CS 236 Homework 1

Instructors: Stefano Ermon and Aditya Grover

{ermon,adityag}@cs.stanford.edu

Available: 10/01/2018; Due: 23:59 PST, 10/15/2018

Problem 1: Maximum Likelihood Estimation and KL Divergence (10 points)

Let $\hat{p}(x, y)$ denote the empirical data distribution over a space of inputs $x \in \mathcal{X}$ and outputs $y \in \mathcal{Y}$. For example, in an image recognition task, x can be an image and y can be whether the image contains a cat or not. Let $p_\theta(y|x)$ be a probabilistic classifier parameterized by θ, e.g., a logistic regression classifier with coefficients θ. Show that the following equivalence holds:

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

where D_{KL} denotes the KL-divergence:

$$D_{\mathrm{KL}}(p(x) \| q(x)) = \mathbb{E}_{x \sim p(x)}[\log p(x) - \log q(x)].$$

Problem 2: Logistic Regression and Naive Bayes (12 points)

A mixture of k Gaussians specifies a joint distribution given by $p_\theta(x, y)$ where $y \in \{1, \ldots, k\}$ signifies the mixture id and $x \in \mathbb{R}^n$ denotes n-dimensional real valued points. The generative process for this mixture can be specified as:

$$p_\theta(y) = \pi_y \text{ , where } \sum_{y=1}^{k} \pi_y = 1 \tag{1}$$

$$p_\theta(x|y) = \mathcal{N}(x|\mu_y, \sigma^2 I). \tag{2}$$

where we assume a diagonal covariance structure for modeling each of the Gaussians in the mixture. Such a model is parameterized by $\theta = (\pi_1, \pi_2, \ldots, \pi_k, \mu_1, \mu_2, \ldots, \mu_k, \sigma)$, where $\pi_i \in \mathbb{R}_{++}$, $\mu_i \in \mathbb{R}^n$, and $\sigma \in \mathbb{R}_{++}$. Now consider the multi-class logistic regression model for directly predicting y from x as:

$$p_\gamma(y|x) = \frac{\exp(x^\top w_y + b_y)}{\sum_{i=1}^{k} \exp(x^\top w_i + b_i)}, \tag{3}$$

parameterized by vectors $\gamma = \{w_1, w_2, \ldots, w_k, b_1, b_2, \ldots, b_k\}$, where $w_i \in \mathbb{R}^n$ and $b_i \in \mathbb{R}$.
Show that for any choice of θ, there exists γ such that

$$p_\theta(y|x) = p_\gamma(y|x). \tag{4}$$

Problem 3: Conditional Independence and Parameterization (16 points)

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

1. **[2 points]** Without any conditional independence assumptions, what is the total number of independent parameters needed to describe the joint distribution over $(X_1, \ldots, X_n)$?

2. **[12 points]** Let $1, 2, \ldots, n$ denote the topological sort for a Bayesian network for the random variables $X_1, X_2, \ldots, X_n$. Let m be a positive integer in $\{1, 2, \ldots, n-1\}$. Suppose, for every $i > m$, the random variable X_i is conditionally independent of all ancestors given m previous ancestors in the topological ordering. Mathematically, we impose the independence assumptions

$$p(X_i|X_{i-1}, X_{i-2}, \ldots X_2, X_1) = p(X_i|X_{i-1}, X_{i-2}, \ldots X_{i-m})$$

for $i > m$. For $i \leq m$, we impose no conditional independence of X_i with respect to its ancestors.

Derive the total number of independent parameters to specify the joint distribution over $(X_1, \ldots, X_n)$?

3. **[2 points]** Under what independence assumptions is it possible to represent the joint distribution $(X_1, \ldots, X_n)$ with $\sum_{i=1}^{n}(k_i - 1)$ total number of independent parameters?

Problem 4: Autoregressive Models (12 points)

Consider a set of n univariate *continuous* real-valued random variables $(X_1, \ldots, X_n)$. You have access to 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 $\sigma_i : \mathbb{R}^{i-1} \to \mathbb{R}_{++}$. We shall, for notational simplicity, define $\mathbb{R}^0 = \{0\}$. You choose to build the following Gaussian autoregressive model in the *forward* direction:

$$p_f(x_1, \ldots, x_n) = \prod_{i=1}^{n} p_f(x_i|x_{<i}) = \prod_{i=1}^{n} \mathcal{N}(x_i|\mu_i(x_{<i}), \sigma_i^2(x_{<i})), \tag{5}$$

where $x_{<i}$ denotes

$$x_{<i} = \begin{cases} (x_1, \ldots, x_{i-1})^\top & \text{if } i > 1 \\ 0 & \text{if } i = 1. \end{cases} \tag{6}$$

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

$$p_r(x_1, \ldots, x_n) = \prod_{i=1}^{n} p_r(x_i|x_{>i}) = \prod_{i=1}^{n} \mathcal{N}(x_i|\hat{\mu}_i(x_{>i}), \hat{\sigma}_i^2(x_{>i})), \tag{7}$$

where $x_{>i}$ denotes

$$x_{>i} = \begin{cases} (x_{i+1}, \ldots, x_n)^\top & \text{if } i < n \\ 0 & \text{if } i = n. \end{cases} \tag{8}$$

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. Else, provide a counterexample.

