Laissez des traces de votre démarche pour toutes les questions !
For all questions, show your work !

1. **(10 points) Neural Networks, Classification)**
   Consider training a standard feed-forward neural network. For the purposes of this question we are interested in a single iteration of SGD on a single training example : $(\boldsymbol{x}, y)$. We denote $f(\boldsymbol{x}, \boldsymbol{\theta})$ as the output of the neural network with model parameters $\boldsymbol{\theta}$. Now let's say g is the output activation function and $a(\boldsymbol{x}, \boldsymbol{\theta})$ is the pre-activation network output such that $f(\boldsymbol{x}, \boldsymbol{\theta}) = g(a(\boldsymbol{x}, \boldsymbol{\theta}))$.
   (a) Assuming the network's goal is to do binary classification (with the detailed structure above), what would be an appropriate activation function for the output layer, i.e. what would be an appropriate function g.
   (b) What does the output represent under this activation function ?
   (c) Let $L_{CE}(f(\boldsymbol{x}, \boldsymbol{\theta}), y)$ be cross-entropy loss, express it as a function of $f(\boldsymbol{x}, \boldsymbol{\theta})$ and y.
   (d) Compute the partial derivative $\frac{\partial L_{CE}(f(\boldsymbol{x},\boldsymbol{\theta}),y)}{\partial a(\boldsymbol{x},\boldsymbol{\theta})}$.
   (e) Let $L_{MSE}(f(\boldsymbol{x}, \boldsymbol{\theta}), y)$ be the mean-squared error, express it as a function of $f(\boldsymbol{x}, \boldsymbol{\theta})$ and y.
   (f) Compute the partial derivative $\frac{\partial L_{MSE}(f(\boldsymbol{x},\boldsymbol{\theta}),y)}{\partial a(\boldsymbol{x},\boldsymbol{\theta})}$.
   (g) Based on your answers to the above questions, what would argue would be the more appropriate loss function for binary classification and why ?

2. **(15 point) Convolutional Neural Nets.**
   (a) Describe three elements that differentiate Convolutional Neural Networks (CNNs) from standard feed-forward neural networks. For each element, discuss the advantages (computation, statistical, etc.) it brings for application on images.
   (b) Most applications of CNNs to images include a data standardization step where the size of the input images are made the same. Describe two different ways one could apply CNNs to variable sized images for a classification task.
   (c) Compute the "full", "valid", and "same" convolutions (with kernel flipping) for the following 1D matrices.

$$\text{input matrix} : [1, 2, 3, 4] * \text{kernel} : [1, 0, 2]$$

3. **(20 point) Recurrent Neural Networks.**
   Consider the following variant of the LSTM :

$$\begin{aligned}
\boldsymbol{f}_t &= \text{sigmoid}(W_f \boldsymbol{x}_t + U_f \boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \\
\boldsymbol{o}_t &= \text{sigmoid}(W_o \boldsymbol{x}_t + U_o \boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \\
\boldsymbol{c}_t &= \boldsymbol{f}_t \odot (1 - \boldsymbol{f}_t) \odot \boldsymbol{c}_{t-1} + \tanh(W_c \boldsymbol{x}_t + U_c \boldsymbol{h}_{t-1} + \boldsymbol{b}_c) \\
\boldsymbol{h}_t &= \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t) \\
y_t &= W_y \boldsymbol{h}_t + d
\end{aligned}$$

   (a) Assume that each of $\boldsymbol{x}_t$, $\boldsymbol{h}_t$, $\boldsymbol{f}_t$, $\boldsymbol{o}_t$, $\boldsymbol{c}_t$, and $y_t \in \mathbb{R}$ are all 1-dimensional and that the model weight matrices W_f, U_f, W_o, U_o, W_c, U_c are just 1×1 (i.e. just scalar parameters) as are the model offset parameters $\boldsymbol{b}_f$, $\boldsymbol{b}_o$, $\boldsymbol{b}_c$, and d. Compute the derivative $\frac{dc_1}{dx_0}$.

(b) Compare this LSTM-variant to a standard sigmoid RNN :

$$\boldsymbol{h}_t = \text{sigmoid}(W\boldsymbol{x}_t + U\boldsymbol{h}_{t-1})$$
$$y_t = W_y\boldsymbol{h}_t + d.$$

How do these two models compare with respect to their ability to learn long-term dependencies between the model's input and output via gradient backpropagation ? (Explain your answer.)

