Some new and interesting research using Softmax

A/Prof Richard Yi Da Xu
richardxu.com

University of Technology Sydney (UTS)

December 24, 2019

Content

References

This lecture is referenced heavily from:

- Maddison, et. al (2017). “The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables”
- Paisley, et. al (2012) “Variational Bayesian Inference with Stochastic Search”
- Yu (2017), “SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient”
- `http://blog.shakirm.com/2015/10/machine-learning-trick-of-the-day-4-reparameterisation-tricks/`
- Kool, Wouter, Herke Van Hoof, and Max Welling. (2019) ”Stochastic Beams and Where To Find Them: The Gumbel-Top-k Trick for Sampling Sequences Without Replacement.” international conference on machine learning: 3499-3508.
- Lee K, Lee H, Lee K, et al. Training Confidence-calibrated Classifiers for Detecting Out-of-Distribution Samples[J]. arXiv: Machine Learning, 2017.

- **consideration 1** $\exp(\mathbf{x}^T\theta_i)$ can become very large

$$\begin{aligned}\pi_i &= \frac{\exp(\mathbf{x}^T\theta_i)}{\sum_{l=1}^{3}\exp(\mathbf{x}^T\theta_l)} \\ &= \frac{\left(\exp(\mathbf{x}^T\theta_i)\right)/C}{\left(\sum_{l=1}^{3}\exp(\mathbf{x}^T\theta_l)\right)/C} = \frac{\exp(\mathbf{x}^T\theta_i - C)}{\sum_{l=1}^{3}\exp(\mathbf{x}^T\theta_l - C)} \\ &= \frac{\exp\left(\mathbf{x}^T\theta_i - \max\left(\{\exp(\mathbf{x}^T\theta_l)\}\right)\right)}{\sum_{l=1}^{3}\exp\left(\mathbf{x}^T\theta_l - \max\left(\{\exp(\mathbf{x}^T\theta_l)\}\right)\right)}\end{aligned}$$

  Softmax is **shift invariant**!
- **consideration 2** $\arg\max$ operation, can be done without $\exp$, i.e.,

$$\underset{i\in\{1,\ldots,k\}}{\arg\max}\ (\pi_1,\ldots\pi_k) \equiv \underset{i\in\{1,\ldots,k\}}{\arg\max}\ (\mathbf{x}^\top\theta_1,\ldots,\mathbf{x}^\top\theta_k)$$

Relationship between Softmax and Sigmoid

$$
\begin{aligned}
\Pr(y_i = 1 | \mathbf{x}_i, \boldsymbol{\theta}) &= \frac{\exp(\mathbf{x}^T \boldsymbol{\theta}_1)}{\exp(\mathbf{x}^T \boldsymbol{\theta}_1) + \exp(\mathbf{x}^T \boldsymbol{\theta}_2)} \\
&= \frac{1}{1 + \exp\left(\mathbf{x}^T (\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1)\right)} \\
&= \frac{1}{1 + \exp\left(\mathbf{x}^T (-\boldsymbol{\theta})\right)} \\
&= \frac{\exp\left(\mathbf{x}^T \boldsymbol{\theta}\right)}{\exp\left(\mathbf{x}^T \boldsymbol{\theta}\right) + 1}
\end{aligned}
$$

What is Logit?

- "tf.nn.softmax_cross_entropy_with_logits", so what is **logit**?

$$\text{logit}(\sigma) = \log\left(\frac{\sigma}{1-\sigma}\right) = \mathbf{x}^\top\theta$$

- logit is the inverse of sigmoid function, let's see why:

$$\begin{aligned}
\implies \frac{\sigma}{1-\sigma} &= \exp(\mathbf{x}^\top\theta) \\
\implies \sigma &= (1-\sigma)\exp(\mathbf{x}^\top\theta) \\
&= \exp(\mathbf{x}^\top\theta) - \sigma\cdot\exp(\mathbf{x}^\top\theta) \\
\implies \sigma\cdot(1+\exp(\mathbf{x}^\top\theta)) &= \exp(\mathbf{x}^\top\theta) \\
\sigma &= \frac{\exp(\mathbf{x}^\top\theta)}{\exp(\mathbf{x}^\top\theta)+1} = \frac{1}{1+\exp(-\mathbf{x}^\top\theta)}
\end{aligned}$$

- therefore:

$$\sigma(.) = \text{logit}^{-1}(\mathbf{x}^\top\theta)$$

Relationship with LogSumExp

- LogSumExp (LSE) function:

$$\text{LSE} = \log\left(\sum_i \exp \mathbf{x}^\top \theta_i\right)$$

$$\begin{aligned}
\max\left[\mathbf{x}^\top \theta_i, \ldots, \mathbf{x}^\top \theta_n\right] &= \log\left(\exp\left(\max\left\{\mathbf{x}^\top \theta_i\right\}\right)\right) \\
&\leq \log\left(\exp(\mathbf{x}^\top \theta_1) + \cdots + \exp(\mathbf{x}^\top \theta_n)\right) \\
&\leq \log\left(n \times \exp(\max\{\mathbf{x}^\top \theta_i\})\right) \\
&= \max\left[\mathbf{x}^\top \theta_1, \ldots, \mathbf{x}^\top \theta_n\right] + \log n
\end{aligned}$$

- therefore:

$$\frac{\partial \log\left(\sum_i \exp \mathbf{x}^\top \bar{\theta}\right)}{\partial \mathbf{x}^\top \theta_i} = \frac{\exp \mathbf{x}^\top \theta_i}{\sum_j \exp \mathbf{x}^\top \theta_j}$$

Limitation of Softmax in classificaiton

- Softmax can only tell the **relative** probablities of membership to each classes
- it is **not** absolute
- no way of telling when a data is **not** belonging to a class
- but result of Softmax may give some “confidence”:

After using some neural networks, then:

- learning the parameters such that:
  - make sample of“in-distribution” classify well, (high confidence/low entropy)
  - at the same time, make sample of ”out-distribution” classify poorly (low confidence/high entropy)
  - samples of ”out-distribution” can’t usually available for training
  - our experience in VET

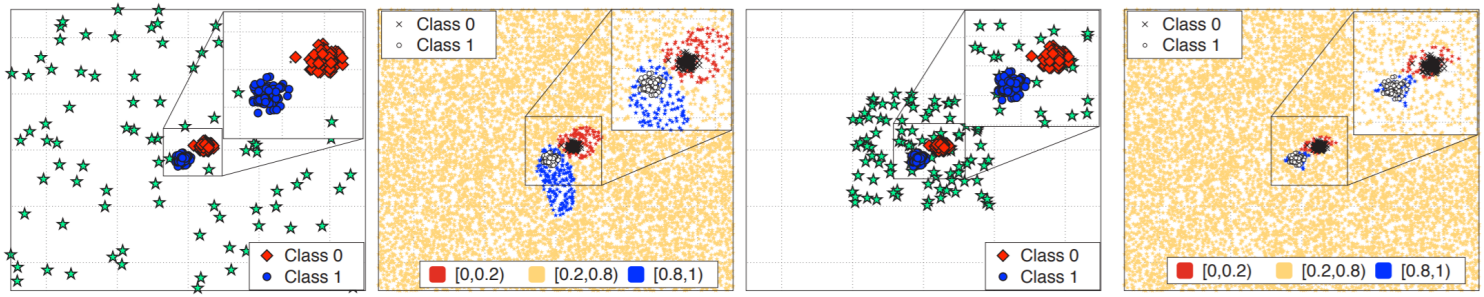
Use GAN to help!

Training Confidence-calibrated classifiers for detecting out-of-distribution samples (ICLR 2018)"

- Imagine someone proposoed a "confidence loss":

$$\min_{\theta} \mathbb{E}_{P_{\text{in}}(\hat{x},\hat{y})}\Big[\underbrace{-\log P_\theta(y=\hat{y}|\hat{x})}_{\text{make in-distribution low entropy}} \Big] + \beta \mathbb{E}_{P_{\text{out}}}\Big[\underbrace{\text{KL}\big(U(y)\|p_\theta(y|x)}_{\text{make out-distribution high entropy}} \big)\Big]$$

- it turns out that too many out-distribution may be bad:



- so the task is: you want only the "border" to be in the out-distribution

Use GAN to help!

