EE239AS, Winter 2018 **Homework #3**

Neural Networks & Deep Learning Prof. J.C. Kao

UCLA ECE TAs: W. Chuang & M. Kleinman & K. Liang & A. Wickstrom

Due Wednesday, 6 Feb 2019, by 11:59pm to Gradescope.
100 points total.

For this HW, you should not be performing any tensor matrix multiplies or calculating 4D tensor derivatives. You will need to know the following backpropagated derivative for matrix-matrix multiplies, which you can use without proof:

Consider $\mathbf{Z} = \mathbf{XY}$. If we have an upstream derivative, $\partial\mathcal{L}/\partial\mathbf{Z}$, then they are backpropagated through this operation as:

$$\begin{aligned}\frac{d\mathcal{L}}{\partial\mathbf{X}} &= \frac{\partial\mathcal{L}}{\partial\mathbf{Z}}\mathbf{Y}^T \\ \frac{d\mathcal{L}}{\partial\mathbf{Y}} &= \mathbf{X}^T\frac{\partial\mathcal{L}}{\partial\mathbf{Z}}\end{aligned}$$

1. (15 points) **Backpropagation for autoencoders.** In an autoencoder, we seek to reconstruct the original data after some operation that reduces the data's dimensionality. We may be interested in reducing the data's dimensionality to gain a more compact representation of the data.

   For example, consider $\mathbf{x} \in \mathbb{R}^n$. Further, consider $\mathbf{W} \in \mathbb{R}^{m\times n}$ where $m < n$. Then $\mathbf{Wx}$ is of lower dimensionality than $\mathbf{x}$. One way to design $\mathbf{W}$ so that $\mathbf{Wx}$ still contains key features of $\mathbf{x}$ is to minimize the following expression

   $$\mathcal{L} = \frac{1}{2}\left\|\mathbf{W}^T\mathbf{Wx} - \mathbf{x}\right\|^2$$

   with respect to $\mathbf{W}$. (To be complete, autoencoders also have a nonlinearity in each layer, i.e., the loss is $\frac{1}{2}\left\|f(\mathbf{W}^T f(\mathbf{Wx})) - x\right\|^2$. However, we'll work with the linear example.)

   (a) (3 points) In words, describe why this minimization finds a $\mathbf{W}$ that ought to preserve information about $\mathbf{x}$.

   (b) (3 points) Draw the computational graph for $\mathcal{L}$.

   (c) (3 points) In the computational graph, there should be two paths to $\mathbf{W}$. How do we account for these two paths when calculating $\nabla_{\mathbf{W}}\mathcal{L}$? Your answer should include a mathematical argument.

   (d) (6 points) Calculate the gradient: $\nabla_{\mathbf{W}}\mathcal{L}$.

2. (20 points) **Backpropagation for Gaussian-process latent variable model.** An important component of unsupervised learning is visualizing high-dimensional data in low-dimensional spaces. One such nonlinear algorithm to do so is from Lawrence, NIPS 2004, called GP-LVM. GP-LVM optimizes the maximum-likelihood of a probabilistic model. We won't get into the details here, but rather to the bottom line: in this paper, a log-likelihood

has to be differentiated with respect to a matrix to derive the optimal parameters.

To do so, we will use apply the chain rule for multivariate derivatives via backpropagation. The log-likelihood is:

$$\mathcal{L} = -c - \frac{D}{2}\log|\mathbf{K}| - \frac{1}{2}\mathrm{tr}(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T)$$

where $\mathbf{K} = \alpha\mathbf{X}\mathbf{X}^T + \beta^{-1}\mathbf{I}$ and c is a constant. To solve this, we'll take the derivatives with respect to the two terms with dependencies on $\mathbf{X}$:

$$\begin{aligned}
\mathcal{L}_1 &= -\frac{D}{2}\log|\alpha\mathbf{X}\mathbf{X}^T + \beta^{-1}\mathbf{I}| \\
\mathcal{L}_2 &= -\frac{1}{2}\mathrm{tr}\left((\alpha\mathbf{X}\mathbf{X}^T + \beta^{-1}\mathbf{I})^{-1}\mathbf{Y}\mathbf{Y}^T\right)
\end{aligned}$$

Hint: To receive full credit, you will be required to show all work. You may use the following matrix derivatives without proof:

$$\begin{aligned}
\frac{\partial \mathrm{tr}\mathbf{X}}{\partial \mathbf{X}} &= \mathbf{I} \\
\frac{\partial \log|\mathbf{K}|}{\partial \mathbf{K}} &= \mathbf{K}^{-T} \\
\frac{\partial \mathcal{L}}{\partial \mathbf{K}} &= -\mathbf{K}^{-T}\frac{\partial \mathcal{L}}{\partial \mathbf{K}^{-1}}\mathbf{K}^{-T}
\end{aligned}$$

That is, the first equation tells you how to backpropagate through $\mathrm{tr}(\cdot)$, the second equation tells you how to backpropagate through $\log|\cdot|$, and the third equation tells you how to backpropagate through a matrix inversion. $\log|\cdot|$ refers to the log of a determinant.

(a) (3 points) Draw a computational graph for $\mathcal{L}_1$.

(b) (6 points) Compute $\frac{\partial \mathcal{L}_1}{\partial \mathbf{X}}$.

(c) (3 points) Draw a computational graph for $\mathcal{L}_2$.

(d) (6 points) Compute $\frac{\partial \mathcal{L}_2}{\partial \mathbf{X}}$.

(e) (2 points) Compute $\frac{\partial \mathcal{L}}{\partial \mathbf{X}}$.

3. (40 points) **2-layer neural network.** Compete the two-layer neural network Jupyter notebook. Print out the entire workbook and relevant code and submit it as a pdf to gradescope. Download the CIFAR-10 dataset, as you did in HW #2.

4. (25 points) **General FC neural network.** Compete the FC Net Jupyter notebook. Print out the entire workbook and relevant code and submit it as a pdf to gradescope.