4. **(10 point) Optimization.**
Briefly compare and contrast the following pairs of optimization algorithms (providing specific differences, advantages and disadvantages of each) :
   (a) Stochastic gradient descent (SGD) and Adagrad,
   (b) Adagrad and RMSprop,
   (c) RMSprop and Adam,
   (d) Newton's Method and Conjugate Gradient,
   (e) Conjugate Gradient and BFGS.

5. **(20 point) Regularization.**
   (a) Explain why the capacity of a neural network grow as the number of iterations increases ?
   (b) Explain the intuition behind the link between early-stopping and L_2 weight decay for a linear model.
   (c) Consider applying dropout to a network without hidden layers applied to a k-class classification task. Specifically, with input $\boldsymbol{x} \in \mathbb{R}^n$, target output $y \in \{0, 1, \ldots, k-1\}$, weight matrix W, and with dropout mask $\boldsymbol{v}$, let the network output with dropout be (with the $\odot$ representing an element-wise multiplication) :

$$P(\text{y} = y \mid \boldsymbol{x}, \boldsymbol{d}) = \text{softmax}\left(W^\top(\boldsymbol{d} \odot \boldsymbol{x}) + \boldsymbol{b}\right)_y$$

Let's define the ensemble of dropout k-class classifiers as

$$P_{\text{ensemble}}(\text{y} = y \mid \boldsymbol{x}) = \frac{\tilde{P}_{\text{ensemble}}(\text{y} = y \mid \boldsymbol{x})}{\sum_{y'} \tilde{P}_{\text{ensemble}}(\text{y} = y' \mid \boldsymbol{x})},$$

where

$$\tilde{P}_{\text{ensemble}}(\text{y} = y \mid \boldsymbol{x}) = \sqrt[2^n]{\Pi_{d \in \{0,1\}^n} P(\text{y} = y \mid \boldsymbol{x}, \boldsymbol{d})}$$

Prove that the non-dropout network with weights scaled by $1/2$:

$$P(\text{y} = y \mid \boldsymbol{x}) \propto \exp\left(\frac{1}{2} W_{y,:}^\top \boldsymbol{x} + \boldsymbol{b}\right)$$

corresponds exactly to an geometric mean over the ensemble of models generated by the set of dropout masks.

6. **(15 point) VAE.**
Consider a generative model that factorizes as follows $p(x, z) = p(x \mid z)p(z)$, where $p(x \mid z)$ is mapped through a neural net, i.e. $p(x \mid z) = p(x; f(z; \theta))$, where θ is the set of parameters for the generative network (i.e. decoder).
We have $z \in \mathbf{R}^K$, which implies a continuous latent space model, and $p(z) = \mathcal{N}(0, I_K)$. The framework of the variational autoencoder considers maximizing the variational lower bound on the log-likelihood $\mathcal{L}(\theta, \phi) \leq \log p(x)$, which is expressed as

$$\mathcal{L}(\theta, \phi) = \mathbf{E}_{q_\phi}[\log p(x|z)] - \mathbf{KL}(q_\phi(z|x)||p_\theta(z)), \tag{1}$$

where ϕ is the set of parameters used for the inference network (i.e. encoder). The reparameterization trick used in the original work rewrites the random variable in the variational distribution as

$$z = \mu(x) + \sigma(x) \odot \epsilon \tag{2}$$

where $\epsilon \sim \mathcal{N}(\epsilon; 0, I)$.

(a) What is the purpose of the reparameterization trick ?

(b) As discussed in class, the traditional mean-field variational method corresponds to factorizing the variational posterior distribution as a product of distributions : $q^{mf}(z_i) = \prod_j \mathcal{N}(z_{i,j}|m_{i,j}, \sigma^2_{i,j})$ and maximizing the lower bound directly with respect to the variational parameters $m_{i,j}$ and $\sigma^2_{i,j}$. Can the inference network used in the standard VAE outperform the mean-field method (in the sense of maximizing the upper bound in Eq. 1) ? (Explain your answer.)

(c) What is the advantage of using an encoder as in the VAE ?

(d) Could the Inverse Auto-regressive Flow (IAF) approach to inference in the VAE outperform the mean-field method (discussed above) ? (Explain your answer.)

7. **(10 point) Attention**
In class we saw two basic strategies to machine translation : one based on the Seq2Seq model, and one based on the soft-attention model of Bahnadau et al. (in Dima's lecture). Describe the two approaches and detail how they differ (include a mathematical description of the soft-attention mechanism). What advantage does the soft-attention approach have over the Seq2Seq approach ?