- the idea is to generate samples of the out-distribution that is **still close enough** to in-distribution
- the algorithm:

Repeat

1. sample $\{z_1, \ldots z_M\} \sim P(z)$ and $\{\mathbf{x}_1, \ldots, \mathbf{x}_M\} \sim P_{\text{in}}(\mathbf{x})$

$$\max_D \frac{1}{M} \sum_{i=1}^{M} \log D(\mathbf{x}_i) + \log(1 - D(G(z_i)))$$

2. sample $\{z_1, \ldots z_M\} \sim P(z)$

$$\min_G \left(\frac{1}{M} \sum_{i=1}^{M} \log(1 - D(G(z_i))) + \frac{\beta}{M} \sum_{i=1}^{M} \left[\text{KL}\left(U(y) \| p_\theta(y|G(z_i))\right) \right] \right)$$

3. sample $\{z_1, \ldots z_M\} \sim P(z)$ and $\{(\mathbf{x}_1, y_1) \ldots, (\mathbf{x}_M, y_M)\} \sim P_{\text{in}}(\mathbf{x}, y)$

$$\min_\theta \frac{1}{M} \sum_{1=1}^{M} \left[-\log P_\theta(y = y_i|\mathbf{x}_i) + \beta \text{KL}\left(U(y) \| p_\theta(y|G(z_i))\right) \right]$$

Calibrating Neural Networks

Entropy Maximization

$$\max_q \left(-\sum_{i=1}^{n}\sum_{k=1}^{K} q(\mathbf{z}_i)_k \log q(\mathbf{z}_i)_k \right)$$

$$\text{s.t.} \quad q(\mathbf{z}_i)_k \geq 0 \quad \forall i, k$$

$$\sum_{k=1}^{K} q(\mathbf{z}_i)_k = 1 \ \forall i \qquad n \text{ constraints}$$

$$\sum_{i=1}^{n} z_{i,y_i} = \sum_{i=1}^{n}\sum_{k=1}^{K} z_{i,k} q(\mathbf{z}_i)_k \qquad \text{only one constraint}$$

average true class logit equal to average weighted logit

Proof of Entropy Maximization

ignore constraints of $q(\mathbf{z}_i)_k \geq 0 \quad \forall i, k$ for now:

$$\mathcal{L} = -\sum_{i=1}^{n}\sum_{k=1}^{K} q(\mathbf{z}_i)_k \log q(\mathbf{z}_i)_k + \lambda \sum_{i=1}^{n}\sum_{k=1}^{K} z_{i,k} q(\mathbf{z}_i)_k + \sum_{i=1}^{n} \beta_i \sum_{k=1}^{K} \left(q(\mathbf{z}_i)_k - 1\right)$$

$$\frac{\partial \mathcal{L}}{\partial q(\mathbf{z}_i)_k} = -1 - \log q(\mathbf{z}_i)_k + \lambda z_{i,k} + \beta_i$$

$$\frac{\partial \mathcal{L}}{\partial q(\mathbf{z}_i)_k} = 0 \implies \log q(\mathbf{z}_i)_k = +\lambda z_{i,k} + \beta_i - 1$$

$$\implies q(\mathbf{z}_i)_k = \exp\left(\lambda z_{i,k} + \beta_i - 1\right)$$

since $\sum_k q(\mathbf{z}_i)_k = 1$, the following also satisfy:

$$\frac{\exp\left(\lambda z_{i,k} + \beta_i - 1\right)}{\sum_k \exp\left(\lambda z_{i,k} + \underbrace{\beta_i - 1}_{\text{constant}}\right)} = \frac{\exp\left(\lambda z_{i,k}\right)}{\sum_k \exp\left(\lambda z_{i,k}\right)}$$

Checkpoint: NCE for Softmax

Noise Contrastive Estimation

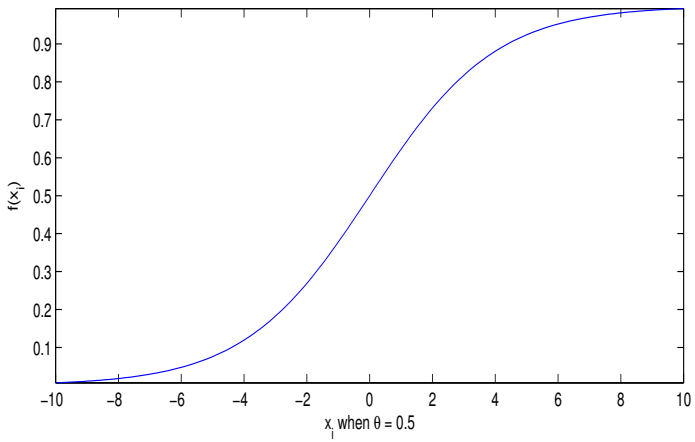
probability and classification

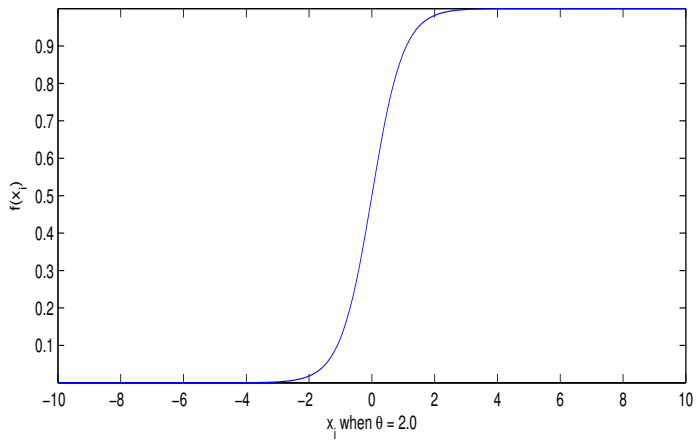
- firstly, probability models and classification are closely related:

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

- 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

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

- Logistic regression

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

Example: Multinomial Distribution & Cross Entropy Loss

- Multinomial Distribution with softmax

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

- 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

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

- **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:

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

- 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 $\{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_k\}$ are target words having high frequencies given $\mathbf{c}$
- $\{\mathbf{r}_1, \mathbf{r}_2, \ldots \mathbf{r}_n\}$ are words having low frequency given $\mathbf{c}$
- say we pick $\mathbf{w}_i \in \{\mathbf{w}_1, \ldots w_k\}$ 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

relationship to GAN

- generator q is **fixed** with no parameter to optimize, in GAN, $g_{\theta_g}(z)$ also needs to be updated
- in NCE, only optimize with respect to parameters of discriminator
- data distribution is learned through discriminator not generator

Change of symbols

- for easier explanation, we change the generic problem into familiar Softmax notation:

$$u_\theta(\mathbf{w}|\mathbf{c}) \equiv \exp(\mathbf{w}^\top \theta)$$

  we dropped $\mathbf{c}$ for notation clarity
- the above of course, applies to any generic un-normalized $u_\theta(\mathbf{w}|\mathbf{c})$

NCE objective function

- probability of $\mathbf{w}$ come from which of the two distributions:

$$\Pr(\mathcal{Y}=1|\theta,\mathbf{w}) = \frac{\exp(\mathbf{w}^\top\theta)}{\exp(\mathbf{w}^\top\theta)+kq(\mathbf{w})} = \sigma\left(\mathbf{w}^\top\theta - \log\left[kq(\mathbf{w})\right]\right)$$
$$\Pr(\mathcal{Y}=0|\theta,\mathbf{w}) = \frac{kq(\mathbf{w})}{\exp(\mathbf{w}^\top\theta)+kq(\mathbf{w})} = 1-\sigma\left(\mathbf{w}^\top\theta - \log\left[kq(\mathbf{w})\right]\right)$$

- check red and blue part are the same

$$\begin{aligned}\sigma\left(\mathbf{w}^\top\theta - \log\left[kq(\mathbf{w})\right]\right) &= \frac{1}{1+\exp\left[-\mathbf{w}^\top\theta+\log\left[kq(\mathbf{w})\right]\right]}\\ &= \frac{1}{1+\exp\left(-\mathbf{w}^\top\theta\right)\times kq(\mathbf{w})}\\ &= \frac{\exp\left[\mathbf{w}^\top\theta\right]}{\exp\left[\mathbf{w}^\top\theta\right]+kq(\mathbf{w})} = \frac{\exp(\mathbf{w}^\top\theta)}{\exp(\mathbf{w}^\top\theta)+kq(\mathbf{w})}\end{aligned}$$

