UNIVERSITY OF PENNSYLVANIA

ESE 546: PRINCIPLES OF DEEP LEARNING

FALL 2020

[11/30] HOMEWORK 5

DUE: 12/10 THURSDAY 1.30P ET

Changelog

Instructions

Read the following instructions carefully before beginning to work on the homework.

- You will submit solutions typeset in LaTeX on Gradescope (strongly encouraged). You can use hw_template.tex on Canvas in the "Homeworks" folder to do so. If your handwriting is *unambiguously legible*, you can submit PDF scans/tablet-created PDFs.
- Please start a new problem on a fresh page and mark all the pages corresponding to each problem. Failure to do so may result in your work not graded completely.
- Clearly indicate the name and Penn email ID of all your collaborators on your submitted solutions.
- For each problem in the homework, you should mention the total amount of time you spent on it.
- You can be informal while typesetting the solutions, e.g., if you want to draw a picture feel free to draw it on paper clearly, click a picture and include it in your solution. Do not spend undue time on typesetting solutions.
- You will see an entry of the form "HW 5 PDF" where you will upload the PDF of your solutions. You will also see an entry named "HW 5 Problem 2 Code" where you will upload the code for Problem 2. **For each programming problem, you should create a fresh Google Colab notebook**. This notebook should contain all the code to reproduce the results of the problem and you will upload the .ipynb file obtained from Colab. Name your notebook to be "pennkey_hw5_problem4.ipynb", e.g., I will name my code for Problem 2 as "pratikac_hw5_problem2.ipynb".
- **Remember that you should include all the relevant plots in the PDF, without doing so you will not get full credit. Your PDF solutions should be completely self-contained, we should not have have to look at/run the Colab notebook to check your solutions.**

Credit The points for the problems add up to 100. You only need to solve for 50 points to get full credit, i.e., your final score will be $\min(\text{your total points}, 50)$.

Problem 1 (40 points, background on variational inference). Given some distribution

$$p(w) = \frac{e^{-\beta \Phi(w)}}{Z}$$

variational inference approximates $p(w)$ using some distribution of our choice, denoted by $q(w)$, by solving the problem

$$\min_{q \in \mathcal{Q} \subset \mathcal{P}(\mathcal{W})} \text{KL}(q \,||\, p). \tag{1}$$

The set $\mathcal{Q}$ is called the variational family which is typically a subset of all probability distributions $\mathcal{P}(\mathcal{W})$ on the domain $\mathcal{W} \ni w$. We should take $\mathcal{Q}$ to be small enough that we can easily solve the optimization problem (1) but still large enough to obtain a small $\text{KL}(q \,||\, p)$.

The quantity

$$\mathcal{F}(q) := -\beta \sum_{w \in \mathcal{W}} q(w)\ \Phi(w) - \sum_{w \in \mathcal{W}} q(w)\ \log q(w)$$

is called the free-energy by physicists. For distributions p, q supported on some discrete set, we can rewrite $\text{KL}(q \,||\, p)$ as

$$\text{KL}(q \,||\, p) = \beta \sum_{w \in \mathcal{W}} q(w)\ \Phi(w) + \sum_{w \in \mathcal{W}} q(w)\ \log q(w) + \log Z(\beta). \tag{2}$$

Since $\text{KL}(q \,||\, p) \geq 0$ we immediately have a lower bound for $\log Z(\beta)$

$$\max_{q \in \mathcal{Q}} \mathcal{F}(q) \leq \log Z(\beta) = \log \sum_{w \in \mathcal{W}} e^{-\beta \Phi(w)}. \tag{3}$$

This is a very useful bound because the log-partition function $\log Z(\beta)$ contains essentially all information about the probability distribution $p(w)$. For example, the average value of $\Phi(w)$ is given by

$$\mathop{\mathbb{E}}_{w \sim p(w)} [\Phi(w)] = -\frac{\partial \log Z(\beta)}{\partial \beta} \approx \mathop{\mathbb{E}}_{w \sim q(w)} [\Phi(w)]\,. \tag{4}$$

The last approximation is reasonable because $q(w) \approx p(w)$ if we have done a good job of minimizing $\text{KL}(q \,||\, p)$. If $\mathcal{Q} = \mathcal{P}(\mathcal{W})$, we certainly have

$$\max_{q \in \mathcal{P}(\mathcal{W})} \mathcal{F}(q) = \log Z(\beta).$$

because in this case the minimum of $\text{KL}(q \,||\, p)$ is zero.

> **Note:** You will notice that the bound on the log-partition function is very similar to the bound on $p(x)$ we obtained when we derived ELBO (x denotes data and z denotes the representation in the equation)
>
> $$\text{ELBO}(q) = \mathop{\mathbb{E}}_{z \sim q(z|x)} [\log p(x|z)] - \text{KL}\left(q(z|x) \,||\, p(z)\right) \leq \log p(x).$$
>
> This is not be surprising because $p(x)$ is indeed the normalizing constant in Bayes law
>
> $$p(z|x) = \frac{p(x,z)}{p(x)} = \frac{p(x,z)}{\sum_z p(x,z)}.$$

(a) **(5 points)** Prove the expression in (2).

(b) **(5 points)** Prove the equality in (4).

