UNIVERSITY OF PENNSYLVANIA

ESE 546: PRINCIPLES OF DEEP LEARNING

FALL 2020

(12/12) PRACTICE FINAL EXAM

DURATION: 180 MINUTES

Read the following instructions carefully before you begin.

- This is a closed book exam. You are allowed to use one A4/Letter paper cheat sheet (front and back). You may not use laptops, phones or the Internet during this exam. You may not discuss with your peers.
- You will have 180 minutes from the time you start the exam on Gradescope to upload your solutions. We will be very strict as regards to late submissions. They will only receive 50% of the credit, so make sure you keep enough buffer to upload solutions into Gradescope.
- You should use pen and paper (do not use a pencil because it might not be readable when scanned) to write your solutions and click pictures and upload your solutions as a PDF. Please make sure to write in legible handwriting. You can also use a tablet to write your solutions. Please make sure your solutions are legible.
- Begin each problem on a fresh page. Solutions that are not correctly annotated on the Gradescope outline will not receive credit.
- Each question mentions how short/long we would like your responses to be. DO NOT write excessively verbose answers.
- None of the questions require long derivations. If you find yourself going through lots of equations reconsider your approach or consider moving on to the next question.
- If you are stuck, explain your answers and what you are trying to do clearly. We will give partial credit if you are on the right track.
- The questions are NOT arranged in order of difficulty, try to attempt every question. **This exam has 7 problems**.
- In case of emergency, send the Instructor an email at pratikac@seas.upenn.edu.

The practice exam has 63 points and is therefore about two-thirds as long as the final exam. You should aim to complete the practice final exam well within 120 minutes. Solutions to the practice exam will be available on Sunday night ET.

Problem 1. (22 points)

1. **(5 points)** Explain how you will evaluate the gradient

$$\nabla_u \mathop{\mathbb{E}}_{x \sim q_u(w)} [\varphi(w)]$$

using the Reparametrization Trick.

Answer: The Reparametrization Trick parametrizes $w = g(u, \epsilon)$ for some deterministic function g and noise $\epsilon \sim N(0, I)$. So the expectation can be rewritten as

$$\begin{aligned}
\nabla_u \mathop{\mathbb{E}}_{w \sim q_u(w)} [\varphi(w)] &= \nabla_u \mathop{\mathbb{E}}_{\epsilon \sim N(0,I)} [\varphi(g(\theta, \epsilon))] \\
&= \mathop{\mathbb{E}}_{\epsilon \sim N(0,I)} [\nabla_u \varphi(g(u, \epsilon))] \\
&= \mathop{\mathbb{E}}_{\epsilon \sim N(0,I)} [\nabla_w \varphi(w) \ \nabla_u g(u, \epsilon))] .
\end{aligned}$$

2. **(10 points)** Exponential families are probability distributions which can be written as

$$p_\theta(w) = e^{\phi(w)^\top \theta - A(\theta)}.$$

where θ are the parameters of the distribution and show up in the exponent as a linear term $\phi(x)^\top \theta$ and an additive term $A(\theta)$. They are particularly useful in variational inference because most quantities of interest are easily expressed in terms of $A(\theta)$.

(a) **(5 points)** First show that

$$\nabla_\theta \ A(\theta) = \mathop{\mathbb{E}}_{x \sim p_\theta(x)} [\phi(x)] .$$

Hint: Use the fact that $p(x)$ is a probability distribution.

Answer: Since $p(x)$ is a probability distribution, we have

$$\begin{aligned}
e^{A(\theta)} &= \int e^{\phi^\top(x)\theta} \, \mathrm{d}x \\
\Rightarrow e^{A(\theta)} \nabla A(\theta) &= \nabla_\theta \int e^{\phi^\top(x)\theta} \, \mathrm{d}x \\
e^{A(\theta)} \nabla A(\theta) &= \int e^{\phi^\top(x)\theta} \, \phi \, \mathrm{d}x \\
\nabla A(\theta) &= \frac{1}{e^{A(\theta)}} \int e^{\phi^\top(x)\theta} \, \phi(x) \, \mathrm{d}x \\
\nabla A(\theta) &= \mathop{\mathbb{E}}_{x \sim p_\theta(x)} [\phi(x)] .
\end{aligned}$$

(b) **(5 points)** Calculate the entropy of the distribution

$$H(p_\theta) = -\int p_\theta(x)\ \log p_\theta(x)\ \mathrm{d}x.$$

Your answer will be in terms of θ, $A(\theta)$ and $\nabla\, A(\theta)$.

Answer:

$$\begin{aligned} H &= -\int e^{\phi^\top(x)\theta - A(\theta)}\ \ \Big(\phi^\top(x)\ \theta - A(\theta)\Big)\,\mathrm{d}x \\ &= -\nabla_\theta\, A(\theta)^\top\ \theta + A(\theta). \end{aligned}$$

3. **(3 points)** If a convolutional layer in a neural network has c_1 input channels, $k \times k$ kernel and c_2 output channels then what is the total number of trainable parameters in this layer? Assume that there is no bias.