- therefore the objective function is:

$$\theta^* = \arg\max_\theta \sum_{\mathbf{w}\in D}\log\left[\sigma\left(\mathbf{w}^\top\theta - \log\left[kq(\mathbf{w})\right]\right)\right] + \sum_{\mathbf{w}\in\widetilde{D}}\log\left[1-\sigma\left(\mathbf{w}^\top\theta - \log\left[kq(\mathbf{w})\right]\right)\right]$$

NCE and Negative Sampling

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

$$P(\mathcal{Y} = 1|\theta, \mathbf{w}) = \frac{\exp(\mathbf{w}^\top \theta)}{\exp(\mathbf{w}^\top \theta) + |\mathcal{V}| \frac{1}{|\mathcal{V}|}} = \frac{\exp(\mathbf{w}^\top \theta)}{\exp(\mathbf{w}^\top \theta) + 1}$$

$$P(\mathcal{Y} = 0|\theta, \mathbf{w}) = \frac{|\mathcal{V}| \frac{1}{|\mathcal{V}|}}{\exp(\mathbf{w}^\top \theta) + |\mathcal{V}| \frac{1}{|\mathcal{V}|}} = \frac{1}{\exp(\mathbf{w}^\top \theta) + 1}$$

- correspondingly, we have:

$$\begin{aligned}
\mathbf{w}^\top \theta - \log\left[kq(\mathbf{w})\right] &\equiv \mathbf{w}^\top \theta - \log\left(|\mathcal{V}| \frac{1}{|\mathcal{V}|}\right) \\
&= \mathbf{w}^\top \theta \\
&= \mathbf{u}_w^\top \mathbf{v}_c \quad \text{in word2vec context}
\end{aligned}$$

- in Skip-gram of word2vec:

$$\theta^* = \arg\max_\theta \sum_{(w,c) \in D} \log\left[\sigma(\mathbf{u}_w^\top \mathbf{v}_c)\right] + \sum_{(w,c) \in \widetilde{D}} \log\left[\sigma(-\mathbf{u}_w^\top \mathbf{v}_c)\right]$$

why un-normalised $\exp(\mathbf{w}^\top \theta)$ still works?

$$
\begin{aligned}
\Pr(\mathcal{Y}=1|\theta,\mathbf{w}) &= \frac{\exp(\mathbf{w}^\top\theta)}{\exp(\mathbf{w}^\top\theta)+kq(\mathbf{w})} \\
&= \sigma\left(\mathbf{w}^\top\theta - \log(kq(\mathbf{w}))\right) \\
&= \frac{1}{1+\exp\left(-\mathbf{w}^\top\theta + \log(kq(\mathbf{w}))\right)} \\
&= \frac{1}{1+\exp\left[-\mathbf{w}^\top\theta\right]\times kq(\mathbf{w})} = \frac{1}{1+\underbrace{\frac{kq(\mathbf{w})}{\exp(\mathbf{w}^\top\theta)}}_{G(\mathbf{w},\theta)}}
\end{aligned}
$$

- G is the ratio between q and un-normalized p

$$
\begin{aligned}
G(\mathbf{w},\theta) &= \frac{k\ q(\mathbf{w})}{\exp(\mathbf{w}^\top\theta)} \\
&= \underbrace{\frac{m}{n}}_{\nu}\frac{q(\mathbf{w})}{\exp(\mathbf{w}^\top\theta)} = \nu\frac{q(\mathbf{w})}{\exp(\mathbf{w}^\top\theta)}
\end{aligned}
$$

what do we need to prove?

$$G(\mathbf{w}, \theta) = \frac{m}{n} \frac{q(\mathbf{w})}{\exp(\mathbf{w}^\top \theta)} = \nu \frac{q(\mathbf{w})}{\exp(\mathbf{w}^\top \theta)} = \text{ a function of } \theta$$

- the trick is:

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

- after optimization in terms of θ and obtain:

$$\theta^* = \arg\max_{\theta} \left[\mathcal{C}_n(\theta) \right]$$

- substitute θ^* into $G(\mathbf{w}, \theta)$ and try to maximize above using $G(\mathbf{w}, \theta^*)$ under large sample size n and m

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

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

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

using functional derivative

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

- take functional derivative:

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

$$\left(G^*(\mathbf{w}, \theta) \equiv \nu \frac{q(\mathbf{w})}{\exp(\mathbf{w}^\top \theta^*)}\right) \to \nu \frac{q(\mathbf{w})}{p_\theta(\mathbf{w})} \implies \exp(\mathbf{w}^\top \theta^*) \to p_\theta(\mathbf{w}) \quad \text{as } \theta \to \theta^*$$

i.e., normalization constant is 1

- take a break to discuss functional derivative, specifically **Euler-Lagrange Equation**

notes on Euler-Lagrange Equation

for normal function

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:

$$\left.\frac{\mathrm{d}J_\varepsilon}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0} = \left(\left.\frac{\mathrm{d}\, J(g_\varepsilon(\mathbf{x}))}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0}\right) = \left(\frac{\mathrm{d}\, J(g_\varepsilon(\mathbf{x}))}{\mathrm{d}\, g_\varepsilon(\mathbf{x})} \underbrace{\left.\frac{\mathrm{d}\, g_\varepsilon(\mathbf{x})}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0}}_{=1}\right) = \left.\frac{\mathrm{d}\, J(g_\varepsilon(\mathbf{x}))}{\mathrm{d}\, g_\varepsilon(\mathbf{x})}\right|_{\varepsilon=0}$$

$$= \left.\frac{\mathrm{d}\, J(\mathbf{x}+\varepsilon)}{\mathrm{d}\, (\mathbf{x}+\varepsilon)}\right|_{\varepsilon=0} = 0$$

$$\implies J'(\mathbf{x}) = 0$$

- 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 $\left.\frac{\mathrm{d}J_\varepsilon}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0}$ (1)

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

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

- 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

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

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

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

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

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

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

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

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

Euler-Lagrange Equation: Standard example

- Find real-valued function $\mathbf{f}$ on interval $[a, b]$, such that:

$$\mathbf{f}(a) = c \qquad \text{and} \qquad \mathbf{f}(b) = d,$$

for which the path length J along the curve traced by $\mathbf{f}$ is as short as possible.

$$\begin{aligned}
(\mathrm{d}s)^2 &= (\mathrm{d}x)^2 + (\mathrm{d}\mathbf{f})^2 \\
&= \left(1 + \frac{(\mathrm{d}\mathbf{f})^2}{(\mathrm{d}x)^2}\right)(\mathrm{d}x)^2 \\
&= \left(1 + \mathbf{f}'^2\right)(\mathrm{d}x)^2 \\
\implies \mathrm{d}s &= \sqrt{1 + \mathbf{f}'^2}\,\mathrm{d}x \\
\implies s &= \int_a^b \underbrace{\sqrt{1 + \mathbf{f}'^2}}_{F(x,\mathbf{f},\mathbf{f}')}\,\mathrm{d}x
\end{aligned}$$

Euler-Lagrange Equation: solution

- the integrand function is:

$$F(x, \mathbf{f}(x), \mathbf{f}'(x)) = \sqrt{1 + \mathbf{f}'^2}$$

- The partial derivatives of F are:

$$\frac{\partial F(x, \mathbf{f}, \mathbf{f}')}{\partial \mathbf{f}'} = \frac{\mathbf{f}'}{\sqrt{1 + \mathbf{f}'^2}} \quad \text{and} \quad \frac{\partial F(x, \mathbf{f}, \mathbf{f}')}{\partial \mathbf{f}} = 0$$