(c) **(10 points)** The mean-field assumption in variational inference means that we pick the family of distributions that factors over the variable w. One example is,

$$\mathcal{Q}_1 = \left\{ q(w) : q(w) = \prod_{i=1}^{N} q_i(w_i) \right\}.$$

where $q_i(w_i)$ is a distribution which only depends on the i^{th} variable w_i. The distributions q_i can also be the same in which case we would have

$$\mathcal{Q}_2 = \left\{ q(w) : q(w) = \prod_{i=1}^{N} q(w_i) \right\}.$$

As you can imagine, it is difficult to pick a variational family without knowing much about the distribution $p(w)$. This problem will consider a simple example. Let true data be generated by

$$p(w) \propto 1 - \prod_{i=1}^{N} w_i$$

where $w_i \in \{0, 1\}$. Clearly the configuration $w_i = 1$ for all $i = 1, \ldots, N$ has zero probability, all other configurations have equal probability $1/(2^N - 1)$. This would seem like a simple distribution $p(w)$ but the highly correlated part (one configuration) makes it difficult to approximate using mean-field families. Let's pick Q_2 as the family. Find the minimizer of

$$q^* = \min_{q \in \mathcal{Q}_2} \text{KL}(q \,||\, p).$$

Does $q^*(w)$ look similar to $p(w)$?

(d) **(20 points)** Now pick a different mean-field family $\mathcal{Q}_2$ and solve

$$q^* = \min_{q \in \mathcal{Q}_1} \text{KL}(q \,||\, p).$$

It will help to notice that $q(1, 1, \ldots, 1) = 0$ is a necessary condition for the minimizer (why?). Notice that the entire problem is symmetric in all the variables. So one way of ensuring our solution satisfies the necessary condition is to simply set $q_1(w_1) = 1 - w_1$. Now does $q^*(w)$ look similar to $p(w)$?

Problem 2 (60 points. Variational Auto-Encoder for MNIST. Do this on Colab/AWS but this can also be done on your laptop.). In this problem we will train a variational auto-encoder for generating MNIST digits. The setup is exactly the same as that of Section 3 in the assigned reading "Auto-Encoding Variational Bayes" (https://arxiv.org/abs/1312.6114) by Kingma & Welling.

(i) **(0 points)** Create the dataset. Use 1000 images of each MNIST digit; this gives a total of 10,000 images. Sub-sample the MNIST images to 14×14 size. You will also binarize the MNIST dataset for this problem: if a pixel is greater than 128 set it to 1, else set it to zero.

(ii) **(40 points)** The encoder will consist of two fully-connected layers, the first has $14 \times 14 = 196$ inputs and 128 outputs (and tanh nonlinearity) and the second layer has 128 inputs and 16 outputs (and no nonlinearity). Therefore, the latent factor $z \in \mathbb{R}^8$ (8 output neurons for the mean and 8 more for the standard deviation). The decoder takes in as input $z \in \mathbb{R}^8$, pushes it through one layer with 128 outputs (and tanh nonlinearity) and then another layer with 196 output neurons (with sigmoid nonlinearity). Note that the final nonlinearity of the decoder should be sigmoid because we want to reconstruct binarized MNIST images. See Sections C1 & C2 in Appendix of the above paper for more details.

Let the parameters of the encoder be u and the parameters of the decoder be v. We will maximize the objective

$$\frac{1}{n}\sum_{i=1}^{n}\ell(u,v;x^i)$$

for images $\left\{x^1,\ldots,x^n\right\}$ with respect to its parameters u, v where

$$\ell(u,v;x)=\mathop{\mathbb{E}}_{z\sim p_\phi(z|x)}\left[\log p_v(x|z)\right]-\mathrm{KL}\left(p_u(z|x)\,||\,N(0,I)\right). \tag{5}$$

and x is one MNIST datum. We will set

$$p_u(z|x)=N\left(\mu_u(x);\sigma_u(x)^2I\right)$$

where $\left[\mu_u(x),\sigma_u(x)\right]\in\mathbb{R}^{16}$ is the output of encoder upon feeding the image x. We are modeling $p_u(z|x)$ as a Gaussian distribution and the neural network predicts the mean $\mu_u(x)$ of the distribution and the diagonal of the covariance $\sigma_u(x)$. The KL-divergence is

$$\mathrm{KL}\left(p_u(z|x)\,||\,N(0,I)\right)=-\frac{1}{2}\sum_{i=1}^{8}\left(1+\log(\sigma_u(x)_i^2)-\mu_u(x)_i^2-\sigma_u(x)_i^2\right).$$

We will use a Bernoulli distribution to model $p_v(x|z)$ because we are using the binarized MNIST dataset:

$$\log p_v(x|z)=\sum_{i=1}^{8}x_i\log y_i+(1-x_i)\log(1-y_i)$$

where $\mathbb{R}^{196}\ni y=$ output of the decoder.

Train the auto-encoder using the re-parametrization trick for computing the gradient of the u and standard back-prop for computing the gradient of the $\log p_v(x|z)$. You should use 2 samples to compute the expectation over $z\sim p_u(z|x)$.

(iii) **(5 points)** Plot the first and second term of ELBO in (5) separately as a function of the number of weight updates.

(iv) **(5 points)** Pick 8 MNIST images, run them through the encoder and the decoder to plot the output of the decoder side-by-side with the original images.

(v) **(5 points)** Sample from the generative model directly: sample $z\in\mathbb{R}^8$ from a standard Gaussian distribution and run the decoder network for this z to synthesize an MNIST image. Plot the synthesized images for a few different samples of z; remember that you can also use a mini-batch of latent variables z to synthesize a few images in one forward-prop of the decoder.

(vi) **(5 points)** We will next compute the validation performance of a generative model. Compute the average validation log-likelihood $\log p(x|z)$ for 100 synthesized images after every 100 weight updates. Draw a plot of the reconstruction log-likelihood term in ELBO on the training mini-batches (one point after every weight update) and compare it with the validation log-likelihood as training progresses (one point after every 100 weight updates).