[Hint: Consider the case where $n = 2$.]

Problem 5: Monte Carlo Integration (10 points)

A latent variable generative model specifies a joint probability distribution $p(x, z)$ between a set of observed variables $x \in \mathcal{X}$ and a set of latent variables $z \in \mathcal{Z}$. From the definition of conditional probability, we can express the joint distribution as $p(x, z) = p(z)p(x|z)$. Here, $p(z)$ is referred to as the prior distribution over z and $p(x|z)$ is the likelihood of the observed data condition on the latent variables. One natural objective for learning a latent variable model is to maximize the marginal likelihood of the observed data given by:

$$p(x) = \int_z p(x, z) \mathrm{d}z. \tag{9}$$

When z is high dimensional, tractable evaluation of the marginal likelihood is computationally intractable even if we can tractably evaluate the prior and the conditional likelihood for any given x and z. We can however

use Monte Carlo to estimate the above integral. To do so, we sample k samples from the prior $p(z)$ and our estimate is given as:

$$A(z^{(1)}, \ldots, z^{(k)}) = \frac{1}{k} \sum_{i=1}^{k} p(x|z^{(i)}), \text{ where } z^{(i)} \sim p(z). \tag{10}$$

1. **[5 points]** An estimator $\hat{\theta}$ is an unbiased estimator of θ if and only if $\mathbb{E}[\hat{\theta}] = \theta$. Show that A is an unbiased estimator of $p(x)$.

2. **[5 points]** Is $\log A$ an unbiased estimator of $\log p(x)$? Explain why or why not.

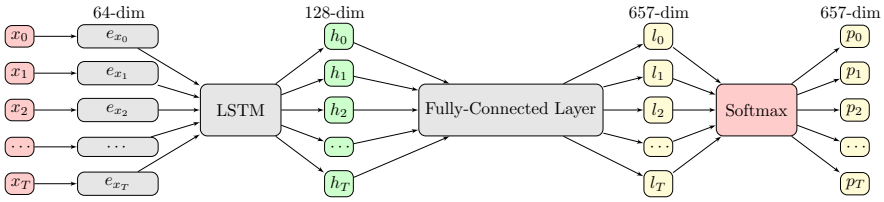

Figure 1: The architecture of our model. T is the sequence length of a given input. x_i is the index token. e_{x_i} is the trainable embedding of token x_i. h_i is the output of LSTMs. l_i is the logit and p_i is the probability. Nodes in gray contain trainable parameters.

Problem 6: Programming assignment (40 points)

In this programming assignment, we will use an autoregressive generative model to generate text from machine learning papers. In particular, we will train a character-based recurrent neural network (RNN) to generate paragraphs. The training dataset consists of all papers published in NIPS 2015.[1] The model used in this assignment is a four-layer Long Short-Term Memory (LSTM) network. LSTM is a variant of RNN that performs better in modeling long-term dependencies. See this blog post for a friendly introduction.

There are a total of 657 different characters in NIPS 2015 papers, including alphanumeric characters as well as many non-ascii symbols. During training, we first convert characters to a number in the range 0 to 656. Then for each number, we use a 64-dimensional trainable vector as its embedding. The embeddings are then fed into a four-layer LSTM network, where each layer contains 128 units. The output vectors of the LSTM network are finally passed through a fully-connected layer to form a 657-way softmax representing the probability distribution of the next token. See Figure 1 for an illustration.

Training such models can be computationally expensive, requiring specialized GPU hardware. In this particular assignment, we provide a pretrained generative model. After loading this pretrained model into *PyTorch*, you are expected to implement and answer the following questions.

1. **[4 points]** Suppose we wish to find an efficient bit representation for the 657 characters. That is, every character is represented as $(a_1, a_2, \cdots, a_n)$, where $a_i \in \{0, 1\}, \forall i = 1, 2, \cdots, n$. What is the minimal n that we can use?

2. **[6 points]** If the size of vocabulary increases from 657 to 900, what is the increase in the number of parameters? [Hint: You don't need to consider parameters in the LSTM module in Fig. 1.]

   Note: For the following questions, you will need to complete the starter code in designated areas. After the code is completed, run `main.py` to provide related files for submission. Run the script `./make_submission.sh` to generate `hw1.zip` and upload it to GradeScope.

3. **[10 points]** In the starter code, complete the method `sample` in `model.py` to generate 5 paragraphs each of length 1000 from this model.

4. **[10 points]** Complete the method `compute_prob` in `model.py` to compute the log-likelihoods for each string. Plot a separate histogram of the log-likelihoods of strings within each file.

5. **[10 points]** Can you determine the category of an input string by only looking at its log-likelihood? We now provide new strings in `snippets.pkl`. Try to infer whether the string is generated randomly, copied from Shakespeare's work or retrieved from NIPS publications. You will need to complete the code in `main.py`.

[1] Neural Information Processing Systems (NIPS) is a top machine learning conference.