Selected topics of Probabilities in Deep Learning

A/Prof Richard Yi Da Xu
richardxu.com

University of Technology Sydney (UTS)

August 18, 2018

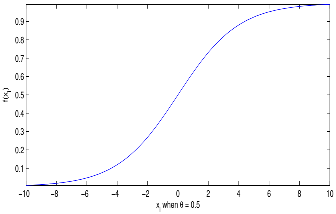
Noise Contrastive Estimation

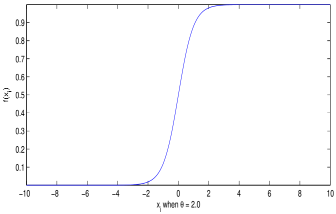
- firstly, probability models and classification are closely related:

$$\arg\max_{\theta} \big(p_\theta(\mathbf{Y})\big) \implies \arg\min_{\theta} \big(- \log p_\theta(\mathbf{Y})\big)$$

- in following example, let's show **classification models** incorporating our favorite sigmoid function:

$$\sigma(\mathbf{x}_i^\top \theta) = \frac{1}{1 + \exp(-\mathbf{x}_i^T \boldsymbol{\theta})}$$





Example: Bernoulli & Logistic regression

- Bernoulli distribution using Sigmoid function

$$p_{\boldsymbol{\theta}}(\mathbf{Y}|\mathbf{X}) = \prod_{i=1}^{n} \left[\frac{1}{1 + \exp(-\mathbf{x}_i^T \boldsymbol{\theta})} \right]^{y_i} \left[1 - \frac{1}{1 + \exp(-\mathbf{x}_i^T \boldsymbol{\theta})} \right]^{1-y_i}$$

- Logistic regression

$$\begin{aligned}
\mathcal{C}(\boldsymbol{\theta}) &= -\log[p_{\boldsymbol{\theta}}(\mathbf{Y}|\mathbf{X})] \\
&= -\left(\sum_{i=1}^{n} y_i \log \left[\frac{1}{1 + \exp(-\mathbf{x}_i^T \boldsymbol{\theta})} \right] + (1 - y_i) \log \left[1 - \frac{1}{1 + \exp(-\mathbf{x}_i^T \boldsymbol{\theta})} \right] \right)
\end{aligned}$$

Example: Multinomial Distribution & Cross Entropy Loss

- Multinomial Distribution with softmax

$$p_{\boldsymbol{\theta}}(\mathbf{Y}|\mathbf{X}) = \prod_{i=1}^{n}\prod_{k=1}^{K}\left[\left(\frac{\exp(\mathbf{x}_i^T\boldsymbol{\theta}_k)}{\sum_{l=1}^{K}\exp(\mathbf{x}_i^T\boldsymbol{\theta}_l)}\right)\right]^{y_{i,k}}$$

- cross entropy loss with Softmax

$$\mathcal{C}(\boldsymbol{\theta}) = -\log[p_{\boldsymbol{\theta}}(\mathbf{Y}|\mathbf{X})] = -\sum_{i=1}^{N}\sum_{k=1}^{K} y_{i,k}\left[\log\left(\frac{\exp(\mathbf{x}_i^T\boldsymbol{\theta}_k)}{\sum_{l=1}^{K}\exp(\mathbf{x}_i^T\boldsymbol{\theta}_l)}\right)\right]$$

Example: Gaussian Distribution & Sum of Square Loss

- this time, let's go from $\mathcal{C}(\boldsymbol{\theta}) \rightarrow p_{\theta}(\mathbf{Y})$
- Sum of Square Loss

$$\mathcal{C}(\boldsymbol{\theta}) = \sum_{k=1}^{K} \left(\hat{y}_k(\boldsymbol{\theta}) - y_k\right)^2$$

- Gaussian distribution

$$p_{\boldsymbol{\theta}}(\mathbf{Y}|\mathbf{X}) \propto \exp\left[-\mathcal{C}(\boldsymbol{\theta})\right] = \exp\left[-\sum_{k=1}^{K} \left(\hat{y}_k(\boldsymbol{\theta}) - y_k\right)^2\right]$$

- **question**: what if we use *Square* loss instead of *Cross Entropy* loss in Softmax, where:

$$\hat{y}_k(\boldsymbol{\theta}) = \frac{\exp(\mathbf{x}_i^T \boldsymbol{\theta}_k)}{\sum_{l=1}^{K} \exp(\mathbf{x}_i^T \boldsymbol{\theta}_l)}$$

Think about Classification's best friend, "Softmax" again!

- for example, in word embedding, we want to align a target word $\mathbf{u}_w$ with center word $\mathbf{v}_c$:
- for simplicity, for the rest of the article, we let $\mathbf{w} \equiv \mathbf{u}_w$ and $\mathbf{c} \equiv \mathbf{v}_c$

$$\Pr_\theta(\mathbf{w}|\mathbf{c}) = \frac{u_\theta(\mathbf{w}|\mathbf{c})}{\sum_{w' \in \mathcal{V}} u_\theta(\mathbf{w}'|\mathbf{c})} = \frac{u_\theta(\mathbf{w}|\mathbf{c})}{Z_c} \equiv \frac{\exp(\mathbf{w}^\top \mathbf{c})}{\sum_{\mathbf{w}' \in \mathcal{V}} \exp(\mathbf{w}'^\top \mathbf{c})}$$

- the denominator, i.e., the $\sum_{\mathbf{w}' \in \mathcal{V}} u(\mathbf{w}'|\mathbf{c})$ can be too computational

Turn the problem around!

- **data distribution**: we sample $\mathbf{w} \sim \bar{p}(\mathbf{w}|\mathbf{c})$ from its empirical (data) distribution, and give a label $\mathcal{Y} = 1$
- **noise distribution**: we can sample k $\bar{\mathbf{w}} \sim q(\mathbf{w})$, and give them labels $\mathcal{Y} = 0$
  importantly, condition for $q(.)$ is: it does **not** assign zero probability to any data.
- Can we build a binary classifier to **classify** its label, i.e., which distribution has generated it?

Noise Contrastive Estimation (NCE)

- **training data generation**: (w, c, y)
  1. sample $(\mathbf{w}, \mathbf{c})$: using $\mathbf{c} \sim \tilde{p}(\mathbf{c})$, $\mathbf{w} \sim \tilde{p}(\mathbf{w}|\mathbf{c})$ and label them as $\mathcal{Y} = 1$
  2. k "noise" samples from $q(.)$, and label them as $\mathcal{Y} = 0$
- can we instead, try to maximize the joint posterior Bernoulli distribution:

$$\Pr_\theta(\mathcal{Y}|\mathbf{W}, \mathbf{c}) = \prod_{i=1}^{k+1} \big(\Pr(\mathcal{Y}_i|\mathbf{w}_i, \mathbf{c})\big)^{y_i} \big(1 - \Pr(\mathcal{Y}_i|\mathbf{w}_i, \mathbf{c})\big)^{1-y_i}$$

- or minimize the corresponding Logistic regression:

$$\begin{aligned}
\mathcal{C} &= -\log[\Pr_\theta(\mathcal{Y}|\mathbf{W}, \mathbf{c})] \\
&= -\sum_{i=1}^{k+1} y_i \log\left[\Pr_\theta(\mathcal{Y}_i|\mathbf{w}_i, \mathbf{c})\right] + (1 - y_i) \log\left[1 - \Pr_\theta(\mathcal{Y}_i|\mathbf{w}_i, \mathbf{c})\right]
\end{aligned}$$

Noise Contrastive Estimation (NCE)

- we assume there are k negative samples per positive sample, so the prior density is:

$$P(\mathcal{Y} = y) = \begin{cases} \frac{1}{k+1} & y = 1 \\ \frac{k}{k+1} & y = 0 \end{cases}$$

- then the posterior of $P(\mathcal{Y}|\mathbf{c}, \mathbf{w})$:

$$\begin{aligned}
P(\mathcal{Y} = 1|\mathbf{c}, \mathbf{w}) = \frac{\Pr(\mathcal{Y} = 1, \mathbf{w}|\mathbf{c})}{\Pr(\mathbf{w}|\mathbf{c})} &= \frac{\Pr(\mathbf{w}|\mathcal{Y} = 1, \mathbf{c})P(\mathcal{Y} = 1)}{\sum_{y \in \{0,1\}} p(w|\mathcal{Y} = y, \mathbf{c})P(\mathcal{Y} = y)} \\
&= \frac{\tilde{p}(\mathbf{w}) \times \frac{1}{1+k}}{\tilde{P}(\mathbf{w}|\mathbf{c}) \times \frac{1}{k+1} + q(\mathbf{w}) \times \frac{k}{1+k}} \\
&= \frac{\tilde{P}(\mathbf{w}|\mathbf{c})}{\tilde{P}(\mathbf{w}|c) + kq(\mathbf{w})}
\end{aligned}$$

$$\begin{aligned}
\Pr(\mathcal{Y} = 0|\mathbf{c}, \mathbf{w}) &= 1 - \Pr(\mathcal{Y} = 1|\mathbf{c}, \mathbf{w}) \\
&= 1 - \frac{\tilde{P}(\mathbf{w}|\mathbf{c})}{\tilde{P}(\mathbf{w}|\mathbf{c}) + kq(\mathbf{w})} \\
&= \frac{kq(\mathbf{w})}{\tilde{P}(\mathbf{w}|\mathbf{c}) + kq(\mathbf{w})}
\end{aligned}$$