Answer:

$$c_1\ c_2\ k^2$$

trainable parameters.

4. **(4 points)** Consider an m-strongly convex function $f(x)$ with L-Lipschitz gradients. Find a quadratic lower *and* upper bound for $f(y)$ for all x expressed in terms of $f(x)$ and $\nabla\, f(x)$. Your bounds should be quadratic functions of $\|y - x\|$. Hint: Use the descent lemma. No explanation or proof is necessary.

Answer: The lower bound is

$$f(y) \le f(x) + \nabla\, f(x)^\top\ (y - x) + \frac{L}{2}\|x - y\|^2.$$

The upper bound is

$$f(y) \ge f(x) + \nabla\, f(x)^\top\ (y - x) + \frac{m}{2}\|x - y\|^2.$$

Problem 2 (9 points). Answer whether the following statements are true or false. Explain every answer in 1-2 sentences.

1. **(3 points)** The variance of the updates in SGD

$$\operatorname*{Var}_{\omega_t}\left(w^{t+1} \mid w^t\right)$$

decreases linearly with the batch-size b. Recall that the SGD update is given by

$$w^{t+1} = w^t - \frac{\alpha}{b} \sum_{k=1}^{b} \nabla f_{\omega_t^k}(w^t).$$

where $\omega_t^k \in \{1, \ldots, n\}$.

Answer: True. The variance goes down as $1/b$.

2. **(3 points)** Consider a set of classifiers which includes all linear classifiers that use different choices of strict subsets of the components of input vectors $x \in \mathbb{R}^d$. The VC-dimension of this hypothesis class cannot be more than $d + 1$.

Answer: True. We can imagine setting the weights corresponding to the complement of the active components of the input to zero. This is a linear hypothesis space and its VC-dimension is at most $d + 1$.

3. **(3 points)** Suppose the non-linearity in a three-layer neural network

$$\hat{y}_i = A\, \sigma(B\, \sigma(C\, x_i)).$$

is quadratic

$$\sigma(x) = x^2,$$

the weight matrices are $A \in \mathbb{R}^{10 \times m}, B \in \mathbb{R}^{m \times m}, C \in \mathbb{R}^{m \times d}$ and $x_i \in \mathbb{R}^d$ is the data. Consider the case when Nature's true model that gives the dataset $\{(x_i, y_i)\}_{i=1,\ldots,n}$ is a polynomial of degree 3. If we train our neural network using the MSE loss $\frac{1}{n} \sum_{i=1}^{n} \|y_i - \hat{y}_i\|_2^2$ without any regularization then we can get zero training error on all finite datasets.

Answer: False. Imagine the true data coming from $y = x^3$. The two quadratic non-linearities (without biases) will force the output $\hat{y}$ to always be a polynomial of degree 4.

Problem 3. (6 points)

1. **(3 points)** Suppose we trained an encoder $q_u(z \mid x)$ and a decoder $p_v(x \mid z)$ using a variational auto-encoder. How can one use these to sample new data that are not in the training dataset?

Answer: We do not use the encoder at test time. First, we sample a new z from the prior $p(z)$ and then we use the decoder to produce a new x. If the decoder outputs the mean and the variance of a Gaussian $p_v(x|z)$ we use those sufficient statistics to sample a new image x.

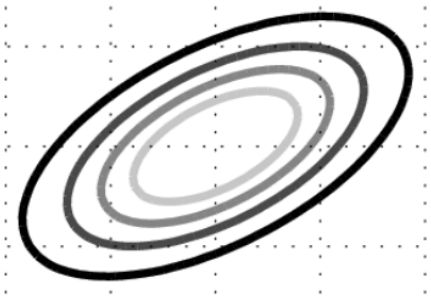
2. **(3 points)** Compute the KL divergence between two Bernoulli distributions

$$\text{KL}(q \,||\, p).$$

one with probability of success q and the other with probability of success p.

Answer: $KL(q||p) = q \log \frac{q}{p} + (1-q) \log \frac{1-q}{1-p}$

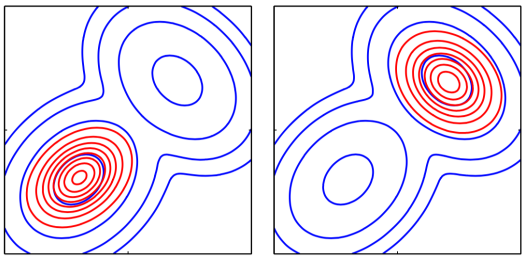
Problem 4. (6 points) Consider the following contour plot. Draw the typical updates of gradient descent, Polyak's heavy ball method and gradient descent with Nesterov's acceleration on this plot from the same initial condition. Clearly label each vector in your plot. You can pick any point (except the global minimum) as your initial condition.



Answer: The updates for all three, gradient descent, Polyak's heavy ball and Nesterov's updates are in the Lecture notes.