- **Euler-Lagrange equation**:

$$\frac{\partial F}{\partial \mathbf{f}} - \frac{\mathrm{d}}{\mathrm{d}x} \frac{\partial F}{\partial \mathbf{f}'} = 0$$

$$\implies \frac{\partial}{\mathrm{d}x} \frac{\mathbf{f}'(x)}{\sqrt{1 + (\mathbf{f}'(x))^2}} = 0$$

anything has a derivative equal 0 must mean it is a constant

$$\frac{\mathbf{f}'(x)}{\sqrt{1 + (\mathbf{f}'(x))^2}} = C = \text{constant}$$

$$\implies \mathbf{f}'(x) = A \text{ another constant}$$

$$\implies \mathbf{f}(x) = Ax + B$$

CHECKPOINT: Softmax Re-Parameterization

Probability Re-Parameterization

Why Re-parameterization: **(1) otherwise infeasible**

- imagine a **Computation Graph**:

$$
\begin{aligned}
y_2 &= {f^1}_{\theta_1}(y_1) \\
y_3 &= {f^2}_{\theta_2}(y_2) = {f^2}_{\theta_2}({f^1}_{\theta_1}(y_1)) \\
&\vdots \\
z &\sim \Pr_{\theta_{n-1}}(y_{n-1}) \\
y_n &= {f^n}_{\theta_n}(z) \\
&\vdots
\end{aligned}
$$

- Monte-Carlo step $\frac{\partial \mathcal{C}}{\partial z(\theta)} = \cdots \times \frac{\partial y_n}{\partial z} \times \ldots$ doesn't have derivative, and but we do need it in the chain rule
- one trick is to use Reinforcement Learning, e.g., Seq-GAN

Tricks to avoid it

- In some applications tricks can be used: for example in GAN.
- Traditional GAN's Generator:

$$\min_G \max_D \left(\mathcal{L}(D, G) \equiv \mathbb{E}_{\mathbf{x} \sim p_r(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{z \sim p_z(z)} \left[\log(1 - D(G(z))) \right] \right)$$

  Generator given D is:

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

  and partial derivative contains $\cdots \times \frac{\partial D}{\partial G} \times \ldots$
- can be thought as fixed D helps G to generate better sample so that D score it higher.
- when Generator generates a sequence of "discrete" words, it has Monte-Carlo samples, unable to take derivatives.
- Reinforcement Learning can be used as such trick!

Seq-GAN algorithm

repeat
  for G-steps **do**
    generate a sequence $Y_{1:T} = (y_1, \ldots, y_T) \sim G_\theta$
    for t in 1 : T **do**

$$Q(s = Y_{1:t-1}, a = y_t) = \begin{cases} \frac{1}{N} \sum_{n=1}^{N} D_\phi(Y^n_{1:T}) & Y^n_{1:T} \in \mathrm{MC}^G_\beta(Y_{1:t}, N) \quad t < T \\ D_\phi(Y^n_{1:T}) & t = T \end{cases}$$

    end for

$$\nabla_\theta J(\theta) = \sum_{t=1}^{T} \mathbb{E}_{Y_{1:t-1} \sim G_\theta} \left[\sum_{y_t \sim \mathcal{Y}} \nabla_\theta G_\theta(y_t | Y_{t-1}) Q(Y_{1:t-1}, y_t) \right]$$

  end for

  for D-steps **do**
    collect negative samples from current G_θ, combine with given positive samples
    Train discriminator D_ϕ
  end for

  $\beta \leftarrow \theta$
end repeat

Comments

- In Traditional GAN, Monte-Carlo step $z \sim p(z)$ occurs before deterministic transform G, so it doesn't affect derivative of G_θ
- In Natural Language Generation, Monte-Carlo step occur during every step of G, i.e., the G_θ participate in the generation of tokens, so the derivatives can't pass
- In Seq-GAN, G_θ is not learned through derivatives of $\left(\mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]\right)$, but instead D and G are acting through an *intermediary* $Q(Y_{1:t-1}, y_t)$
- Colloquially, G_θ is learned through Policy Gradient, where $Q(s, a)$ is **indirectly** guided by D in a separate step.
- same applies to continuous sequence: where G_θ outputs Gaussian parameters at time t, then sample from it is fed into input at time $t + 1$

Why Re-parameterization: **(2) lower variance in Monte-Carlo Integral**

- begin with 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 $\Pi \equiv \{s_1, a_1, \ldots, s_T, a_T\}$

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

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

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

Re-parameterization trick

Re-parameterization trick is then:
instead of sampling $z \sim \Pr_\theta(y)$ directly, we sample:

$$\epsilon \sim p(\epsilon), \qquad z = g(\epsilon, \theta | y)$$

more concretely:

- only need to know deterministic function $z = g(\epsilon, \theta)$ and distribution $p(\epsilon)$
- does **not** always need to explicitly know distribution of z

example, Gaussian variable: $z \sim \mathcal{N}(z; \mu(\theta), \sigma(\theta))$ can be re-parameterised into as a function of a standard Gaussian variable:

$$\epsilon \sim \underbrace{\mathcal{N}(0, 1)}_{p(\epsilon)}$$
$$z = g(\epsilon, \theta) = \underbrace{\mu(\theta) + \epsilon\sigma(\theta)}_{g(\epsilon, \theta)}$$

revision on change of variable

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

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

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

- without change of limits

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

- with change of limits

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

After re-parameterization trick:

- when computing expectation, $p(\epsilon)$ is **no longer** parameterized by θ:

$$\mathbb{E}_{\epsilon \sim p(\epsilon)}[f(\underbrace{g(\epsilon, \theta)}_{z})] = \int_{\epsilon} f(g(\epsilon, \theta))p(\epsilon)\mathrm{d}\epsilon$$

- taking derivative, (note you can change ∇_θ from outside of $\mathbb{E}(.)$ to inside):

$$\begin{aligned} \implies \nabla_\theta \mathbb{E}_{\epsilon \sim p(\epsilon)}\left[f(g(\epsilon, \theta))\right] &= \mathbb{E}_{\epsilon \sim p(\epsilon)}\left[\nabla_\theta f(g(\epsilon, \theta))\right] \\ &= \int_{\epsilon} \nabla_\theta f(g(\epsilon, \theta))p(\epsilon)\mathrm{d}\epsilon \end{aligned}$$

- note **without** re-parameterization, can **not** change ∇_θ from outside of $\mathbb{E}(.)$ to inside

$$\underline{\nabla_\theta \mathbb{E}_{p(z|\theta)}\left[f(z)\right]} = \int_{z} f(z)\nabla_\theta p(z|\theta) = \mathbb{E}_{p(z|\theta)}\left[f(z)\nabla_\theta \log(p(z|\theta))\right] \underbrace{\neq \mathbb{E}_{p(z|\theta)}\left[\nabla_\theta f(z)\right]}$$

- during gradient decent, ϵ are sampled independent of θ

Simple example

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

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

- 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 $\sigma(\theta) = 1$:

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

- comment: $\nabla_\theta \log\left(\frac{1}{\sigma\sqrt{2\pi}}\exp^{-\frac{(z-\mu)^2}{2\sigma^2}}\right)$ can be a bit fiddly

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

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

$$\epsilon \sim \mathcal{N}(0, 1)$$
$$z \equiv g(\epsilon, \theta) = \mu(\theta) + \epsilon\sigma(\theta)$$