Apply NCE to NLP problem

- in summary:

$$\Pr(\mathcal{Y} = y|\mathbf{c}, \mathbf{w}) = \begin{cases} \frac{\tilde{P}(\mathbf{w}|\mathbf{c})}{\tilde{P}(\mathbf{w}|\mathbf{c}) + kq(\mathbf{w})} & y = 1 \\ \frac{kq(\mathbf{w})}{\tilde{P}(\mathbf{w}|\mathbf{c}) + kq(\mathbf{w})} & y = 0 \end{cases}$$

- it can be replaced by un-normalized function:

$$\Pr(\mathcal{Y} = y|\mathbf{c}, \mathbf{w}) = \begin{cases} \frac{u_\theta(\mathbf{w}|\mathbf{c})}{u_\theta(\mathbf{w}|\mathbf{c}) + kq(\mathbf{w})} & y = 1 \\ \frac{kq(\mathbf{w})}{u_\theta(\mathbf{w}|\mathbf{c}) + kq(\mathbf{w})} & y = 0 \end{cases}$$

- formal proof can be found "*Gutmann, 2012, Noise-Contrastive Estimation of Unnormalized Statistical Models, with Applications to Natural Image Statistics*"
- let's see an intuition through **softmax**

Intuition through Softmax

- think about Softmax in word embedding:

$$\Pr_\theta(\mathbf{w}|\mathbf{c}) = \frac{u_\theta(\mathbf{w}|\mathbf{c})}{\sum_{w' \in \mathcal{V}} u_\theta(\mathbf{w}'|\mathbf{c})} = \frac{u_\theta(\mathbf{w}|\mathbf{c})}{Z_c} \equiv \frac{\exp(\mathbf{w}^\top \mathbf{c})}{\sum_{\mathbf{w}' \in \mathcal{V}} \exp(\mathbf{w}'^\top \mathbf{c})}$$

- say $\{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_k\}$ are target words having high frequencies given $\mathbf{c}$
- $\{\mathbf{r}_1, \mathbf{r}_2, \dots \mathbf{r}_n\}$ are words having low frequency given $\mathbf{c}$
- say we pick $\mathbf{w}_i \in \{\mathbf{w}_1, \dots w_k\}$ to optimize: at each round, we aim to increase $\mathbf{w}_i^\top \mathbf{c}$; at the same time, sum of rest of softmax weights: $\left\{\{\mathbf{w}_j^\top \mathbf{c}\}_{j \neq i} \cup \{\mathbf{r}_j^\top \mathbf{c}\}\right\}$ decrease
- in softmax, such decrease is guaranteed by the sum in denominator
- each $\mathbf{w}_i$ has a chance to increase $\mathbf{w}_i^\top \mathbf{c}$, but each $\mathbf{r}_i^\top \mathbf{c}$ will (hopefully) stay low
- **intuition**: in NCE, instead of using sum in the denominator, we "designed" a probability $q(.)$, such that, while letting $\mathbf{w}_i$ be a positive training sample, we also have chance to let $\mathbf{w}_{j \neq i}$ to be part of negative training sample, i.e., to reduce the value of $\mathbf{w}_j^\top \mathbf{c}$; it somewhat has a similar effect as **softmax**

NCE in a nutshell

NCE transforms:

- a problem of model estimation (computationally expensive) to:
- a problem of estimating parameters of probabilistic binary posterior classifier (computationally acceptable):
- main advantage: it allows us to fit models that are not explicitly normalized, making training time effectively independent of the vocabulary size

NCE objective function

- let $u_\theta(\mathbf{w}|\mathbf{c}) = \exp[s_\theta(\mathbf{w}|\mathbf{c})]$:

$$
\begin{aligned}
\Pr(\mathcal{Y} = 1|\mathbf{c}, \mathbf{w}) &= \frac{u_\theta(\mathbf{w}|\mathbf{c})}{u_\theta(\mathbf{w}|\mathbf{c}) + kq(\mathbf{w})} = \sigma\big(\triangle s_\theta(\mathbf{w}|\mathbf{c})\big) \\
\Pr(\mathcal{Y} = 0|\mathbf{c}, \mathbf{w}) &= \frac{kq(\mathbf{w})}{u_\theta(\mathbf{w}|\mathbf{c}) + kq(\mathbf{w})} = 1 - \sigma\big(\triangle s_\theta(\mathbf{w}|\mathbf{c})\big) \\
&\text{where } \triangle s_\theta(\mathbf{w}|\mathbf{c}) \equiv s_\theta(\mathbf{w}|\mathbf{c}) - \log(kq(\mathbf{w})) \qquad \text{let's see why}
\end{aligned}
$$

$$
\begin{aligned}
\sigma\big(\triangle s_\theta(\mathbf{w}|\mathbf{c})\big) &= \frac{1}{1 + \exp\big[- s_\theta(\mathbf{w}|\mathbf{c}) + \log(kq(\mathbf{w}))\big]} \\
&= \frac{1}{1 + \exp\big(- s_\theta(\mathbf{w}|\mathbf{c})\big) \times kq(\mathbf{w})} \\
&= \frac{\exp\big[s_\theta(\mathbf{w}|\mathbf{c})\big]}{\exp\big[s_\theta(\mathbf{w}|\mathbf{c})\big] + kq(\mathbf{w})} = \frac{u_\theta(\mathbf{w}|\mathbf{c})}{u_\theta(\mathbf{w}|\mathbf{c}) + kq(\mathbf{w})}
\end{aligned}
$$

- therefore the objective function is:

$$
\theta^* = \arg\max_\theta \sum_{(\mathbf{w},\mathbf{c})\in D} \sigma(\triangle s_\theta(\mathbf{w}|\mathbf{c})) + \sum_{(\bar{\mathbf{w}},c)\in \tilde{D}} \sigma(-\triangle s_\theta(\bar{\mathbf{w}}|\mathbf{c}))
$$

NCE and Negative Sampling

- **negative sampling** is a special case of NCE
- we let $k = |\mathcal{V}|$ and $q(.)$ is uniform:

$$
\begin{aligned}
P(\mathcal{Y} = 1|\mathbf{c}, \mathbf{w}) &= \frac{u_\theta(\mathbf{w}|\mathbf{c})}{u_\theta(\mathbf{w}|\mathbf{c}) + |\mathcal{V}|\frac{1}{|\mathcal{V}|}} = \frac{u_\theta(\mathbf{w}|\mathbf{c})}{u_\theta(\mathbf{w}|\mathbf{c}) + 1} \\
P(\mathcal{Y} = 0|\mathbf{c}, \mathbf{w}) &= \frac{|\mathcal{V}|\frac{1}{|\mathcal{V}|}}{u_\theta(\mathbf{w}|\mathbf{c}) + |\mathcal{V}|\frac{1}{|\mathcal{V}|}} = \frac{1}{u_\theta(\mathbf{w}|\mathbf{c}) + 1}
\end{aligned}
$$

- correspondingly, we have:

$$
\triangle s_\theta(\mathbf{w}|\mathbf{c}) \equiv s_\theta(\mathbf{w}|\mathbf{c}) - \log\left(|\mathcal{V}|\frac{1}{|\mathcal{V}|}\right) = s_\theta(\mathbf{w}|\mathbf{c}) = \mathbf{w}^\top\mathbf{c}
$$

- in Skip-gram:

$$
\begin{aligned}
\theta^* &= \arg\max_\theta \sum_{(\mathbf{w},\mathbf{c})\in D} \sigma(\mathbf{w}^\top\mathbf{c}) + \sum_{(\bar{\mathbf{w}},c)\in\tilde{D}} \sigma(-\bar{\mathbf{w}}^\top\mathbf{c}) \\
&= \arg\min_\theta \sum_{(w,c)\in D} \sigma(-\mathbf{u}_w^\top\mathbf{v}_c) + \sum_{(\bar{w},c)\in\bar{D}} \frac{1}{1+\exp\left(-\bar{\mathbf{w}}^\top\mathbf{c}\right)}
\end{aligned}
$$

why un-normalised $u_\theta(\mathbf{w}, \mathbf{c})$ still works?