For gradient descent, note that at each iteration the algorithm makes a step that is exactly orthogonal to the contour at that point. The step-size determines how far the algorithm moves in this direction before changing it in the next iteration.

Problem 5. (4 points) We can minimize either $\text{KL}(q \,||\, p)$ or $\text{KL}(p \,||\, q)$ to approximate a given distribution $p(x)$ using candidate distributions $q \in \mathcal{Q}$ from a variational family. In the following two pictures, explain which one of the two loss functions was used to compute the approximation. The blue curves denote the contours of $p(x)$ and the red curves denote the contours of $q^*(x)$.



Answer: We discussed in class how $\mathrm{KL}(q \mid\mid p) = \int q(x) \log \frac{q(x)}{p(x)} \, \mathrm{d}x$ ensures that $q(x)$ is zero if $p(x)$ is zero (otherwise the KL-divergence is infinite). The other form $\mathrm{KL}(p \mid\mid q)$ instead ensures that $q(x)$ also puts probability mass in areas where $p(x)$ has no probability mass.

Therefore the two pictures above both correspond to $\mathrm{KL}(q \mid\mid p)$. They are simply two minima obtained from two different initializations.

Problem 6 (15 points). Given a finite dataset $D = \{(x^i, y^i)\}_{i=1,\ldots,n}$ the cross entropy loss of a model given by parameters w is given by

$$\ell(w) = -\frac{1}{n} \sum_{i=1}^{n} \log p_w(y^i \mid x^i).$$

(a) **(10 points)** Show that this can be written as

$$\ell(w) = \mathrm{KL}(p \mid\mid p_w).$$

where

$$p(x, y) = \frac{1}{n} \sum_{i=1}^{n} \delta_{x=x^i,\, y=y^i}$$

is the Dirac-delta distribution of the given dataset. In other words, training a neural network using the cross-entropy loss is akin to computing a variational approximation to the empirical data distribution.

(b) **(5 points)** Following this line of thinking, argue whether

$$\mathrm{KL}(p_w \mid\mid p)$$

is a good loss function or not. Explain why this function will overfit to the training data.

Answer: (Part a) We have

$$\begin{aligned}
\ell(w) &= \mathrm{KL}(p \,||\, p_w) \\
&= \int p(x,y)\ \log \frac{p(x,y)}{p_w(y \mid x)\, p_w(x)}\ \mathrm{d}x\ \mathrm{d}y \qquad (a) \\
&= \int \frac{1}{n} \sum_{i=1}^{n} \delta_{x=x^i, y=y^i} \log \delta_{x=x^i, y=y^i}\ \mathrm{d}x\mathrm{d}y \\
&\quad - \int \frac{1}{n} \sum_{i=1}^{n} \delta_{x=x^i, y=y^i}\ (\log p_w(y \mid x) + \log p_w(x))\, \mathrm{d}x\mathrm{d}y \\
&= 0 - \frac{1}{n} \sum_{i=1}^{n} \log p_w(y^i \mid x^i) - \log p_w(x^i) \qquad (b, c).
\end{aligned}$$

(a) note that this integrand is non-zero only at values $\left\{(x^i, y^i)\right\}$

(b) is true because the entropy of a Dirac delta distribution is zero.

(c) We do not care take into account the likelihood of datum x when we learn the classifier p_w and consider each data equally likely under both Nature's distribution and our classifier's distribution $p_w(x_i)$. The third term is therefore taken to be a constant.

Notes. In other words, minimizing the cross-entropy loss while training a supervised model can be thought of as a variational optimization problem

$$w^* = \operatorname*{argmin}_{p_w} \mathrm{KL}(p \,||\, p_w)$$

where p is the empirical data distribution. This is surprising although a bit inconsequential; it becomes more interesting when you think of the population risk (which is given by the same objective with p being Nature's distribution).

(Part b) To answer the second question, note that $\mathrm{KL}(p \,||\, p_w)$ encourages p_w to also put probability mass in areas where $p(x) = \frac{1}{n} \sum_{i=1}^{n} \delta_{x=x^i,\ y=y^i}$ does not put any mass, i.e., outside the training samples. This is crucial for generalization. If we instead minimize

$$\mathrm{KL}(p_w \,||\, p)$$

the classifier will only put probability mass in places where p has non-zero mass, i.e., on the training data. It will simply memorize all labels in the dataset and its generalization performance will be poor.

Problem 7 (3 points). Suppose you have a discrete-state discrete-time Markov Chain with n states. We know that this Markov chain has a unique steady-state distribution. If you

additionally know that the transition matrix $P \in \mathbb{R}^{n \times n}$ is doubly stochastic, i.e., both its rows and columns sum to 1, then what is that unique steady-state distribution?

Answer: We are given that the steady-state distribution is unique. We know that this distribution satisfies $\pi^* = P^\top \pi^*$. Guess that $\pi^* = \Big[1/n, \ldots, 1/n\Big]$. This is a legitimate distribution because it sums up to 1 and it also satisfies the steady-state constraint. Hence the steady-state distribution is uniform.

END OF EXAM