CS 239 Winter 2023
Deep Generative Models
Problem Set 1

October 17, 2023

SUNet ID:	jchan7
Name:	Jason Chan
Collaborators:	None

By turning in this assignment, I agree by the Stanford honor code and declare that all of this is my own work.

1 Maximum Likelihood Estimation and KL Divergence (10 points)

Show that the following equivalence holds:

$$\arg\max_{\theta\in\Theta} \mathbb{E}_{\hat{p}(x,y)}[\log p_\theta(y|x)] = \arg\min_{\theta\in\Theta} \mathbb{E}_{\hat{p}(x)}[D_{KL}(\hat{p}(y|x)||p_\theta(y|x))] \tag{1}$$

Given that

- $\hat{p}(y|x)$ is the empirical distribution of space of inputs $x \in \mathcal{X}$ and outputs $y \in \mathcal{Y}$
- $p_\theta(y|x)$ is a probabilistic classifier parameterized by θ

Answer. Let's examine the right hand side and show its equivalence to the left hand side

$$\begin{aligned}\arg\min_{\theta\in\Theta} \mathbb{E}_{\hat{p}(x)}[D_{KL}(\hat{p}(y|x)||p_\theta(y|x))]] &= arg\min_{\theta\in\Theta} \mathbb{E}_{\hat{p}(x)}[\mathbb{E}_{\hat{p}(y|x))}[log\hat{p}(y|x) - logp_\theta(y|x)]] \\ &= arg\min_{\theta\in\Theta} \mathbb{E}_{\hat{p}(x)}[\mathbb{E}_{\hat{p}(y|x))}[-logp_\theta(y|x)]] \\ &= arg\max_{\theta\in\Theta} \mathbb{E}_{\hat{p}(x)}[\mathbb{E}_{\hat{p}(y|x))}[logp_\theta(y|x)]] \\ &= arg\max_{\theta\in\Theta} \mathbb{E}_{\hat{p}(x,y)}[\log p_\theta(y|x)] \text{ due to law of total expectations}\end{aligned}$$

2 Logistic Regression and Naive Bayes (10 points)

Show that for any choice of θ there exists a γ such that

$$p_\theta(y|x) = p_\gamma(y|x) \tag{2}$$

Given that

- $p_\theta(x, y)$ is a mixture of k Gaussians where $y \in 1, ..., k$ is the mixture id and $x \in \mathbb{R}^n$
- $p_\theta(y) = \pi_y$ where $\sum_{y=1}^k \pi_y = 1$
- $p_\theta(x|y) = \mathcal{N}(x|\mu_y, \sigma^2 I)$
- We assume a diagonal covariance such that the model for the Gaussian's in the mixture is parameterised by $\theta = (\pi_1, \pi_2, ...\pi_k, \mu_1, \mu_2, ...\mu_k, \sigma)$ where $\pi_i \in \mathbb{R}_{++}$ and $\mu_i \in \mathbb{R}^n$ and $\sigma \in \mathbb{R}_{++}$
- The multi-class logistic regression model for predicting y from x as

$$p_\gamma(y|x) = \frac{exp(x^T w_y + b_y)}{\sum_{i=1}^k exp(x^T w_i + b_i)}$$

- The multi-class logistic model is parameterized by $\gamma = w_1, w_2, ..., w_k, b_1, b_2, ..., b_k$ where $w_i \in \mathbb{R}^n$ and $b_i \in \mathbb{R}$

Answer. Let's start with the left hand side and show equivalence to the right hand side. We first use bayes rule to convert $p_\theta(x|y)$ to $p_\theta(y|x)$ and get expressions for $p(y)$ and $p(x)$.