- talk a look at this again, let $u_\theta(\mathbf{w}|\mathbf{c}) = \exp[s_\theta(\mathbf{w}|\mathbf{c})]$:

$$\Pr(\mathcal{Y} = 1|\mathbf{c}, \mathbf{w}) = \frac{u_\theta(\mathbf{w}|\mathbf{c})}{u_\theta(\mathbf{w}|\mathbf{c}) + kq(\mathbf{w})} = \sigma\big(\triangle s_\theta(\mathbf{w}|\mathbf{c})\big)$$

$$\text{where } \triangle s_\theta(\mathbf{w}|\mathbf{c}) \equiv s_\theta(\mathbf{w}|\mathbf{c}) - \log(kq(\mathbf{w}))$$

- we already know:

$$= \sigma\big(\triangle s_\theta(\mathbf{w}|\mathbf{c})\big) = \frac{1}{1 + \underbrace{\exp\big(-s_\theta(\mathbf{w}|\mathbf{c})\big) \times kq(\mathbf{w})}_{G(\mathbf{w},\theta)}}$$

- in this case,

$$G(\mathbf{w}, \theta) = \exp\big(-s_\theta(\mathbf{w}|\mathbf{c})\big) \times kq(\mathbf{w})$$
$$= \frac{kq(\mathbf{w})}{\exp(s_\theta(\mathbf{w}|\mathbf{c}))} = \frac{kq(\mathbf{w})}{u_\theta(\mathbf{w}|\mathbf{c})}$$

- or more generically:

$$G(\mathbf{w}, \theta) = \frac{m}{n}\frac{q(\mathbf{w})}{u_\theta(\mathbf{w}|\mathbf{c})}$$

what do we need to prove?

- look at $G(\mathbf{w}, \theta) = \frac{m}{n} \frac{q(\mathbf{w})}{u_\theta(\mathbf{w}|\mathbf{c})}$:
- $G(\mathbf{w}, \theta)$ is a function of θ, so this ratio changes; However, the **real trick** is if let:

$$\theta^* = \arg\max_\theta \frac{1}{n} \left(\sum_{i=1}^{n} \mathcal{Y}_i \log \Pr(\mathcal{Y}_i = 1|\mathbf{w}_i, \theta) + \sum_{i=1}^{m} (1 - \mathcal{Y}_i) \log[\Pr(\mathcal{Y}_i = 0|\mathbf{w}_i, \theta)] \right)$$

- and we prove the following: (under large sample size n and m):

$$G(\mathbf{w}, \theta^*) \to \frac{m}{n} \frac{q(\mathbf{w})}{p(\mathbf{w})} \implies u_{\theta^*}(\mathbf{w}|\mathbf{c}) \to p(\mathbf{w}) \qquad \text{as } \theta \to \theta^*$$

so why does $G(\mathbf{w}, \theta^*) \to \frac{m}{n}\frac{q(\mathbf{w})}{p(\mathbf{w})}$?

- let,

$$
\begin{aligned}
\mathcal{C}_n(\theta) &= \frac{1}{n}\left(\sum_{i=1}^{n} \mathcal{Y}_i \log \Pr(\mathcal{Y}_i = 1|\mathbf{w}_i, \theta) + \sum_{i=1}^{m}(1 - \mathcal{Y}_i)\log[\Pr(\mathcal{Y}_i = 0|\mathbf{w}_i, \theta)]\right) \\
&= \frac{1}{n}\sum_{i=1}^{n} \mathcal{Y}_i \log \Pr(\mathcal{Y}_i = 1|\mathbf{w}_i, \theta) + \underbrace{\frac{m}{n}}_{\nu}\frac{1}{m}\sum_{i=1}^{m}(1 - \mathcal{Y}_i)\log[\Pr(\mathcal{Y}_i = 0|\mathbf{w}_i, \theta)]
\end{aligned}
$$

- let $n \to \infty$ and $m \to \infty$: $\mathcal{C}_n \to \mathcal{C}$:

$$
\begin{aligned}
\mathcal{C} &= \mathbb{E}_{\mathbf{w}\sim p(\mathbf{w})}[\log \Pr(\mathcal{Y}_i = 1|\mathbf{w}_i, \theta)] + \nu\mathbb{E}_{\mathbf{w}\sim q(\mathbf{w})}[\log[\Pr(\mathcal{Y}_i = 0|\mathbf{w}_i, \theta)]] \\
&= \mathbb{E}_{\mathbf{w}\sim p(\mathbf{w})}\left[\log\frac{1}{1 + G(\mathbf{w}, \theta)}\right] + \nu\mathbb{E}_{\mathbf{w}\sim q(\mathbf{w})}\left[\log\frac{G(\mathbf{w}, \theta)}{1 + G(\mathbf{w}, \theta)}\right] \\
&= -\mathbb{E}_{\mathbf{w}\sim p(\mathbf{w})}\Big[\log(1 + G(\mathbf{w}, \theta))\Big] + \nu\mathbb{E}_{\mathbf{w}\sim q(\mathbf{w})}\Big[\log G(\mathbf{w}, \theta) - \log(1 + G(\mathbf{w}, \theta))\Big] \\
&= -\int \log\big(1 + G(\mathbf{w}, \theta)\big)p(\mathbf{w})d\mathbf{w} + \nu\int\big(\log G(\mathbf{w}, \theta) - \log(1 + G(\mathbf{w}, \theta))\big)q(\mathbf{w})d\mathbf{w}
\end{aligned}
$$

using functional derivative

$$\mathcal{C} = -\int \log\left(1 + G(\mathbf{w},\theta)\right) p(\mathbf{w}) \mathrm{d}\mathbf{w} + \nu \int \left(\log G(\mathbf{w},\theta) - \log(1 + G(\mathbf{w},\theta))\right) q(\mathbf{w}) \mathrm{d}\mathbf{w}$$

- take functional derivative:

$$\begin{aligned}
\frac{\delta\mathcal{C}(G)}{\delta G} &= -\frac{p(\mathbf{w})}{1 + G(\mathbf{w},\theta)} + \nu q(\mathbf{w})\left(\frac{1}{G(\mathbf{w})} - \frac{1}{1 + G(\mathbf{w})}\right) \\
&= -\frac{p(\mathbf{w})}{1 + G(\mathbf{w},\theta)} + \frac{\nu q(\mathbf{w})}{G(\mathbf{w})(1 + G(\mathbf{w}))} = 0 \\
\implies \frac{\nu q(\mathbf{w})}{G(\mathbf{w})(1 + G(\mathbf{w}))} &= \frac{p(\mathbf{w})}{1 + G(\mathbf{w},\theta)} \\
\implies \frac{\nu q(\mathbf{w})}{G(\mathbf{w})} &= p(\mathbf{w}) \\
\implies G(\mathbf{w}) &= \nu \frac{q(\mathbf{w})}{p(\mathbf{w})}
\end{aligned}$$

- let's take a break to discuss functional derivative for a normal **function** f:

- if $\mathbf{x}$ is a stationary point, then any slight perturbation of $\mathbf{x}$ must:
  - either increase $J(x)$ (if $\mathbf{x}$ is a minimizer) or
  - decrease $J(x)$ (if $\mathbf{x}$ is a maximizer)
- let $g_\varepsilon(\mathbf{x}) = \mathbf{x} + \varepsilon$ be result of such a perturbation, where ε is small, then define:

$$
\begin{aligned}
\left.\frac{\mathrm{d}J_\varepsilon}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0} &= \left(\left.\frac{\mathrm{d}\ J(g_\varepsilon(\mathbf{x}))}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0}\right) = \left(\left.\frac{\mathrm{d}\ J(g_\varepsilon(\mathbf{x}))}{\mathrm{d}\ g_\varepsilon(\mathbf{x})} \underbrace{\frac{\mathrm{d}\ g_\varepsilon(\mathbf{x})}{\mathrm{d}\varepsilon}}_{=1}\right|_{\varepsilon=0}\right) = \left.\frac{\mathrm{d}\ J(g_\varepsilon(\mathbf{x}))}{\mathrm{d}\ g_\varepsilon(\mathbf{x})}\right|_{\varepsilon=0} \\
&= \left.\frac{\mathrm{d}\ J(\mathbf{x}+\varepsilon)}{\mathrm{d}\ (\mathbf{x}+\varepsilon)}\right|_{\varepsilon=0} = 0 \\
\implies\ & J'(\mathbf{x}) = 0
\end{aligned}
$$

- showing $\left.\frac{\mathrm{d}J_\varepsilon}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0} = J'(\mathbf{x}) = 0$ above is obvious, and doesn't help anything;
- however, it does LOT for functional:

for a **functional** F:

- to find stationary function $\mathbf{f}$ of functional F, satisfy boundary condition $\mathbf{f}(a) = A, \mathbf{f}(b) = B$:

$$J = \int_a^b F\big(x, \mathbf{f}(x), \mathbf{f}'(x)\big)\, \mathrm{d}x$$

- slight perturbation of $\mathbf{f}$ that preserves boundary values must:
  - either increase J (if $\mathbf{f}$ is a minimizer) or
  - decrease J (if $\mathbf{f}$ is a maximizer)
- let $g_\varepsilon(x) = \mathbf{f}(x) + \varepsilon\eta(x)$ be result of such a perturbation $\varepsilon\eta(x)$ of $\mathbf{f}$, where ε is small and $\eta(x)$ is a differentiable function satisfying $\eta(a) = \eta(b) = 0$:

$$J_\varepsilon = \int_a^b \underbrace{F(x, g_\varepsilon(x), g'_\varepsilon(x))}_{F_\varepsilon}\, \mathrm{d}x$$

compute $\frac{dJ_\varepsilon}{d\varepsilon}\Big|_{\varepsilon=0}$ (1)

- $g_\varepsilon(x) = \mathbf{f}(x) + \varepsilon\eta(x) \implies g'_\varepsilon \equiv \frac{g_\varepsilon(x)}{\mathbf{dx}} = \mathbf{f}'(x) + \varepsilon\eta'(x) \implies \frac{dg'_\varepsilon}{d\varepsilon} = \eta'(x)$
- now calculate the total derivative of J_ε with respect to ε:

$$
\begin{aligned}
\frac{dJ_\varepsilon}{d\varepsilon} &= \frac{d}{d\varepsilon}\int_a^b F_\varepsilon \, dx = \int_a^b \frac{d F_\varepsilon}{d\varepsilon} \, dx \\
&= \int_a^b \left[\frac{\partial F_\varepsilon}{\partial x}\frac{dx}{d\varepsilon} + \frac{\partial F_\varepsilon}{\partial g_\varepsilon}\frac{dg_\varepsilon}{d\varepsilon} + \frac{\partial F_\varepsilon}{\partial g'_\varepsilon}\frac{dg'_\varepsilon}{d\varepsilon} \right] dx \\
&= \int_a^b \left[\frac{\partial F_\varepsilon}{\partial g_\varepsilon}\frac{dg_\varepsilon}{d\varepsilon} + \frac{\partial F_\varepsilon}{\partial g'_\varepsilon}\frac{dg'_\varepsilon}{d\varepsilon} \right] dx \qquad x \text{ is independent of } \varepsilon \\
&= \int_a^b \left[\frac{\partial F_\varepsilon}{\partial g_\varepsilon}\eta(x) + \frac{\partial F_\varepsilon}{\partial g'_\varepsilon}\eta'(x) \right] dx
\end{aligned}
$$

- when $\varepsilon = 0$:
  1. $g_\varepsilon = \mathbf{f}$
  2. $F_\varepsilon = F(x, \mathbf{f}(x), \mathbf{f}'(x))$ and
  3. J_ε has an extremum value

$$
\frac{dJ_\varepsilon}{d\varepsilon}\Big|_{\varepsilon=0} = \int_a^b \left[\frac{\partial F}{\partial \mathbf{f}}\eta(x) + \frac{\partial F}{\partial \mathbf{f}'}\eta'(x) \right] dx = 0
$$

compute $\frac{dJ_\varepsilon}{d\varepsilon}\Big|_{\varepsilon=0}$ (2)

$$\frac{dJ_\varepsilon}{d\varepsilon}\Big|_{\varepsilon=0} = \int_a^b \left[\eta(x)\frac{\partial F}{\partial \mathbf{f}} + \underbrace{\eta'(x)}_{v'}\underbrace{\frac{\partial F}{\partial \mathbf{f}'}}_{u}\right] dx = 0$$

- use integration by parts: $\int u\, v' = uv - \int v\, u'$ on second term:

$$\begin{aligned}
\frac{dJ_\varepsilon}{d\varepsilon}\Big|_{\varepsilon=0} &= \int_a^b \left[\eta(x)\frac{\partial F}{\partial \mathbf{f}}\right] + \underbrace{\int_a^b \left[\eta'(x)\frac{\partial F}{\partial \mathbf{f}'}\right] dx} \\
&= \int_a^b \left[\eta(x)\frac{\partial F}{\partial \mathbf{f}}\right] + \left[\eta(x)\frac{\partial F}{\partial \mathbf{f}'}\right]_a^b - \int_a^b \eta(x)\,\frac{d}{dx}\frac{\partial F}{\partial \mathbf{f}'}dx \\
&= \int_a^b \left[\frac{\partial F}{\partial \mathbf{f}} - \frac{d}{dx}\frac{\partial F}{\partial \mathbf{f}'}\right]\eta(x)\,dx + \left[\eta(x)\frac{\partial F}{\partial \mathbf{f}'}\right]_a^b = 0
\end{aligned}$$

- using the boundary conditions $\eta(a) = \eta(b) = 0$:

$$\int_a^b \left[\frac{\partial F}{\partial \mathbf{f}} - \frac{d}{dx}\frac{\partial F}{\partial \mathbf{f}'}\right]\eta(x)\,dx = 0$$

Euler-Lagrange Equation

- **Fundamental lemma of calculus of variations says**:
  if a continuous function f on an open interval (a, b) satisfies equality:

$$\int_a^b f(x)h(x)\,\mathrm{d}x = 0 \implies f(x) = 0$$

- then,

$$\int_a^b \left[\frac{\partial F}{\partial \mathbf{f}} - \frac{\mathrm{d}}{\mathrm{d}x} \frac{\partial F}{\partial \mathbf{f}'} \right] \eta(x)\,\mathrm{d}x = 0$$
$$\implies \frac{\partial F}{\partial \mathbf{f}} - \frac{\mathrm{d}}{\mathrm{d}x} \frac{\partial F}{\partial \mathbf{f}'} = 0$$

- back to our example, $\mathcal{C}$ contains no $G'(\mathbf{w}, \theta)$ terms, therefore, we only need to show: $\frac{\delta \mathcal{C}(G)}{\delta G} = 0$

Probability density re-parameterization

Score Function Estimator

- we love to have integral in a form:

$$\mathcal{I} = \int_z f(z)p(z)\mathrm{d}z \equiv \mathbb{E}_{z\sim p(z)}[f(z)]$$

as we can approximate the **expectation** with:

$$\mathcal{I} \approx \frac{1}{N}\sum_{i=1}^{N} f(z^{(i)}) \qquad\qquad z^{(i)} \sim p(z)$$

- we do **not** love $\int_x f(z)\nabla_\theta p(z|\theta)\mathrm{d}z$,
- in general, $\nabla_\theta p(z|\theta)$ is **not** a probability, e.g., look at derivative of a Gaussian distribution:

$$\frac{\partial}{\partial\mu}\left(\frac{\exp^{-(z-\mu)^2/\sigma^2}}{\sqrt{2\pi}\sigma}\right) = \frac{2(z-\mu)}{\sigma^2}\frac{\exp^{-(z-\mu)^2/\sigma^2}}{\sqrt{2\pi}\sigma}$$

Score Function Estimator

- however, in machine learning, we have to deal with:

$$\nabla_\theta \left[\int_z f(z)p(z|\theta)\mathrm{d}z \right] = \int_z \nabla_\theta \left[f(z)p(z|\theta) \right] \mathrm{d}z = \int_z f(z) \left[\nabla_\theta p(z|\theta) \right] \mathrm{d}z$$

- i.e, θ is the parameter of the distribution
- e.g., in **Reinforcement Learning**: let $\Pi \equiv \{s_1, a_1, \dots, s_T, a_T\}$

$$p_\theta(\Pi) \equiv p_\theta(s_1, a_1, \dots s_T, a_T) = p(s_1) \prod_{t=1}^{T} \pi_\theta(a_t|s_t) p(s_{t+1}|s_t, a_t)$$

$$\implies \theta^* = \arg\max_{\theta} \left\{ \mathbb{E}_{\Pi \sim p_\theta(\Pi)} \underbrace{\left[\sum_{t=1}^{T} R(s_t, a_t) \right]}_{f(z)} \right\}$$

Score Function Estimator

- we use **REINFORCE trick**, with the follow property:

$$p(z|\theta)f(z)\nabla_\theta[\log p(z|\theta)] = p(z|\theta)f(z)\frac{\nabla_\theta p(z|\theta)}{p(z|\theta)} = f(z)\nabla_\theta p(z|\theta)$$

- looking at the original integral:

$$\begin{aligned}\int_z f(z)\nabla_\theta p(z|\theta)\mathrm{d}z &= \int_z p(z|\theta)f(z)\nabla_\theta[\log p(z|\theta)]\mathrm{d}z \\ &= \mathbb{E}_{z\sim p(z|\theta)}\Big[f(z)\nabla_\theta[\log p(z|\theta)]\Big]\end{aligned}$$