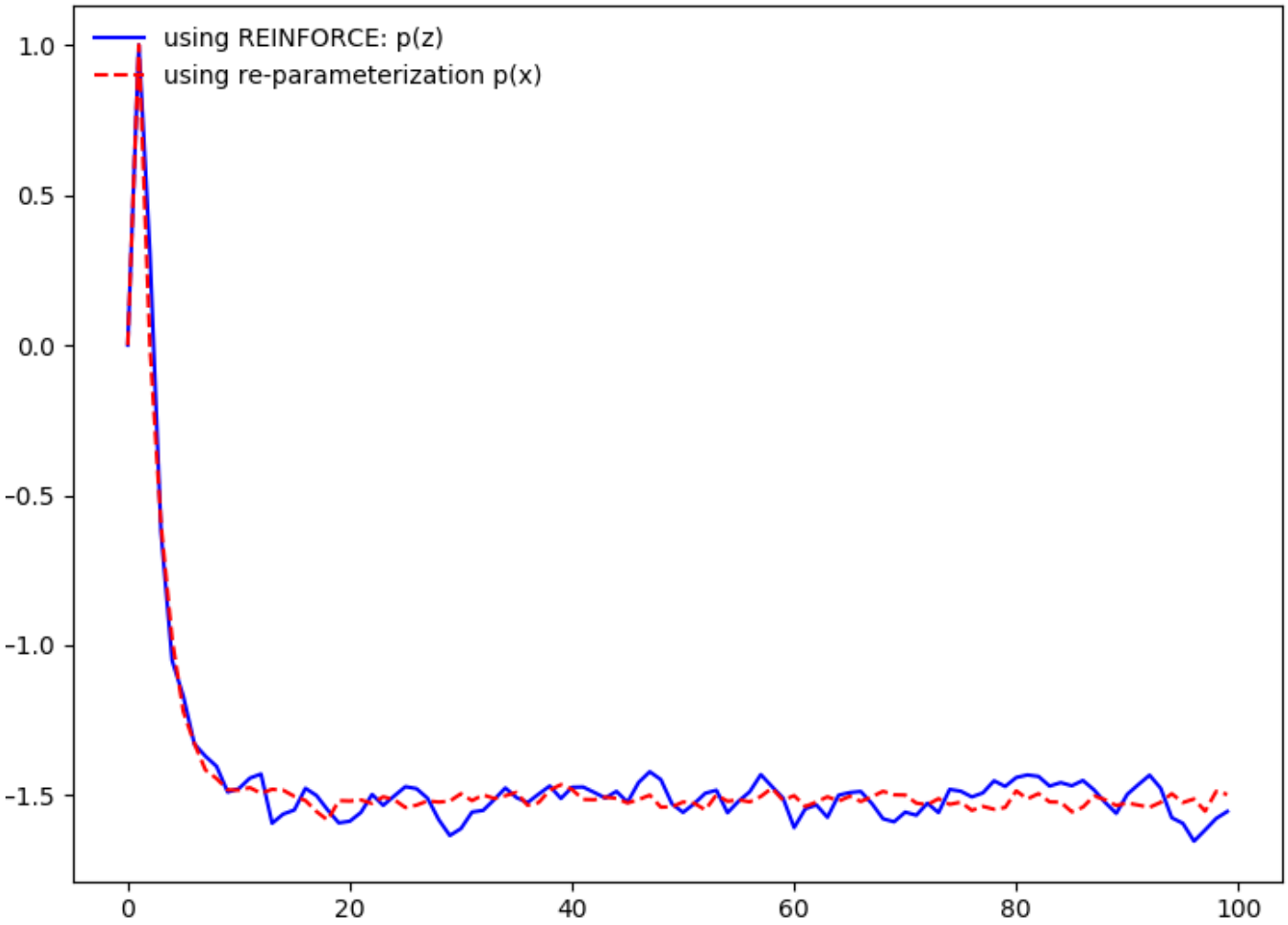
- the re-parameterised version is:

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

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

results

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



Application: Replace $z \sim q_\phi(z)$ with $\epsilon \sim q(\epsilon)$ in Variation Inference

- ELBO:

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

- obviously $q_\phi(z)$ are dependent on ϕ, so we need the re-parameterization

$$z \sim q_\phi(z) \equiv \epsilon \sim p(\epsilon) \text{ and } z = g(\phi, \epsilon)$$

- after re-parameterization, it appears to be:

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

Log-likelihood and Evidence Lower Bound (ELBO)

- 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 $q(z)$:

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

Auto-Encoder (VAE)

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 for **minimization**:

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

- to have high value in $\mathbb{E}_{z \sim Q_\theta(z|x_i)}\left[\log P_\phi(x_i|z)\right]$, it needs:

$$\begin{aligned} Q_\theta(z|x_i) \uparrow &\implies P_\phi(x_i|z) \uparrow \text{ and} \\ Q_\theta(z|x_i) \downarrow &\implies P_\phi(x_i|z) \downarrow \end{aligned}$$

- can think the setting as a joint density $\mathcal{P}(x_i, z)$ where conditionals are $Q_\theta(z|x_i)$ and $P_\phi(x_i|z)$
  - **reconstruction** makes $\mathcal{P}(x_i, z)$ as highly correlated as possible (high accuracy, but less diversity - unable to generate "new" sample)
  - **regularizer** makes $\mathcal{P}(x_i, z)$ least correlated as possible. when KL is minimized, conditional $Q_\theta(z|x_i)$ is independent of x_i, i.e., $p(z)$) (low accuracy, but has high diversity)

look at the ELBO again

- we are not using normal ELBO, i.e., $q(z)$ to maximize:

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

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

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

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

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

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

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

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

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

Variation auto-encoder

- knowing

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

$$\implies -\ln\left(p(\mathbf{y})\right) + \underbrace{\mathrm{KL}\left(q(z|\mathbf{y})\|p(z|\mathbf{y})\right)}_{\geq 0} = \underbrace{-\mathbb{E}_{z\sim q(z|\mathbf{y})}\left[\ln(p(\mathbf{y}|z))\right] + \mathrm{KL}\left(q(z|\mathbf{y})\|p(z)\right)}_{\mathcal{L}(\cdot)}$$

choose $q(z|y)$ to minimize $\mathcal{L}(.)$:

$$\implies \mathrm{KL}\left(q(z|\mathbf{y})\|p(z|\mathbf{y})\right) = 0 \qquad \text{by letting } q(z|\mathbf{y}) = p(z|\mathbf{y})$$

- so the lower bound of $\mathcal{L}(.)$ is $\ln\left(p(\mathbf{y})\right)$.

objective function illustration

new intepretation:

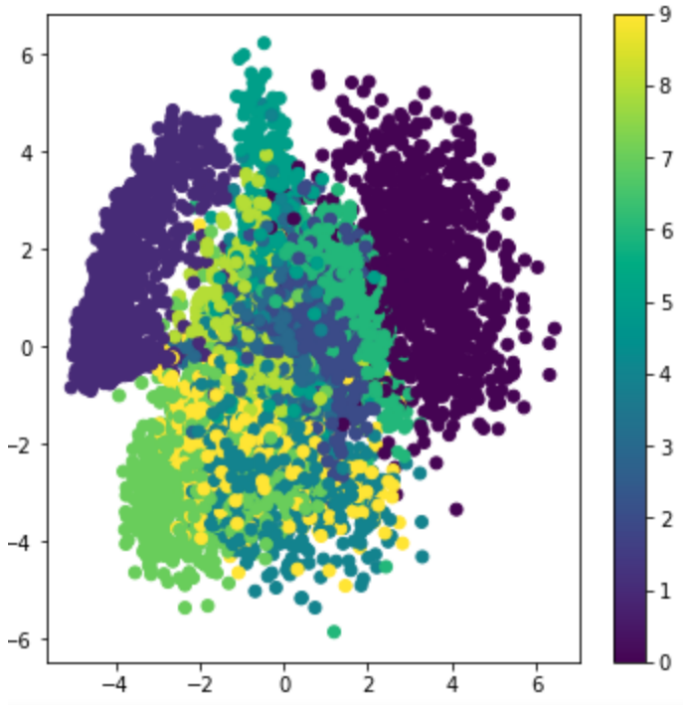
- loss at loss function again:

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

- 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 $\mathrm{KL}(\mathcal{N}(\mu_1, \Sigma_1) \| \mathcal{N}(\mu_2, \Sigma_2))$