$$
\begin{aligned}
p_\theta(y|x) &= p_\theta(x|y)\frac{p(y)}{p(x)} \\
p_\theta(x|y) &= \mathcal{N}(x|\mu_y, \sigma^2 I) \\
&= \frac{1}{(2\pi)^{\frac{n}{2}}\det(\Sigma)^{\frac{1}{2}}}\exp\left(-\frac{1}{2}(x-\mu_y)^T\Sigma^{-1}(x-\mu_y)\right) \\
&= \frac{1}{(2\pi)^{\frac{n}{2}}\sigma^{2n\frac{1}{2}}}\exp\left(-\frac{1}{2}(x-\mu_y)^T\frac{1}{\sigma^2}(x-\mu_y)\right) \\
&= \frac{1}{(2\pi)^{\frac{n}{2}}\sigma^{n}}\exp\left(-\frac{1}{2\sigma^2}(x^Tx-2x^T\mu_y+\mu_y^T\mu_y)\right) \\
p_\theta(y) &= \pi_y \\
p_\theta(x) &= \sum_{i=1}^{k} p_\theta(x|y_i)p_\theta(y_i) \text{ (the marginal distribution of } x\text{)} \\
&= \sum_{i=1}^{k}\frac{1}{(2\pi)^{\frac{n}{2}}\sigma^{n}}\exp\left(-\frac{1}{2\sigma^2}(x^Tx-2x^T\mu_{y_i}+\mu_{y_i}^T\mu_{y_i})\right)\pi_{y_i} \\
p_\theta(y|x) &= \frac{\frac{1}{(2\pi)^{\frac{n}{2}}\sigma^{n}}\exp\left(-\frac{1}{2\sigma^2}(x^Tx-2x^T\mu_y+\mu_y^T\mu_y)\right)\pi_y}{\sum_{i=1}^{k}\frac{1}{(2\pi)^{\frac{n}{2}}\sigma^{n}}\exp\left(-\frac{1}{2\sigma^2}(x^Tx-2x^T\mu_{y_i}+\mu_{y_i}^T\mu_{y_i})\right)\pi_{y_i}} \\
&= \frac{\exp\left(-\frac{1}{2\sigma^2}(x^Tx-2x^T\mu_y+\mu_y^T\mu_y)\right)\pi_y}{\sum_{i=1}^{k}\exp\left(-\frac{1}{2\sigma^2}(x^Tx-2x^T\mu_{y_i}+\mu_{y_i}^T\mu_{y_i})\right)\pi_{y_i}}
\end{aligned}
$$

Now examine the contents in the exponential expression and find equivalence to $x^Tw_y + b_y$

$$
\begin{aligned}
-\frac{1}{2\sigma^2}(x^Tx-2x^T\mu_y+\mu_y^T\mu_y)+log(\pi_y) &= -\frac{1}{2\sigma^2}x^Tx+\frac{1}{\sigma^2}x^T\mu_y-\frac{1}{2\sigma^2}\mu_y^T\mu_y+log(\pi_y) \\
&= constant + \frac{\mu_y}{\sigma^2}x^T - \frac{\mu_y^T\mu_y}{2\sigma^2}+log(\pi_y) \\
w_y &= \frac{\mu_y}{\sigma^2} \\
b_y &= -\frac{\mu_y^T\mu_y}{2\sigma^2}+log(\pi_y)+constant
\end{aligned}
$$

Thus for any choice of $\theta = (\pi_1, \pi_2, ...\pi_k, \mu_1, \mu_2, ...\mu_k, \sigma)$ where $\pi_i \in \mathbb{R}_{++}$, there will exist an equivalent γ where $\gamma = w_1, w_2, ..., w_k, b_1, b_2, ..., b_k$ where $w_i \in \mathbb{R}^n$ and $b_i \in \mathbb{R}$

3 Conditional Independence and Parameterization (15 points)

Consider a collection of n discrete random variables $\{X_i\}_{i=1}^n$ where the number of outcomes for X_i is $|val(X_i)| = k_i$

1. Without any conditional independence assumptions, what is the total number of independent parameters needed to describe the joint distribution over $(X_1, ..., X_n)$?
2. Under what independence assumptions is it possible to represent the joint distribution $(X_1, ..., X_n)$? with $\sum_{i=1}^n (k_i - 1)$ total number of independent parameters?
3. Let $1, 2, ..., n$ denote the topological sort for a Bayesian network for the random variables $X_1, X_2, ..., X_n$. Let m be a positive integer in $1, 2, ..., n-1$. Suppose for every $i > m$, the random variable X_i is conditionally independent of all ancestors given the previous m ancestors in the topological ordering. Mathematically, we impose the independence assumptions $p(X_i|X_{i-1}, X_{i-1}, ...X_2, X_1) = p(X_i|X_{i-1}, X_{i-1}, ...X_{i-m})$ Derive the total number of independent parameters to specify the joint distribution over $(X_1, ..., X_n)$.