- can approximated by:

$$\frac{1}{N}\sum_{i=1}^{N} f(z^{(i)})\nabla_\theta[\log p(z^{(i)}|\theta)] \qquad z^{(i)} \sim p(z|\theta)$$

- suffers from **high variance** and is slow to converge

- we let $z = g(x)$:

$$
\begin{aligned}
\mathbb{E}_{x \sim p(x)}[g(x)] &= \mathbb{E}_{z \sim p(z)}[z] \\
\mathbb{E}_{x \sim p(x)}[g(x, \theta)] &= \mathbb{E}_{z \sim p_\theta(z)}[z] \qquad \text{paramterize the distribution with } \theta \\
\mathbb{E}_{x \sim p(x)}[f(g(x, \theta))] &= \mathbb{E}_{z \sim p_\theta(z)}[f(z)] \qquad \text{introduce function } f(.) \\
\int_{x \in \Omega_x} f(g(x, \theta)) {\color{red} p(x) \mathrm{d}x} &= \int_{z \in \Omega_z} f(z) {\color{blue} p_\theta(z) \mathrm{d}z}
\end{aligned}
$$

- only need to know deterministic function $z = g(x, \theta)$ and distribution $p(x)$
- does **not** need to explicitly know distribution of z
- e.g., Gaussian variable: $z \sim \mathcal{N}(z; \mu(\theta), \sigma(\theta))$ can be rewritten as a function of a standard Gaussian variable:

$$
z = g(x, \theta) = \underbrace{\mu(\theta) + x\sigma(\theta)}_{g(x,\theta)} \qquad \text{can be re-parameterised into} \qquad x \sim \underbrace{\mathcal{N}(0, 1)}_{p(x)}
$$

revision on change of variable

- Let $y = T(x) \implies x = T^{-1}(y)$:

$$F_Y(y) = \Pr(T(X) \leq y) = \Pr(X \leq T^{-1}(y)) = F_X\left(T^{-1}(y)\right) = F_X(x)$$

$$f_Y(y) = \frac{\mathrm{d}F_Y(y)}{\mathrm{d}y} = \frac{\mathrm{d}F_X(x)}{\mathrm{d}y} = \frac{\mathrm{d}F_X(x)}{\mathrm{d}x}\frac{\mathrm{d}x}{\mathrm{d}y} = f_x\left(x\right)\frac{\mathrm{d}x}{\mathrm{d}y}$$

- without change of limits

$$f_Y(y)\left|\mathrm{d}y\right| = f_x\left(x\right)\left|\mathrm{d}x\right|$$

- with change of limits

$$f_Y(y)\mathrm{d}y = f_x\left(x\right)\mathrm{d}x$$

re-parameterization trick (2)

- **main motivation** $p(x)$ is **no longer** parameterized by θ:

$$
\begin{aligned}
\mathbb{E}_{x\sim p(x)}[f(g(x,\theta))] &= \int_x f(g(x,\theta))p(x)\mathrm{d}x \\
\Longrightarrow \quad \frac{\partial}{\partial\theta}\mathbb{E}_{x\sim p(x)}[f(g(x,\theta))] &= \frac{\partial}{\partial\theta}\int_x f(g(x,\theta))p(x)\mathrm{d}x \\
&= \int_x \left[\frac{\partial}{\partial\theta} f(g(x,\theta))\right] p(x)\mathrm{d}x \\
&\approx \frac{1}{N}\sum_{i=1}^{N}\frac{\partial}{\partial\theta} f(g(x^{(i)},\theta)) \qquad x \sim p(x) \\
&= \frac{1}{N}\sum_{i=1}^{N}\nabla_\theta f(g(x^{(i)},\theta)) \qquad \text{use shorthand notation: } \nabla_\theta[\cdot] \equiv \frac{\partial}{\partial\theta}[\cdot]
\end{aligned}
$$

- during gradient decent, x are sampled independent of θ

Simple example

- let $\mu(\theta) = a\theta + b$, and $\sigma(\theta) = 1$, and we would like to compute:

$$
\begin{aligned}
\theta^* &= \arg\max_\theta [F(\theta)] \\
&= \arg\min \mathbb{E}_{z \sim \mathcal{N}(\mu(\theta), \sigma(\theta))}[z^2] \\
&= \arg\min_\theta \left[\int_z \underbrace{z^2}_{f(z)} \mathcal{N}\Big(\underbrace{a\theta + b}_{\mu(\theta)}, \underbrace{1}_{\sigma(\theta))} \Big) \right]
\end{aligned}
$$

- we can solve it by imagine its diagram ...
- in words, it says: find mean of Gaussian, so that the "expected square of samples" from this Gaussian are minimized;
- it's obvious that you want to move μ to close to **zero** as possible
- which implies $\theta = -\frac{b}{a} \implies \mu(\theta) = 0$
- without using any tricks, the gradient is computed by:

$$
\nabla_\theta F(\theta) = \int_z \underbrace{z^2}_{f(z)} \times \underbrace{\frac{2(z-\mu)}{\sigma^2} \frac{\exp^{-(z-\mu)^2/\sigma^2}}{\sqrt{2\pi}\sigma}}_{\frac{\partial \mathcal{N}(\mu, \sigma^2)}{\partial \mu}} \times \underbrace{a}_{\frac{\partial \mu}{\partial \theta}} \mathrm{d}z
$$

- very hard!

solve it using **REINFORCE trick**

- let's solve it by gradient descend by **REINFORCE**:
- let $\mu(\theta) = a\theta + b$, and $\sigma(\theta) = 1$:

$$
\begin{aligned}
\int_z f(z)\nabla_\theta p(z|\theta)\mathrm{d}z &= \mathbb{E}_{z\sim p(z|\theta)}\big[f(z)\nabla_\theta[\log p(z|\theta)]\big] \\
&= \mathbb{E}_{z\sim p(z|\theta)}\left[z^2\nabla_\theta \log\left(\frac{1}{\sigma\sqrt{2\pi}}\exp^{-\frac{(z-\mu)^2}{2\sigma^2}}\right)\right] \\
&= \mathbb{E}_{z\sim p(z|\theta)}\left[z^2\nabla\mu\left[-\log(\sqrt{2\pi}\sigma) - \frac{(z-\mu)^2}{2\sigma^2}\right]\times\frac{\partial\mu(\theta)}{\theta}\right] \\
&= \mathbb{E}_{z\sim\mathcal{N}\left(z;a\theta+b,1\right)}\left[z^2(z-\mu(\theta))\times a\right] \qquad \text{let } \sigma = 1 \\
&= \mathbb{E}_{z\sim\mathcal{N}\left(z;a\theta+b,1\right)}\left[z^2 a(z-a\theta-b)\right]
\end{aligned}
$$

solve it using **re-parameterization trick:**

- $z \sim \mathcal{N}\big(z; \mu(\theta), \sigma(\theta)\big)$ can be **re-parameterised** into:
- if we need to compute: $f(z) = z^2$

$$
\begin{aligned}
x &\sim \mathcal{N}(0, 1) \\
z &\equiv g(x, \theta) = \mu(\theta) + x\sigma(\theta)
\end{aligned}
$$

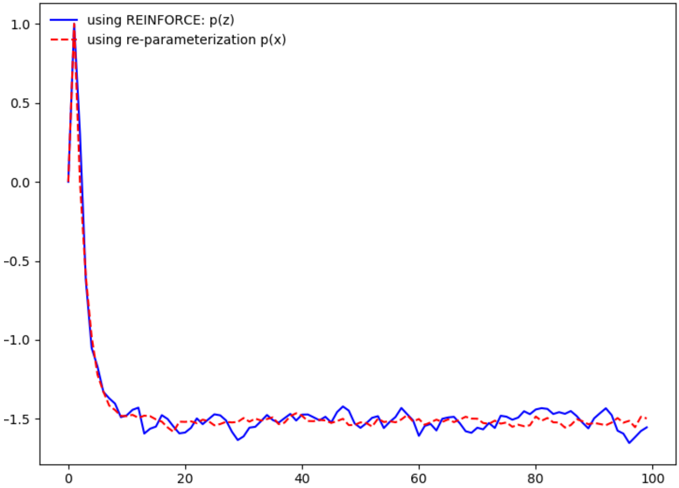
- the re-parameterised version is:

$$
\begin{aligned}
\nabla_\theta \mathbb{E}_{x \sim p(x)}[f(g(x, \theta))] &\equiv \mathbb{E}_{x \sim \mathcal{N}\left(x;0,1\right)} \left[\nabla_\theta \left(z^2\right)\right] \\
&= \mathbb{E}_{x \sim \mathcal{N}\left(x;0,1\right)} \left[\nabla_\theta \left(\mu(\theta) + x\sigma(\theta)\right)^2\right] \\
&= \mathbb{E}_{x \sim \mathcal{N}\left(x;0,1\right)} \left[\nabla_\theta \left(a\theta + b + x\right)^2\right] \\
&= \mathbb{E}_{x \sim \mathcal{N}\left(x;0,1\right)} \left[2a(a\theta + b + x)\right]
\end{aligned}
$$