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

- substitute $\bar{\mu}_1 = [\mu_1, \ldots, \mu_K]^\top$ and $\Sigma_1 = \mathrm{diag}(\sigma_1, \ldots, \sigma_K)$, $\mu_2 = \mathbf{0}$ and $\Sigma_2 = \mathbf{I}$:

$$
\begin{aligned}
\mathrm{KL} &= \frac{1}{2} \left(\mathrm{tr}(\Sigma_1) + \bar{\mu}_1^T \bar{\mu}_1 - K - \log \det(\Sigma_1) \right) \\
&= \frac{1}{2} \left(\sum_k \sigma_k^2 + \sum_k \mu_k^2 - \sum_k 1 - \log \prod_k \sigma_k^2 \right) \\
&= \frac{1}{2} \sum_k \left(\sigma_k^2 + \mu_k^2 - 1 - \log \sigma_k^2 \right)
\end{aligned}
$$

there is an even simpler way to compute KL, when $p(x_1, x_2) = p(x_1)p(x_2)$ and $q(x_1, x_2) = q(x_1)q(x_2)$

$$\mathrm{KL}(p,\, q) = -\left(\int p(x_1)\log q(x_1)\mathrm{d}x_1 - \int p(x_1)\log p(x_1)\mathrm{d}x_1\right)$$

$$\implies \mathrm{KL}(p(x_1)p(x_2)\|q(x_1)q(x_2))$$

$$= -\left(\int_{x_1}\int_{x_2} p(x_1)p(x_2)\left[\log q(x_1) + \log q(x_2)\right]\mathrm{d}x_1 - p(x_1)p(x_2)\left[\log p(x_1) + \log p(x_2)\right]\mathrm{d}x_1\right)$$

$$= -\left(\int_{x_1}\int_{x_2} \left[p(x_1)p(x_2)\log q(x_1) + p(x_1)p(x_2)\log q(x_2) - p(x_1)p(x_2)\log p(x_1) - p(x_1)p(x_2)\log p(x_2)\right]\mathrm{d}x_1\right)$$

$$= -\left(\int_{x_1}\int_{x_2} p(x_1)p(x_2)\log q(x_1) + \int_{x_1}\int_{x_2} p(x_1)p(x_2)\log q(x_2) - \int_{x_1}\int_{x_2} p(x_1)p(x_2)\log p(x_1) - \int_{x_1}\int_{x_2} p(x_1)p(x_2)\log p(x_2)\mathrm{d}x_1\right)$$

$$= -\left(\int_{x_1} p(x_1)\log q(x_1)\int_{x_2} p(x_2) + \int_{x_1} p(x_1)\int_{x_2} p(x_2)\log q(x_2) - \int_{x_1} p(x_1)\log p(x_1)\int_{x_2} p(x_2) - \int_{x_1} p(x_1)\int_{x_2} p(x_2)\log p(x_2)\right)$$

$$= -\left(\int_{x_1} p(x_1)\log q(x_1) + \int_{x_2} p(x_2)\log q(x_2) - \int_{x_1} p(x_1)\log p(x_1) - \int_{x_2} p(x_2)\log p(x_2)\right)$$

$$= -\left(\int_{x_1} p(x_1)\log q(x_1) - \int_{x_1} p(x_1)\log p(x_1)\right) - \left(\int_{x_2} p(x_2)\log q(x_2) - \int_{x_2} p(x_2)\log p(x_2)\right)$$

$$= \mathrm{KL}(p(x_1)\|q(x_1)) + \mathrm{KL}(p(x_2)\|q(x_2))$$

therefore,

$$\mathrm{KL}(p(x_1)p(x_2)\|q(x_1)q(x_2)) = \mathrm{KL}(p(x_1)\|q(x_1)) + \mathrm{KL}(p(x_2)\|q(x_2))$$

$$\implies \mathrm{KL}\left(\prod_k p(x_k)\Big\|\prod_k q(x_k)\right) = \sum_{i=1}^{k}\mathrm{KL}(p(x_i)\|q(x_i))$$

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 $p(x) = \mathcal{N}(\mu, \sigma)$ and $q(x) = \mathcal{N}(0, 1)$:

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

- moving into k dimensions, and apply $\mathrm{KL}\left(\prod_k p(x_k) \| \prod_k q(x_k)\right) = \sum_{i=1}^k \mathrm{KL}(p(x_i) \| q(x_i))$:

$$\mathrm{KL}\left(\prod_k p(x_k) \| \prod_k q(x_k)\right) = \frac{1}{2} \sum_k \left[\frac{\sigma^2}{2} + \frac{\mu^2}{2} - \frac{1}{2} - \log \sigma^2\right]$$

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 $Q_\theta(z|x_i) \equiv \mathcal{N}\left(\mu(x_i,\theta), \Sigma(x_i,\theta)\right)$
- 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 $\mu(x_i,\theta), \Sigma(x_i,\theta)$

Other re-parameterizations available?

- many available!

name	$p(z;\theta)$	$p(\epsilon)$	$g(\epsilon,\theta)$
Exponential	$\exp(-x); x > 0$	$\epsilon \sim [0;1]$	$\ln(1/\epsilon)$
Cauchy	$\frac{1}{\pi(1+x^2)}$	$\epsilon \sim [0;1]$	$\tan(\pi\epsilon)$
Laplace	$\mathcal{L}(0;1) = \exp(-\lvert x\rvert)$	$\epsilon \sim [0;1]$	$\ln(\frac{\epsilon_1}{\epsilon_2})$
Laplace	$\mathcal{L}(\mu;b)$	$\epsilon \sim [0;1]$	$\mu - b\text{sgn}(\epsilon)\ln(1-2\lvert\epsilon\rvert)$
Gaussian	$\mathcal{N}(0;1)$	$\epsilon \sim [0;1]$	$\sqrt{\ln(\frac{1}{\epsilon_1})}\cos(2\pi\epsilon_2)$
Gaussian	$\mathcal{N}(\mu;RR^\top)$	$\epsilon \sim \mathcal{N}(0;1)$	$\mu + R\epsilon$
Rademacher	$Rad(\frac{1}{2})$	$\epsilon \sim \text{Bern}(\frac{1}{2})$	$2\epsilon - 1$
Log-Normal	$\ln\mathcal{N}(\mu;\sigma)$	$\epsilon \sim \mathcal{N}(\mu;\sigma^2)$	$\exp(\epsilon)$
Inv Gamma	$\mathcal{IG}(lk;\theta)$	$\epsilon \sim \mathcal{G}(k;\theta^{-1})$	$\frac{1}{\epsilon}$

- however, today we are interested only in **Softmax distribution** parameterizations!

Apply re-parameterization to Softmax

- when we have the following

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

- can we find their corresponding:

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

Re-parameterization using Gumbel-max trick

- Gumbel-max trick also means:

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

$$k = \underbrace{\underset{i \in \{1,\ldots,K\}}{\arg\max} \{\mu_1(\theta) + \mathcal{G}_1, \ldots, \mu_K(\theta) + \mathcal{G}_K\}}_{g(\mathcal{G},\theta)} \qquad \mathbf{v} = \text{one-hot}(k) \qquad f(\mathbf{v})$$

- this is a form of re-paramterization:

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

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

Gumbel distribution definitions

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

$$p(\mathcal{G}|\mu,1) \equiv \text{gumbel}(Z=\mathcal{G}\,;\,\mu) = \exp\Big[-(\mathcal{G}-\mu)-\exp(-(\mathcal{G}-\mu))\Big]$$

- CDF of Gumbel:

$$\Pr(\mathcal{G}|\mu,1) \equiv \text{Gumbel}(Z\leq\mathcal{G}\,;\,\mu) = \exp\Big[-\exp(-(\mathcal{G}-\mu))\Big]$$

- it is obvious that:

$$p(\mathcal{G}|\mu,1) = \exp(-\mathcal{G}+\mu)\Pr(\mathcal{G}|\mu)$$

which is a property you must know to work with Gumbels!

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

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

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 summary

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

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

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

$$\text{where } \underbrace{G_i \sim \text{gumbel}(\mathcal{G}\,;\, \mu_i) \equiv \exp\left[-(\mathcal{G} - \mu_i) - \exp\{-(\mathcal{G} - \mu_i)\} \right]}$$

$$\implies K = \underset{i \in \{1,\ldots,L\}}{\arg\max} \{\mu_1 + \mathcal{G}_1, \ldots, \mu_L + \mathcal{G}_L\}$$

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

- 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 :

$$\equiv \underset{i \in \{1,\ldots,L\}}{\arg\max} \{\log(\mu_1) + \mathcal{G}_1, \ldots, \log(\mu_L) + \mathcal{G}_L\}$$

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

how to sample a Gumbel?