Answer. 1. Without any conditional independence assumptions we need $(\prod_{i=1}^n k_i)$ - 1 number of parameters

2. If we assume each random variable is mutually independent of each other then we can achieve $\sum_{i=1}^n (k_i - 1)$

3.

$$\text{For } X_{i\le m} \text{ the num. parameters for the joint probability} = (\prod_{i=0}^{m} k_i) - 1$$

$$\text{For } X_{i>m} \text{ the num. parameters for the joint probability} = \sum_{i=m+1}^{n} \left((k_i - 1) \prod_{j=i-m}^{i-1} k_j \right)$$

$$\text{The num. parameters for the full joint probability} = (\prod_{i=0}^{m} k_i) - 1 + \sum_{i=m+1}^{n} \left((k_i - 1) \prod_{j=i-m}^{i-1} k_j \right)$$

4 Auto-regressive Models (15 points)

Consider a set of n univariate *continuous* real-valued random variables $X_1, .., X_n$. You can access powerful neural networks $\{\mu_i\}_{i=1}^n$ and $\{\sigma_i\}_{i=1}^n$ that can represent any function $\mu_i : \mathbb{R}^{i-1} \to \mathbb{R}$ and $\sigma_i : \mathbb{R}^{i-1} \to \mathbb{R}_{++}$. For notation simplicity $\mathbb{R}^0 = \{0\}$. You choose to build the Gaussian auto-regressive model in the forward direction.

$$p_f(x_1, ..., x_n) = \prod_{i=1}^{n} p_f(x_i|x_{<i}) = \prod_{i=1}^{n} \mathcal{N}(x_i|\mu_i(x_{<i}), \sigma_i^2(x_{<i})) \tag{3}$$

where $x_{<i} = \begin{cases} (x_1, ..., x_{i-1})^T & \text{if } i > 1 \\ 0 & \text{if } i = 1 \end{cases}$

Your friend does the reverse order using equally powerful neural networks where $\{\hat{\mu}_i\}_{i=1}^n$ and $\{\hat{\sigma}_i\}_{i=1}^n$ that can represent any function $\hat{\mu}_i : \mathbb{R}^{n-i} \to \mathbb{R}$ and $\hat{\sigma}_i : \mathbb{R}^{n-i} \to \mathbb{R}_{++}$

$$p_r(x_1, ..., x_n) = \prod_{i=1}^n p_f(x_i|x_{>i}) = \prod_{i=1}^n \mathcal{N}(x_i|\hat{\mu}_i(x_{>i}), \hat{\sigma}_i^2(x_{>i})) \tag{4}$$

where $x_{>i} = \begin{cases} (x_{i+1}, ..., x_n)^T & \text{if } i < n \\ 0 & \text{if } i = n \end{cases}$

Do these models cover the same hypothesis space of distributions? In other words, given any choice of $\{\mu_i, \sigma_i\}_{i=1}^n$ does there always exist a choice of $\{\hat{\mu}_i, \hat{\sigma}_i\}_{i=1}^n$ such that $p_f = p_r$? If yes provide a proof. If no provide a counter example.

Answer. Consider the case of n=2.

$$p_f(x_1, x_2) = p(x_1)p(x_2|x_1)$$
$$p_r(x_1, x_2) = p(x_2)p(x_1|x_2)$$

Suppose $X_2 = X_1 + \epsilon$ where ϵ is some arbitrary but small value. In the forward case, let

$$p_f(x_1) = \mathcal{N}(x_1|0, 1)$$
$$p_f(x_2|x_1) = \mathcal{N}(x_2|\mu_2(x_1), \epsilon)$$

We can show that $p_f(x_2)$ is a mixture of two gaussians, which cannot be the case for $p_r(x_2)$

$$\begin{aligned} p_f(x_2) &= \int_{-\infty}^{\infty} p_f(x_1, x_2) dx_1 \\ &= \int_{-\infty}^{0} p_f(x_1)\mathcal{N}(x_2|0, \epsilon)dx_1 + \int_{0}^{\infty} p_f(x_1)\mathcal{N}(x_2|1, \epsilon)dx_1 \\ &= \frac{\mathcal{N}_0(x_2) + \mathcal{N}_1(x_2)}{2} \end{aligned}$$

Thus, these models don't cover the same hypothesis space.