- both REINFORCE and re-parameterization must achieve the same result!
- knowing $p(X)$ and $g(x, \theta)$ is sufficient, we do **not** need to know explicitly $p(Z)$

results

- compare both methods using $a = 2$, $b = 3$:



other examples: Evidence Lower Bound (ELOB)

- ELOB:

$$
\begin{aligned}
\mathcal{L}_{\phi,\theta} &= \int q(z) \ln(p(\mathbf{y}, z))\mathrm{d}Z - \int q(z) \ln(q(z))\mathrm{d}z \\
&= \int q_\phi(z) \ln(p_\theta(\mathbf{y}, z))\mathrm{d}z - \int q_\phi(z) \ln(q_\phi(z))\mathrm{d}z \qquad \text{parameterize} \\
&= \mathbb{E}_{q_\phi(z)}\big[\ln(p_\theta(\mathbf{y}, z))\big] - \mathbb{E}_{q_\phi(z)}\big[\ln(q_\phi(z))\big]
\end{aligned}
$$

- after re-parameterization, it appears to be:

$$
\mathcal{L}_{\phi,\theta} = \mathbb{E}_{x\sim p(x)}\big[\log(p_\theta(\mathbf{y}, g(\phi, x))) - \log(q_\phi(g(\phi, x)))\big]
$$

Log-likelihood and Evidence Lower Bound (ELOB)

- It is universally true that:

$$\ln\left(p(\mathbf{y})\right) = \ln\left(p(\mathbf{y}, z)\right) - \ln\left(p(z|\mathbf{y})\right)$$

- It's also true (a bit silly) that:

$$\ln\left(p(\mathbf{y})\right) = \left[\ln(p(\mathbf{y}, z)) - \ln(q(z))\right] - \left[\ln(p(z|\mathbf{y})) - \ln(q(z))\right]$$

- The above is so that we can insert an arbitrary pdf $q(z)$ into, now we get:

$$\ln\left(p(\mathbf{y})\right) = \ln\left(\frac{p(\mathbf{y}, z)}{q(z)}\right) - \ln\left(\frac{p(z|\mathbf{y})}{q(z)}\right)$$

- Taking the expectation on both sides, given $q(z)$:

$$\begin{aligned}
\ln\left(p(\mathbf{y})\right) &= \int q(z)\ln\left(\frac{p(\mathbf{y}, z)}{q(z)}\right)\mathrm{d}z - \int q(z)\ln\left(\frac{p(z|\mathbf{y})}{q(z)}\right)\mathrm{d}z \\
&= \underbrace{\int q(z)\ln(p(\mathbf{y}, z))\mathrm{d}Z - \int q(z)\ln(q(z))\mathrm{d}z}_{\mathcal{L}(q)} + \underbrace{\left(-\int q(z)\ln\left(\frac{p(z|\mathbf{y})}{q(z)}\right)\mathrm{d}z\right)}_{\mathrm{KL}(q\|p)} \\
&= \mathcal{L}(q) + \mathrm{KL}(q\|p)
\end{aligned}$$

firstly, what is an auto-encoder:

- **encoder** $x \rightarrow z$
- **decoder** $z \rightarrow x'$, such you want x and x' to be as close as possible
- autoencoders generate things “as it is”

would be better, if we could feed z to **decoder** that **were not** encoded from the images in actual dataset

- then, we can synthesis new, reasonable data
- an idea: when feed database of images $\{x\}$ to encoder, the corresponding $\{z\}$ are “forced into” to form a distribution, so that a **new** sample z' randomly drawn from this distribution creates a reasonable data

- loss at a particular data point x_i:

$$\mathcal{L}_i(\theta, \phi) = \underbrace{-\mathbb{E}_{z \sim Q_\theta(z|x_i)}\left[\log P_\phi(x_i|z)\right]}_{\text{reconstruction error}} + \underbrace{\text{KL}(Q_\theta(z|x_i)||p(z))}_{\text{regularizer}}$$

- we want $\mathbb{E}_{z \sim Q_\theta(z|x_i)}\left[\log P_\phi(x_i|z)\right]$ to be high, it needs for:
- $Q_\theta(z|x_i) \uparrow \implies P_\phi(x_i|z) \uparrow$ and $Q_\theta(z|x_i) \downarrow \implies P_\phi(x_i|z) \downarrow$
- therefore, the optimal solution may be for $Q_\theta(z|x_i)$ and $P_\phi(x_i|z)$ to be just a single delta function in a $x - z$ plane
- and all rest of $\{x, z\}$ are delta functions lies on a monotonic curve on the $x - z$ plane
- regularizer $\text{KL}(Q_\theta(z|x_i)||P(z))$ ensure $Q_\theta(z|x_i)$ doesn't behalf the above, i.e., $Q_\theta(z|x_i)$ are distributed as close to Gaussian distribution as possible
- $P_\phi(x_i|z)$ is just supervised learning: pixel value x_i is its label/value

look at the ELBO again

- we are not choosing our normal ELBO to maximize:

$$\ln\left(p(\mathbf{y})\right) = \underbrace{\int q(z)\ln(p(\mathbf{y},z))\mathrm{d}z - \int q(z)\ln(q(z))\mathrm{d}z}_{\mathcal{L}(q)} + \underbrace{\left(-\int q(z)\ln\left(\frac{p(z|\mathbf{y})}{q(z)}\right)\mathrm{d}z\right)}_{\mathrm{KL}(q\|p)}$$

$$q(z) \to q(z|\mathbf{y})$$

$$= \int q(z|\mathbf{y})\ln(p(z,\mathbf{y}))\mathrm{d}z - \int q(z|\mathbf{y})\ln(q(z|\mathbf{y}))\mathrm{d}z + \left(-\int q(z|\mathbf{y})\ln\left(\frac{p(z|\mathbf{y})}{q(z|\mathbf{y})}\right)\mathrm{d}z\right)$$

$$= \int q(z|\mathbf{y})\ln(p(\mathbf{y}|z))\mathrm{d}z + \int q(z|\mathbf{y})\ln(p(z))\mathrm{d}z - \int q(z|\mathbf{y})\ln(q(z|\mathbf{y}))\mathrm{d}z + \mathrm{KL}\left(q(z|\mathbf{y})\|p(z|\mathbf{y})\right)$$

$$= \int q(z|\mathbf{y})\ln(p(\mathbf{y}|z))\mathrm{d}z + \int q(z|\mathbf{y})\ln(p(z))\mathrm{d}z - \int q(z|\mathbf{y})\ln(q(z|\mathbf{y}))\mathrm{d}z + \mathrm{KL}\left(q(z|\mathbf{y})\|p(z|\mathbf{y})\right)$$

$$= \int q(z|\mathbf{y})\ln(p(\mathbf{y}|z))\mathrm{d}z - \mathrm{KL}\left(q(z|\mathbf{y})\|p(z)\right) + \mathrm{KL}\left(q(z|\mathbf{y})\|p(z|\mathbf{y})\right)$$

- therefore,

$$\ln\left(p(\mathbf{y})\right) - \mathrm{KL}\left(q(z|\mathbf{y})\|p(z|\mathbf{y})\right) = \int q(z|\mathbf{y})\ln(p(\mathbf{y}|z))\mathrm{d}z - \mathrm{KL}\left(q(z|\mathbf{y})\|p(z)\right)$$

$$= \underbrace{\mathbb{E}_{z\sim q(z|\mathbf{y})}\left[\ln(p(\mathbf{y}|z))\right] - \mathrm{KL}\left(q(z|\mathbf{y})\|p(z)\right)}_{1\ \mathcal{L}}$$

- by minimizing 1 $\mathcal{L} \implies q(z|\mathbf{y}) \to p(z|\mathbf{y}) \implies \ln\left(p(\mathbf{y})\right)$ is maximized

real example on variational auto-encoder

- knowing

$$\ln\left(p(\mathbf{y})\right) - \mathrm{KL}\big(q(z|\mathbf{y})\|p(z|\mathbf{y})\big) = \underbrace{\mathbb{E}_{z\sim q(z|\mathbf{y})}\big[\ln(p(\mathbf{y}|z))\big] - \mathrm{KL}\big(q(z|\mathbf{y})\|p(z)\big)}_{\mathcal{L}(\cdot)}$$