- CDF of a Gumbel:

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

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

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

- therefore, sampling strategy:

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

Second problem with Softmax re-parameterisation

- the other remaining **problem**: sample **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**

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

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

- 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:

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

Problem with Gumble Softmax

- the problem with Gumbel Softmax is it is **biased**
- Bias-ness is obvious: Only sampling Gumbel Hard-max is unbiased

CHECKPOINT: Take Softmax to the sequential manner

Stochastic Beam Search

Max of “value” and “index” are independent

- these two operations are independent:

$$\max_{i \in B} \{G_{\phi_i}\} \sim \text{Gumbel}\left(\underbrace{\log \sum_{j \in B} \exp \phi_j}_{\text{logSumExp}} \right)$$

$$\arg\max_{i \in B} \{G_{\phi_i}\} \sim \text{Categorical}\left(\frac{\exp(\phi_i)}{\sum_{j \in B} \exp \phi_j}, i \in B \right)$$

Gumbel Top-k Trick (1)

- use $*$ means it's ordered
- $N^*_{-k} = N \setminus \{i^*_1, \ldots, i^*_{k-1}\}$, i.e., the remaining set also include k^{th} element

$$P\left(I^*_k = i^*_k \middle| I^*_i = i^*_1, \ldots, I^*_{k-1} = i^*_{k-1}, \{\phi_i\}, \{G_{\phi_i}\}\right)$$

$$= P\left(i^*_k = \underset{i \in N^*_{-k}}{\arg\max}\{G_{\phi_i}\} \middle| I^*_i = i^*_1, \ldots, I^*_{k-1} = i^*_{k-1}\right) \quad \text{omit } \{\phi_i\}, \{G_{\phi_i}\} \text{ for clarity}$$

$$= P\Bigg(\underbrace{i^*_k = \underset{i \in N^*_{-k}}{\arg\max}\{G_{\phi_i}\}}_{i^*_k \text{ is the largest of the remaining set}} \Bigg| \underbrace{G_{\phi^*_{i_{k-1}}} > \underset{i \in N^*_{-k}}{\max}\{G_{\phi_i}\}}_{\text{the previous one is larger than remaining set}} \Bigg)$$

$$= P\left(i^*_k = \underset{i \in N^*_{-k}}{\arg\max}\{G_{\phi_i}\}\right) \quad \max \text{ and } \arg\max \text{ are independent}$$

$$= \frac{\exp \phi_{i^*_k}}{\sum_{l \in N^*_{-k}} \exp \phi_l}$$

- **meaning** sample $i^*_k \sim \frac{\exp \phi_{i^*_k}}{\sum_{l \in N^*_{-k}} \exp \phi_l}$ can just simply choose $i^*_k = \arg\max_{i \in N^*_{-k}}$

Gumbel Top-k Trick (2)

$$
\begin{aligned}
&P\left(I_1^* = i_1^*, \ldots, I_k^* = i_k^*\right) \\
&= P\left(I_k^* = i_k^* \middle| I_i^* = i_1^*, \ldots, I_{k-1}^* = i_{k-1}^*\right) \times \\
&\quad P\left(I_{k-1}^* = i_{k-1}^* \middle| I_i^* = i_1^*, \ldots, I_{k-2}^* = i_{k-2}^*\right) \times \\
&\quad \cdots \times \\
&\quad P\left(I_1^* = i_1^*\right) \\
&= \prod_{j=1}^{k} \frac{\exp \phi_{i_j^*}}{\sum_{l \in N_{-j}^*} \exp \phi_l}
\end{aligned}
$$

- **moral of the story**: if we choose k largest elements from $\{G_{\phi_i}\}$ then $I_1^*, \ldots, I_k^*$ is an ordered "sample **without** replacement" (i.e., choose each from the remaining set recursively) from:

$$
\text{categorical}\left(\frac{\exp(\phi_i)}{\sum_{j \in N} \exp(\phi_j)}\right)
$$

Sampling set of Gumbels with truncation T (1)

- **CDF** of truncated Gumbel

$$
\begin{aligned}
F_{\phi,T}(g) &= P(G \leq g | G \leq T) \\
&= \frac{P(G \leq g \ \cap \ G \leq T)}{P(G \leq T)} \\
&= \frac{F_{\phi}(\min(g, T))}{F_{\phi}(T)} \\
&= \frac{\exp(-\exp(\phi - \min(g, T)))}{\exp(-\exp(\phi - T))} \\
&= \exp\left(\exp(\phi - T) - \exp(\phi - \min(g, T))\right)
\end{aligned}
$$

- without truncation threshold T, CDF is $F_{\phi}(g)$;
- With truncation threshold T, $F_{\phi}(g)$ is being normalized using $F_{\phi}(T)$ instead of 1 (visualize it)
- reason have $\min(g, T)$ is that we cannot prevent people putting $g > T$.

Sampling set of Gumbels with maximum T (2)

- **inverse CDF** of truncated Gumbel

$$
\begin{aligned}
u &= \exp\left(\exp(\phi - T) - \exp(\phi - \min(g, T))\right) \\
\implies \log(u) &= \exp(\phi - T) - \exp(\phi - \min(g, T)) \\
\implies \exp(\phi - \min(g, T)) &= \exp(\phi - T) - \log(u) \\
\implies \phi - \min(g, T) &= \log\left(\exp(\phi - T) - \log(u)\right) \\
\min(g, T) &= \phi - \log\left(\exp(\phi - T) - \log(u)\right) \\
\implies G = F^{-1}_{\phi,T} &= \phi - \log\left(\exp(\phi - T) - \log(u)\right)
\end{aligned}
$$

- unlike computing CDF where g may be larger than T, when computing CDF^{-1}, any $u \in (0, \ldots 1)$ will do

$$
\implies G = F^{-1}_{\phi,T} = \phi - \log\left(\exp(\phi - T) - \log(u)\right)
$$

re-scale $G_{\phi_i} \rightarrow \tilde{G}_{\phi_i}$ (1)

- $Z = \max\{G_{\phi_i}\}$
- G_{ϕ_i} is sampled using truncation of Z. Then "re-scaled" to $\tilde{G}_{\phi_i}$ which has truncation T:

$$
\begin{aligned}
\tilde{G}_{\phi_i} &= F^{-1}_{\phi,T}\left(F_{\phi_i,Z}(G_{\phi_i})\right) \\
&= \phi_i - \log\left(\exp(\phi_i - T)\underbrace{-\exp(\phi_i - Z) + \exp(\phi_i - G_{\phi_i})}_{-u \equiv -F_{\phi_i,Z}(G_{\phi_i})}\right) \\
&= \phi_i - \log\left(\exp(\phi_i)\left(\exp(-T) - \exp(-Z) + \exp(-G_{\phi_i})\right)\right) \\
&= \phi_i - \phi_i - \log\left(\left(\exp(-T) - \exp(-Z) + \exp(-G_{\phi_i})\right)\right) \\
&= -\log\left(\exp(-T) - \exp(-Z) + \exp(-G_{\phi_i})\right)
\end{aligned}
$$

re-scale $G_{\phi_i} \to \tilde{G}_{\phi_i}$ (2)

- look at $G_{\phi_i} \to \tilde{G}_{\phi_i}$:

$$\tilde{G}_{\phi_i} = f(G_{\phi_i}) = \underbrace{-\log}_{\text{monotone}} \left(\underbrace{\exp(-T) - \exp(-Z)}_{\text{constant}} + \underbrace{\exp(-G_{\phi_i})}_{\text{monotone}} \right)$$

- since $\tilde{G}_{\phi_i} = f(G_{\phi_i})$ is monotonically increasing, it preserves $\arg\max$:

$$\arg\max_i \tilde{G}_{\phi_i} = \arg\max_i G_{\phi_i} \sim \text{Categorical}\left(\frac{\exp \phi_i}{\sum_j \exp \phi_j} \right)$$

Sampling Gumbels with maximum T: **Step 1**

- **Conventional thinking**:

$$i^* \sim \text{Categorical}\left(\frac{\exp \phi_i}{\sum_j \exp \phi_j}\right) \qquad \text{via Gumbel-max trick of } G_{\phi_i}$$

  no need to condition on T as $\arg\max_i$ independent of the $\max$

- **New method**: (re-scale $G_{\phi_i} \to \tilde{G}_{\phi_i}$)

$$i^* \sim \text{Categorical}\left(\frac{\exp \phi_i}{\sum_j \exp \phi_j}\right) \qquad \text{via Gumbel-max trick of } \tilde{G}_{\phi_i}$$

