Deep Learning Worksheet 3:
Neural Networks and Minimization

Note: All questions have equal weight.

Note: Multiple answers are sometimes possible, only the correct combination of answers counts.

Submission deadline: 23:59 MEST, Friday, May 19th, 2018.

Characterization of the learning problem

Q1: Neural network structure

Consider a densely connected two-layer neural network with one input and one output node and 10 hidden layers. The network parameters $\boldsymbol{\theta}$ are defined by the weights of the input to the hidden and the hidden to the output layer, as well as the biases of the hidden and output neurons. The hidden and output neurons have sigmoid activation functions $f(x) = (1 + \mathrm{e}^{-x})^{-1}$.

Question: (yes/no)

For this neural network, there is a setting of parameters $\boldsymbol{\theta}$, such that the network approximates any given smooth function $y(x)$ in the sense $|\hat{y}(x;\theta) - y(x)| < \epsilon$ for all $x \in \mathbb{R}$ for a given $\epsilon > 0$?

Q2: In a two-layer NN (input, one hidden, one output), with sigmoid activation functions, how many hidden neurons are at least needed to approximate the function

$$y(x) = h(x + d) - h(x - d)$$

with the Heavyside step function

$$h(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases}$$

in the sense $|\hat{y}(x;\theta) - y(x)| < 1e^{-5}$?

A	B	C	D
1	2	4	∞

Q3: Autoencoder. We are given a densely connected network, called "Autoencoder" with 100 input and 100 output neurons. In between we have hidden layers with $(30, 10, 2, 10, 30)$ and neurons. The network parameters $\boldsymbol{\theta}$ are defined by the weights of the input to the hidden and the hidden to the output layer, as well as the biases of the hidden and output neurons. Compute the number of free (learnable) parameters. If we store these parameters in single precision floating point (float32), how much memory do we need?

A	B	C	D	
$1,000$ to $1,500$	$26,000$ to $27,000$	$27,000$ to $28,000$	$1,000,000$	Byte

---

Q4. Which of these packages does not offer neural network optimization with Graphical Processor Unit (GPU) acceleration?

A	B	C	D
PyTorch	scikit-learn	Tensorflow	Keras

Q5. We want to minimize the functions:

$$C_1(\boldsymbol{\theta}) = \frac{1}{2}\theta_1^2 + \frac{1}{2}\theta_2^2$$
$$C_2(\boldsymbol{\theta}) = \frac{1}{2}\theta_1^2 + \frac{1}{2000}\,\theta_2^2$$

using either simple gradient descent, or using the Newton method starting from an initial point $\boldsymbol{\theta}_0 = (1,1)^\top$. For gradient descent, assume we use the same learning rate for both functions, and we choose this learning rate such that the algorithm converges asymptotially. Suppose that if we minimize C_1 with gradient descent we need $1,000$ iterations to reach the minimum $\boldsymbol{\theta} = (0,0)$ within a small error tolerance ϵ. How many steps (order of magnitude) do you expect gradient descent and Newton to take for minimizing C_2 to within the same error tolerance ϵ.

	A	B	C	D	E	F
Newton	1	1	1	10^3	10^3	10^3
Gradient descent	1	10^3	10^6	10^3	10^6	1

Q6. Consider the minimization of the quadratic function

$$C(\boldsymbol{\theta}) = \boldsymbol{\theta}^\top \mathbf{A}\boldsymbol{\theta}$$

where $\boldsymbol{\theta} \in \mathbb{R}^N$ is the parameter vector and $\mathbf{A} \in \mathbb{R}^{N\times N}$ is a dense matrix. We consider minimizing this function using Newton or gradient descent from initial point $\boldsymbol{\theta}_0 = (1,1)^\top$. How much slower is a single Newton step than a single gradient descent step (order of magnitude)?

A	B	C	D	E
N^{-1}	1	N	N^2	N^3

Q7. We want to train a deep neural network by minimizing a given loss function. We assuming that the global minimum of the loss function is significantly lower than local minima. Which minimizer do you recomment for reliably finding this global minimum from an arbitrary starting point. Argue why.

A	B	C
Gradient descent	Newton method	Stochastic gradient descent

Q8. We have built a neural network with the loss function $C(\boldsymbol{\theta})$. We want to use stochastic gradient descent with batchsize B and learning rate η to train the network. Which of these strategies is the best to learn a model that makes reliable predictions?

A	Set $B = N$ (data size) and $\eta = 10^2$. Find parameters $\boldsymbol{\theta}$ by minimizing $C(\boldsymbol{\theta})$
B	Divide the data into training and validation set $(\mathbf{X}^{\text{train}}, \mathbf{X}^{\text{val}})$. For different combinations of (B, η), optimize $$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} C(\boldsymbol{\theta}, \mathbf{X}^{\text{train}})$$ and call the resulting loss $C^{\text{train}}_{B,\eta} = C(\boldsymbol{\theta}^*, \mathbf{X}^{\text{train}})$. Use the solution with $$\boldsymbol{\theta}^\dagger = \arg\min_{B,\eta} C^{\text{train}}_{B,\eta}.$$
C	Same as B, but use the solution $$\boldsymbol{\theta}^* = \arg\min_{B,\eta} C^{\text{val}}_{B,\eta}$$ where $C^{\text{val}}_{B,\eta} = C(\boldsymbol{\theta}^*, \mathbf{X}^{\text{val}})$.
D	Choose all parameters $(\boldsymbol{\theta}, B, \eta)$ by minimizing over the joint space defined by $(\boldsymbol{\theta}, B, \eta)$ using all data.

Q9. We consider fitting a function $y(\mathbf{x}) : \mathbb{R}^{10} \to \mathbb{R}$ by training a neural network. We consider a 10-layer dense network, but want to want to find the best number of neurons in each layer. We want to use one nonlinear activation function, but are not sure which one. We will use the Adam optimizer to train the NN.

How many dimensions does the Hyperparameter space have?

A	B	C	D	E
8	10	14	16	105