- our aim is if we do:

$$Z_i \sim q_\theta(z|\mathbf{y}_i) \qquad \mathcal{Y}_i \sim p_\phi(\mathcal{Y}|Z_i)$$

  we want to $\mathcal{Y}_i$ to resemble $\mathbf{y}_i$ with high probability
- in VAE, loss at each data point:

$$\mathcal{L}_i(\theta,\phi) = \underbrace{-\mathbb{E}_{z\sim q_\theta(z|\mathbf{y}_i)}\big[\log p_\phi(\mathbf{y}_i|z)\big]}_{\text{reconstruction loss}} + \underbrace{\mathrm{KL}(q_\theta(z\|\mathbf{y}_i)\|p(z))}_{\text{regularizer}}$$

objective function illustration

new intepretation:

- loss at loss function again:

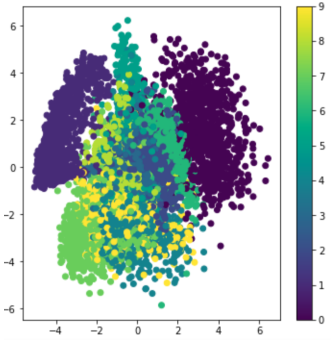
$$\mathcal{L}_i(\theta,\phi) = \underbrace{-\mathbb{E}_{z\sim q_\theta(z|\mathbf{y}_i)}\Big[\log p_\phi(\mathbf{y}_i|z)\Big]}_{\text{reconstruction loss}} + \underbrace{\mathrm{KL}(q_\theta(z\|\mathbf{y}_i)\|p(z))}_{\text{regularizer}}$$

- without reconstruction loss, same numbers may not be close together, i.e., they spread across the entire multivariate normal distribution, when we perform:

$$Z_i \sim q_\theta(z|\mathbf{y}_i) \qquad \mathcal{Y}_i \sim p_\phi(\mathcal{Y}|Z_i)$$

  i.e., $\mathcal{Y}_i$ has low probability to look like $\mathbf{y}_i$

- without regularizer, you may recover digits back, but they don't form overall multivariate Gaussian distribution (so you can't sample)



https://towardsdatascience.com/variational-auto-encoders-fc701b9fc569

KL between two Gaussian distributions

- compute $\text{KL}\left(\mathcal{N}(\mu_1, \Sigma_1) \| \mathcal{N}(\mu_2, \Sigma_2)\right)$

$$
\begin{aligned}
\text{KL} &= \int_x \left[\frac{1}{2} \log \frac{|\Sigma_2|}{|\Sigma_1|} - \frac{1}{2}(x - \mu_1)^T \Sigma_1^{-1}(x - \mu_1) + \frac{1}{2}(x - \mu_2)^T \Sigma_2^{-1}(x - \mu_2) \right] \times p(x) \mathrm{d}x \\
&= \frac{1}{2} \log \frac{|\Sigma_2|}{|\Sigma_1|} - \frac{1}{2} \text{tr} \left\{ \mathbb{E}[(x - \mu_1)(x - \mu_1)^T] \, \Sigma_1^{-1} \right\} + \frac{1}{2} \mathbb{E}[(x - \mu_2)^T \Sigma_2^{-1}(x - \mu_2)] \\
&= \frac{1}{2} \log \frac{|\Sigma_2|}{|\Sigma_1|} - \frac{1}{2} \text{tr} \, \{I_d\} + \frac{1}{2}(\mu_1 - \mu_2)^T \Sigma_2^{-1}(\mu_1 - \mu_2) + \frac{1}{2} \text{tr}\{\Sigma_2^{-1}\Sigma_1\} \\
&= \frac{1}{2} \left[\log \frac{|\Sigma_2|}{|\Sigma_1|} - d + \text{tr}\{\Sigma_2^{-1}\Sigma_1\} + (\mu_2 - \mu_1)^T \Sigma_2^{-1}(\mu_2 - \mu_1) \right]
\end{aligned}
$$

- substitute $\mu_2 = 1$ for each dimension, $\Sigma_2 = \text{I}$ is a Σ_2 is a diagonal matrix:

$$
\begin{aligned}
\text{KL}[N(\mu(X), \Sigma(X)) \| N(0, 1)] &= \frac{1}{2} \left(\text{tr}(\Sigma(X)) + \mu(X)^T \mu(X) - k - \log \, \det(\Sigma(X)) \right) \\
&= \frac{1}{2} \left(\sum_k \sigma_k^2 + \sum_k \mu_k^2 - \sum_k 1 - \log \prod_k \sigma_k^2 \right) \\
&= \frac{1}{2} \sum_k \left(\sigma_k^2 + \mu_k^2 - 1 - \log \, \sigma_k^2 \right)
\end{aligned}
$$

there is an even simpler way to compute KL, when $p(x, y) = p(x)p(y)$ and $q(x, y) = q(x)q(y)$

- let

$$
\begin{aligned}
&\text{KL}(p, q) = -\left(\int p(x) \log q(x) \mathrm{d}x - \int p(x) \log p(x) \mathrm{d}x \right) \\
&\implies \text{KL}(p(x)p(y), q(x)q(y)) \\
&= -\left(\int_x \int_y p(x)p(y) \big[\log q(x) + \log q(y) \big] \mathrm{d}x - p(x)p(y) \big[\log p(x) + \log p(y) \big] \mathrm{d}x \right) \\
&= -\left(\int_x \int_y \big[p(x)p(y) \log q(x) + p(x)p(y) \log q(y) - p(x)p(y) \log p(x) - p(x)p(y) \log p(y) \big] \mathrm{d}x \right) \\
&= -\left(\int_x \int_y p(x)p(y) \log q(x) + \int_x \int_y p(x)p(y) \log q(y) - \int_x \int_y p(x)p(y) \log p(x) - \int_x \int_y p(x)p(y) \log p(y) \mathrm{d}x \right) \\
&= -\left(\int_x p(x) \log q(x) \int_y p(y) + \int_x p(x) \int_y p(y) \log q(y) - \int_x p(x) \log p(x) \int_y p(y) - \int_x p(x) \int_y p(y) \log p(y) \right) \\
&= -\left(\int_x p(x) \log q(x) + \int_y p(y) \log q(y) - \int_x p(x) \log p(x) - \int_y p(y) \log p(y) \right) \\
&= -\left(\int_x p(x) \log q(x) - \int_x p(x) \log p(x) \right) - \left(\int_y p(y) \log q(y) - \int_y p(y) \log p(y) \right) \\
&= \text{KL}(p(x) \| q(x)) + \text{KL}(p(y) \| q(y))
\end{aligned}
$$

there is an even simpler way to compute KL, when $p(x, y) = p(x)p(y)$ and $q(x, y) = q(x)q(y)$

- let $p(x) = \mathcal{N}(\mu_p, \sigma_p)$ and $q(x) = \mathcal{N}(\mu_q, \sigma_q)$:

$$
\begin{aligned}
KL(p, q) &= -\int p(x)\log q(x)\mathrm{d}x + \int p(x)\log p(x)\mathrm{d}x \\
&= \frac{1}{2}\log(2\pi\sigma_q^2) + \frac{\sigma_p^2 + (\mu_p - \mu_q)^2}{2\sigma_q^2} - \frac{1}{2}(1 + \log 2\pi\sigma_p^2) \\
&= \log\frac{\sigma_q}{\sigma_p} + \frac{\sigma_p^2 + (\mu_p - \mu_q)^2}{2\sigma_q^2} - \frac{1}{2} \\
&= \log\sigma_q - \log\sigma_p + \frac{\sigma_p^2}{2\sigma_q^2} + \frac{(\mu_p - \mu_q)^2}{2\sigma_q^2} - \frac{1}{2}
\end{aligned}
$$

- let $q(x) = \mathcal{N}(0, 1)$:

$$
\begin{aligned}
\mathrm{KL}(p, q) &= \frac{\sigma_p^2}{2} + \frac{\mu_p^2}{2} - \frac{1}{2} - \log\sigma_p \\
&= \frac{1}{2}\left[\frac{\sigma_p^2}{2} + \frac{\mu_p^2}{2} - \frac{1}{2} - \log\sigma_p^2\right]
\end{aligned}
$$

- $P(X) = \prod_k p(x_k)$ and $Q(X) = \prod_k q(x_k)$:

where does neural network come in to play?