5 Monte Carlo Integration (10 points)

1. An estimate $\hat{\theta}$ is an unbiased estimator of θ iff $\mathbb{E}[\hat{\theta}] = \theta$. Show that A is an unbiased estimator of p(x), where

$$A(z^{(1)}, ...z^{(k)}) = \frac{1}{k}\sum_{i=1}^k p(x|z^{(i)}) \text{ where } z^{(i)} \sim p(z)$$

Answer.

$$\begin{aligned}\mathbb{E}_{z^{(1)},...z^{(k)}}[A(z^{(1)},...z^{(k)})] &= \mathbb{E}_{z^{(i)}}\left[\frac{1}{k}\sum_{i=1}^{k}p(x|z^{(i)})\right] \\ &= \frac{1}{k}\sum_{i=1}^{k}\mathbb{E}_{z^{(i)}}[p(x|z^{(i)})] \text{ due to linearity of expectation} \\ &= \frac{1}{k}\sum_{i=1}^{k}p(x) \text{ because the expectation of a conditional is the marginal distr.} \\ &= p(x)\end{aligned}$$

2. Is log A an unbiased estimator of log p(x)? Prove why or why not

Answer. Jensen's inequality is defined as the below where f is a convex function

$$\begin{aligned} f(\mathbb{E}[X]) &\leq \mathbb{E}[f(x)] \\ \log(\mathbb{E}[A]) &\leq \mathbb{E}[log(A)] \\ \log(p(x)) &\leq \mathbb{E}[log(A)]\end{aligned}$$

Therefore log A is a not an unbiased estimator because $\mathbb{E}[log(A)]$ does not strictly equal $\log(p(x))$.

6 Programming (40 points)

1. What is the minimal bit representation for 50257 tokens?

Answer. If we give each token a unique ID then we need the bit representation for 50257 - 1 numbers, which is 16 bits (2 bytes).

2. If the number of possible tokens increases from 50257 to 60000, what is the increase in the number of parameters?

Answer.
- There will be proportional increase in embeddings and in the fully connected layer.
- Each token is represented by a 768 parameter embedding so we need 768(60000-50257) to describe the extra tokens.
- Each GPT2 output connects to the fully connected layer, so we need another $768(60000 - 50257)$.
- Finally, we need to account for additional parameters for bias softmax output of the fully connected layer of (60000-50257).

- In total we need $768(60000-50257)+768(60000-50257)+1(60000-50257) = 14,974,991$ more in parameters.

3. Programming see sample.py
4. Programming see likelihood.py
5. Programming see classifier.py
6. Programming see sample.py
7. (a) We are given that single token temperature scaling is defined as the below where temperature, $T > 0$

$$p_T(x_i|x_{<i}) \propto e^{\frac{\log p(x_i|x_{<i})}{T}}$$

What if we want to make likely sentences even more likely? In this case, we should consider scaling the joint temperature

$$p_T^{joint}(x_0x_1...x_M) \propto e^{\frac{\log p(x_0x_1...x_M)}{T}}$$

Does applying chain run with single token temperature scaling recover joint temperature scaling? In other words, determine if the following equation holds for arbitrary T?

$$\prod_{i=0}^{M} p_T(x_i|x_{<i}) \stackrel{?}{=} p_T^{joint}(x_0x_1...x_M)$$

Answer. Yes, applying chain rule with single token temperature scaling recovers joint temperature scaling. Let's look at the left hand side

$$\begin{aligned}
\prod_{i=0}^{M} p_T(x_i|x_{<i}) &= p_T(x_0) \cdot p_T(x_1|x_0) \cdot \ldots \cdot p_T(x_M|x_0, x_1, ..., x_{M-1}) \\
&\propto e^{\frac{\log p(x_0)}{T}} \cdot e^{\frac{\log p(x_1|x_0)}{T}} \cdot \ldots \cdot e^{\frac{\log p(x_M|x_{<M-1})}{T}} \\
&= e^{\frac{\log p(x_0)}{T} + \frac{\log p(x_1|x_0)}{T} + \ldots + \frac{\log p(x_M|x_{<M-1})}{T}} \\
&= e^{\frac{\log p(x_0) + \log p(x_1|x_0) + \ldots + \log p(x_M|x_{<M-1})}{T}}
\end{aligned}$$

Now look at the right hand side

$$\begin{aligned}
p^{joint}(x_0x_1...x_M) &\propto e^{\frac{\log\left(p(x_0)\cdot p(x_1|x_0)\cdot\ldots\cdot p(x_M|x_0,x_1,\ldots,x_{M-1}\right)}{T}} \\
&= e^{\frac{\log p(x_0) + \log p(x_1|x_0) + \ldots + \log p(x_M|x_{<M-1})}{T}}
\end{aligned}$$

Thus, applying chain rule with single token temperature scaling recovers joint temperature scaling.

(b) Programming see sample.py