Sampling Gumbels with maximum T: **Step 2**

For $i = i^* = \arg\max_i G_{\phi_i}$

- **Conventional thinking**:
  Set $\tilde{G}_{\phi_i} = T$, since this follows from conditioning on $\max T$ and $\arg\max i$
- **New method**: (re-scale $G_{\phi_i} \to \tilde{G}_{\phi_i}$)

  can set $T = \tilde{G}_{\phi_i}$ because, substitute $Z = \max\{G_{\phi_i}\}$:

$$\begin{aligned}\tilde{G}_{\phi_i} &= F^{-1}_{\phi,T}(F_{\phi_i,Z}(G_{\phi_i})) \\ &= F^{-1}_{\phi,T}(F_{\phi_i,Z}(Z)) \\ &= T\end{aligned}$$

Sampling Gumbels with maximum T: **Step 3 (a)**

For $i \neq i^*$:

- **Conventional thinking**:
  Sample $\tilde{G}_{\phi_i} \sim \text{TruncatedGumbel}(\phi_i, T)$, condition on $\max T$ and $\arg\max i^*$:

$$P\big(\tilde{G}_{\phi_i} < g | \max_i\{\tilde{G}_{\phi_i}\} = T, \arg\max_i\{\tilde{G}_{\phi_i}\} = i^*, i \neq i^*\big)$$
$$= P\big(\tilde{G}_{\phi_i} < g | \tilde{G}_{\phi_i} < T\big)$$

- **New method**: (re-scale $G_{\phi_i} \to \tilde{G}_{\phi_i}$)
  under re-scaling, we can simply pick the rest of "top $k-1$" $\{\tilde{G}_{\phi_i}\}$ and their indices

  we need to prove CDF:

$$P(\tilde{G}_{\phi_i} \leq g | i \neq \arg\max_i G_{\phi_i}) = F_{\phi_i, T}(g)$$

  i.e., $\tilde{G}_{\phi_i} \sim \text{TruncatedGumbel}(\phi, T)$ with its CDF equal to $F_{\phi_i, T}(g)$

Sampling Gumbels with maximum T: **Step 3 (b)**

- look at the CDF of $\tilde{G}_{\phi_i}$ when it's the largest:

$$
\begin{aligned}
& P(\tilde{G}_{\phi_i} \leq g | i \neq \arg\max_i \{G_{\phi_i}\}) \\
&= \int_z P\big(\tilde{G}_{\phi_i} \leq g \big| Z, i \neq \arg\max_i \{G_{\phi_i}\}\big) P(Z) \mathrm{d}z \\
&= \mathbb{E}_Z\big[P\big(\tilde{G}_{\phi_i} \leq g \big| Z, i \neq \arg\max_i \{G_{\phi_i}\}\big)\big] \\
&= \mathbb{E}_Z\big[P\big(\underline{F^{-1}_{\phi_i,T}(F_{\phi_i,Z}(G_{\phi_i}))} \leq g \big| Z, G_{\phi_i} < Z\big)\big]
\end{aligned}
$$

- $F^{-1}_{\phi_i,T}(F_{\phi_i,Z}(G_{\phi_i}))$: $G_{\phi_i} \to \tilde{G}_{\phi_i}$,
- $F^{-1}_{\phi_i,Z}(F_{\phi_i,T}(\tilde{G}_{\phi_i}))$: $\tilde{G}_{\phi_i} \to G_{\phi_i}$, and the two should equal

$$
\begin{aligned}
&= \mathbb{E}_Z\big[P\big(G_{\phi_i} \leq \underbrace{F^{-1}_{\phi_i,Z}(F_{\phi_i,T}(g))}_{u \text{ in CDF}} \,\big|\, \underbrace{Z, G_{\phi_i} < Z}_{\text{also use these conditions}}\big)\big] \\
&= \mathbb{E}_Z\big[F_{\phi_i,Z}(F^{-1}_{\phi,Z}(F_{\phi_i,T}(g)))\big] \\
&= \mathbb{E}_Z\big[F_{\phi_i,T}(g)\big] \\
&= F_{\phi_i,T}(g)
\end{aligned}
$$

application: stochastic beam search

- sequential model, conditional:

$$p_\theta(y_t|Y_{1:t-1}) = \frac{\exp\left(\phi_\theta(y_t|Y_{1:t-1})\right)}{\sum_{y'_t}\exp\left(\phi_\theta(y'_t|Y_{1:t-1})\right)}$$

- joint density:

$$p_\theta(Y_{1:t}) = \prod_\tau^t p_\theta(y_\tau|Y_{1:\tau-1})$$

$$= \frac{\exp\left(\phi_\theta(y_t|Y_{1:t-1})\right)}{\sum_{y'_t}\exp\left(\phi_\theta(y'_t|Y_{1:t-1})\right)} \times \frac{\exp\left(\phi_\theta(y_{t-1}|Y_{1:t-2})\right)}{\sum_{y'_{t-1}}\exp\left(\phi_\theta(y'_{t-1}|Y_{1:t-2})\right)} \times \cdots \times \frac{\exp\left(\phi_\theta(y_1)\right)}{\sum_{y'_1}\exp\left(\phi_\theta(y'_1)\right)}$$

- then we can deduce, for some t:

$$\phi_i \equiv \phi_\theta(y_t|Y_{1:t-1}) - \log\left(\sum_{y'_t}\exp\left(\phi_\theta(y'_t|Y_{1:t-1})\right)\right)$$

$$\text{because} \quad \exp(\phi_i) = \frac{\exp\left(\phi_\theta(y_t|Y_{1:t-1})\right)}{\sum_{y'_t}\exp\left(\phi_\theta(y'_t|Y_{1:t-1})\right)} = p_\theta(y_t|Y_{1:t-1})$$

Partial tree

- log-probability of partial tree Y^S:

$$\phi_S = \log p_\theta(Y^S) = \log \sum_{i \in S} \exp \phi_i$$

$$\implies \underbrace{\exp(\phi_S)}_{\text{prob of node } S} = p_\theta(Y^S) = \sum_{i \in S} \underbrace{\exp(\phi_i)}_{\text{prob of child}}$$

- Gumbel variable for partial tree correspond to node S:

$$G_{\phi_S} = \max_{i \in S} \{G_{\phi_i}\} \sim \text{Gumbel}(\phi_S)$$

$$\sim \text{Gumbel}\left(\log \sum_{i \in S} \exp \phi_i \right)$$

- this process by construction is **bottom-up**, obviously impractical for our purpose
- **the problem**: given G_{ϕ_S}, how can one transform $\{G_{\phi_i}\} \to \{\tilde{G}_{\phi_i}\}$ to maintain:

$$G_{\phi_S} = \max_{i \in S} \{\tilde{G}_{\phi_i}\} \sim \text{Gumbel}\left(\log \sum_{i \in S} \exp \phi_i \right)$$

Top-down Stochastic Beams search

- Given a partial tree Y^S (we know its value of ϕ_S and G_{ϕ_S}):
  - For each $S' \in \text{children}(S)$:

$$\phi_{S'} \leftarrow \phi_S + \log p_\theta(Y^{S'} | Y^S): \qquad \text{i.e., } \exp(\phi_{S'}) = \exp(\phi_S) p_\theta(Y^{S'} | Y^S)$$

S' extends partial tree length S by one token

$$G_{\phi_{S'}} \sim \text{Gumbel}(\phi_{S'})$$

  - $Z = \max\{G_{\phi_{S'}}\}$
  - For each $S' \in \text{children}(S)$:

$$\tilde{G}_{\phi_{S'}} \leftarrow -\log\left(\exp(-G_{\phi_S}) - \exp(-Z) + \exp(-G_{\phi_{S'}})\right)$$

$$\tilde{G}_{\phi_{S'}} \leftarrow -\log\left(\exp(-T) - \exp(-Z) + \exp(-G_{\phi_i})\right)$$

- BEAM $\leftarrow$ take top k of expansion according to $\{\tilde{G}_{\phi_{S'}}\}$ then expand to add $(Y^{S'}, \phi_{S'}, \tilde{G}_{\phi_{S'}})$