- to do Bayesian properly, we need:

$$P(z|x_i) \propto \underbrace{P_\theta(x_i|z)}_{\text{Encoder network}} \underbrace{P(z)}_{\mathcal{N}(0,I)}$$

- this is certainly not Gaussian! therefore, we need to use variational approach, and to define $Q_\theta(z|x_i) \equiv \mathcal{N}\big(\mu(x_i,\theta), \Sigma(x_i,\theta)\big)$
- we can choose any distribution, but having Normal distribution making KL computation a lot easier in objective function
- how do we obtain the parameter value of this Gaussian?
- of course a linear, or a kernel won't do its trick, we need a Neural Network for both $\mu(x_i,\theta), \Sigma(x_i,\theta)$

apply re-parameterization trick to softmax

- when we have the following

$$
\begin{aligned}
\mathbb{E}_{K \sim \text{softmax}(\mu_1(\theta), \dots, \mu_L(\theta))}[f(\mathbf{v}(K))] &= \sum_{k=1}^{L} f(\mathbf{v}(k)) \Pr(k|\theta) \\
&\equiv \sum_{k=1}^{L} f(\mathbf{v}(k)) \big(\text{softmax}(\mu_1(\theta), \dots, \mu_L(\theta))\big)_k
\end{aligned}
$$

- can we find their corresponding:

$$
\mathcal{K} = g(\mathcal{G}, \theta) \qquad \mathcal{G} \sim p(\mathcal{G})
$$

Re-parameterization using Gumbel-max trick

- Gumbel-max trick also means:

$$U \sim \underbrace{\mathcal{U}(0,1) \qquad \mathcal{G} = -\log(-\log(U))}_{p(\mathcal{G})}$$

$$k = \underbrace{\underset{i \in \{1,\dots,K\}}{\arg\max} \; \{\mu_1(\theta) + \mathcal{G}, \dots, \mu_K(\theta) + \mathcal{G}\}}_{g(\mathcal{G},\theta)} \qquad \mathbf{v} = \text{one-hot}(k)$$

- this is a form of re-paramterization:

  instead of sample $\mathcal{K} \sim \text{softmax}(\mu_1(\theta), \dots, \mu_K(\theta))$, we i.i.d. sample $\mathcal{G}$ instead

- well, there is two problems, firstly **why is such true**?

Gumbel-max trick and Softmax (1)

- pdf of Gumbel with **unit scale** and location parameter μ:

$$\text{gumbel}(Z = z\,;\ \mu) = \exp\left[-(z-\mu) - \exp\{-(z-\mu)\}\right]$$

- CDF of Gumbel:

$$\text{Gumbel}(Z \leq z\,;\ \mu) = \exp\left[-\exp\{-(z-\mu)\}\right]$$

Gumbel-max trick and Softmax (1)

- given a set of Gumbel random variables $\{Z_i\}$, each having own location parameters $\{\mu_i\}$, probability of all other $Z_{i\neq k}$ are less than a particular value of z_k:

$$p\left(\max\{Z_{i\neq k}\} = z_k\right) = \prod_{i\neq k} \exp\left[-\exp\{-(z_k - \mu_i)\}\right]$$

- obviously, $Z_k \sim \text{gumbel}(Z_k = z_k; \mu_k)$:

$$\begin{aligned}
&\Pr(k \text{ is largest} \mid \{\mu_i\}) \\
&= \int \exp\left\{-(z_k - \mu_k) - \exp\{-(z_k - \mu_k)\}\right\} \prod_{i\neq k} \exp\left\{-\exp\{-(z_k - \mu_i)\}\right\} \mathrm{d}z_k \\
&= \int \exp\left[-z_k + \mu_k - \exp\{-(z_k - \mu_k)\}\right] \exp\left[-\sum_{i\neq k} \exp\{-(z_k - \mu_i)\}\right] \mathrm{d}z_k \\
&= \int \exp\left[-z_k + \mu_k - \exp\{-(z_k - \mu_k)\} - \sum_{i\neq k} \exp\{-(z_k - \mu_i)\}\right] \mathrm{d}z_k \\
&= \int \exp\left[-z_k + \mu_k - \sum_{i} \exp\{-(z_k - \mu_i)\}\right] \mathrm{d}z_k \\
&= \int \exp\left[-z_k + \mu_k - \sum_{i} \exp\{-z_k + \mu_i)\}\right] \mathrm{d}z_k \\
&= \int \exp\left[-z_k + \mu_k - \exp\{-z_k\} \sum_{i} \exp\{\mu_i)\}\right] \mathrm{d}z_k
\end{aligned}$$

Gumbel-max trick and Softmax (2)

- keep on going:

$$
\begin{aligned}
\Pr(k \text{ is largest} \mid \{\mu_i\}) &= \int \exp\left[- z_k + \mu_k - \exp\{-z_k\} \sum_i \exp\{\mu_i)\} \right] \mathrm{d}z_k \\
&= \exp^{\mu_k} \int \exp\left[- z_k - \exp\{-z_k\} C \right] \mathrm{d}z_k \\
&= \exp^{\mu_k} \left[\frac{\exp(-C \exp(-z_k))}{C} \Big|_{z_k=-\infty}^{\infty} \right] \\
&= \exp^{\mu_k} \left[\frac{1}{C} - 0 \right] = \frac{\exp^{\mu_k}}{\sum_i \exp\{\mu_i)\}}
\end{aligned}
$$

Gumbel-max trick and Softmax (2)

- moral of the story is, if one is to sample the largest element from **softmax**:

$$K \sim \left\{ \frac{\exp(\mu_1)}{\sum_i \exp(\mu_i)}, \dots, \frac{\exp(\mu_L)}{\sum_i \exp(\mu_i)} \right\}$$

$$\implies K = \underset{i \in \{1,\dots,L\}}{\arg\max} \{G_1, \dots, G_L\}$$

$$\text{where } \underbrace{G_i \sim \text{gumbel}(z\,;\,\mu_i) \equiv \exp\Big[-(z-\mu_i) - \exp\{-(z-\mu_i)\}\Big]}$$

$$\implies K = \underset{i \in \{1,\dots,L\}}{\arg\max} \{\mu_1 + \mathcal{G}, \dots, \mu_L + \mathcal{G}\}$$

$$\text{where } \underbrace{\mathcal{G} \overset{\text{iid}}{\sim} \text{gumbel}(z\,;\,0) \equiv \exp\Big[-(z) - \exp\{-(z)\}\Big]}$$

- what is μ_i? for example,
  - $\mu_i \equiv \mathbf{x}^\top \theta_i$ in classification
  - $\mu_i \equiv \mathbf{u}_i^\top \mathbf{v}_c$ for word vectors
- some literature writes it as :

$$\equiv \underset{i \in \{1,\dots,L\}}{\arg\max} \{\log(\mu_1) + \mathcal{G}, \dots, \log(\mu_L) + \mathcal{G}\}$$

meaning, they let $\mu_i \equiv \exp(\mathbf{x}^\top \theta_i)$

how to sample a Gumbel?

- CDF of a Gumbel:

$$
\begin{aligned}
u &= \exp^{-\exp^{-(x-\mu)/\beta}} \\
\implies \log(u) &= -\exp^{-(x-\mu)/\beta} \\
\implies \log(-\log(u)) &= -(x-\mu)/\beta \\
\implies -\beta\log(-\log(u)) &= x-\mu \\
\implies x &= \text{CDF}^{-1}(u) \equiv \mu - \beta\log(-\log(u))
\end{aligned}
$$

- for standard Gumbel, i.e., $\mu = 0, \beta = 1$:

$$
x = \text{CDF}^{-1}(u) \equiv -\log(-\log(u))
$$

- therefore, sampling strategy:

$$
\begin{aligned}
U &\sim \mathcal{U}(0,1) \\
\mathcal{G} &= -\log(-\log(U)) \\
K &= \underset{i\in\{1,\dots,K\}}{\arg\max} \{\mu_1 + \mathcal{G}, \dots, \mu_L + \mathcal{G}\} \\
\mathbf{v} &= \text{one-hot}(K)
\end{aligned}
$$

Second problem with Softmax re-parameterisation

- the other remaining **problem**: sample $\mathbf{v}$ also has an $\arg\max$ operation, it's a discrete distribution!
- one can **relax** the softmax distribution, for example **softmax map**
- several solutions proposed, for example:
  "*Maddison, Mnih, and Teh (2017),The Concrete Distribution: a Continuous Relaxation of Discrete Random Variables*"

Relax the Softmax

- **softmax map**

$$f_\tau(x)_k = \frac{\exp(\mu_k/\tau)}{\sum_{k=1}^K \exp(\mu_k/\tau)} \qquad\qquad \mu_k \equiv \mu_k(x_k)$$

$$\text{as} \quad \tau \to 0 \implies f_\tau(x) = \max\left(\left\{ \frac{\exp(\mu_k)}{\sum_{k=1}^K \exp(\mu_k)} \right\}_{k=1}^K \right)$$

- questions can you also think about the relationship between Gaussian Mixture Model and K-means?
- one can say $\tau = 1$ is softmax, and $\tau = 0$ is hard-max!
- then we can apply the same softmax map with added Gumbel variables:

$$(X_k^\tau)_k = f_\tau(\mu + G)_k = \left(\frac{\exp(\mu_k + G_k)/\tau)}{\sum_{i=1}^K \exp(\mu_i + G_i)/\tau)} \right)_k$$