs the log density ratio, the exponential is taken in the estimator loss function, resulting in a value exceeding  $\text{float64}(\text{max})$ .
- $D_\alpha(z, x)$  does not experience this issue as output ranges in  $(0, 1)$
- It can be shown that the value in the neural network before passing through the sigmoid output is the estimation of  $\log \frac{q(x)}{p(x)}$ . Therefore the class probability estimator parametrisation does not experience values exceeding  $\text{float64}(\text{max})$ .
- This alone shows the class probability estimator is the best parametrisation for the vast majority of problems that use variational Bayesian machine learning approaches, as they are typically high-dimensional data generation problems.
- Also reverse KL divergence is again better than CPE divergence.

# Summary

- The **first main message** of your talk in one or two lines.
- The **second main message** of your talk in one or two lines.
- Perhaps a **third message**, but not more than that.
- Outlook
  - Something you haven't solved.
  - Something else you haven't solved.

# For Further Reading I

-  A. Author.  
*Handbook of Everything*.  
Some Press, 1990.
-  S. Someone.  
On this and that.  
*Journal of This and That*, 2(1):50–100, 2000.