UNIVERSITY OF TORONTO
FACULTY OF ARTS AND SCIENCE

FINAL EXAMINATION, APRIL 2017

DURATION: 3 hours

CSC 411 H1S — Machine Learning and Data Mining

Aids allowed: Non-programmable calculators and
Aid sheets distributed with the exam
Examiner(s): M. Guerzhoy

Student Number: |_|_|_|_|_|_|_|_|_|_|

Family Name(s): ________________________________

Given Name(s): ________________________________

Do **not** *turn this page until you have received the signal to start.*
In the meantime, please read the instructions below *carefully*.

This final examination paper consists of 7 questions on 28 pages (including this one), printed on both sides of the paper. *When you receive the signal to start, please make sure that your copy is complete, fill in the identification section above, and write your student number where indicated at the bottom of every odd-numbered page (except page 1).*

Answer each question directly on this paper, in the space provided, and use the reverse side of the previous page for rough work. If you need more space for one of your solutions, use the reverse side of a page or the pages at the end of the exam and *indicate clearly the part of your work that should be marked.*

Write up your solutions carefully! In particular, use notation and terminology correctly and explain what you are trying to do—part marks *will* be given for showing that you know some aspects of the answer, even if your solution is incomplete.

A mark of at least 40% (after adjustment, if there is an adjustment) on this exam is required to obtain a passing grade in the course.

Marking Guide

# 1:	______/ 10
# 2:	______/ 15
# 3:	______/ 20
# 4:	______/ 10
# 5:	______/ 10
# 6:	______/ 15
# 7:	______/ 20
TOTAL:	______/100

Good Luck!

Use this page for rough work—clearly indicate any section(s) to be marked.

Question 1. [10 MARKS]

Draw a design of a small neural network that takes in two inputs, x_1 and x_2, and outputs a number close to 1 if $x_1 < 0$ and $x_2 < 0$ and a number close to 0 if $x_1 > 0$ and $x_2 > 0$. You may only use sigmoid activation functions. Include the weights you used. Briefly explain why your network computes what it's required to compute.

Use this page for rough work—clearly indicate any section(s) to be marked.

Question 2. [15 MARKS]

In this question, we are considering generating a dataset, which will then be randomly split into a test set and a training set. The dataset will consist of N 2-dimensional vectors, with each vector having the label 0 or 1.

Part (a) [5 MARKS]

Describe a dataset for which 3-Nearest-Neighbours will perform substantially better than Linear Regression on the test set. Explain your reasoning.

Part (b) [5 MARKS]

Describe a dataset for which Linear Regression will perform better than a one-hidden-layer neural network on the test set. Explain your reasoning.

Use this page for rough work—clearly indicate any section(s) to be marked.

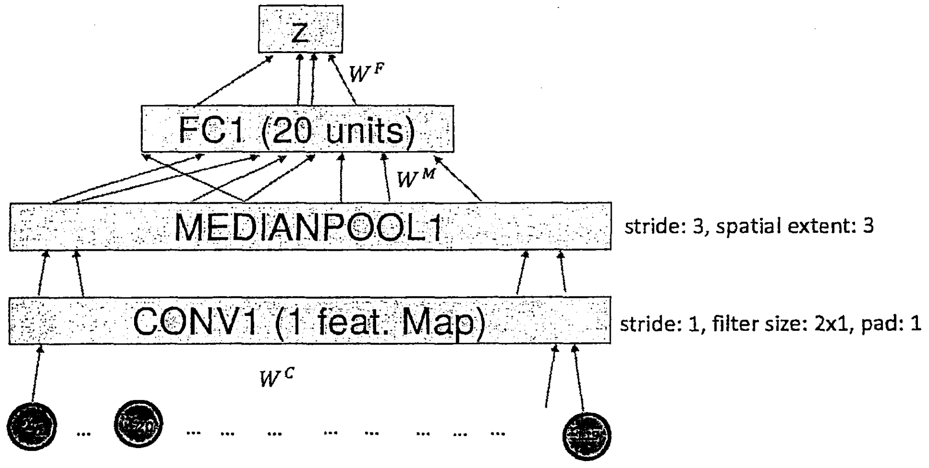
Part (c) [5 MARKS]

Describe how to generate a dataset on which a 5-hidden-layer neural network could be expected to perform bettern than a single-hidden-layer neural network (if trained appropriately.) Explain your reasoning. Use pseudocode to accompany your description of how to generate the dataset.

Use this page for rough work—clearly indicate any section(s) to be marked.

Question 3. [20 MARKS]

Consider the Convolutional Neural Network below.


Z
W^F
FC1 (20 units)
W^M
MEDIANPOOL1
stride: 3, spatial extent: 3
CONV1 (1 feat. Map)
stride: 1, filter size: 2x1, pad: 1
W^C

The network takes in an input of dimension 119 × 1, and its output is of dimension 1 × 1. The network consists of the input layer X (with a 0-pad of witdth 1), a convolutional layer CONV1 which consists of one feature map with a 2×1 filter which uses the *ReLU* nonlinearity, a median-pooling layer MEDIANPOOL1, a fully-connected layer FC1 which uses a *ReLU* nonlinearity, and an output layer Z of size 1 × 1, which is fully connected to the FC1 layer and uses a *sigmoid* nonlinearity. Recall that $\sigma'(t) = \sigma(t)(1 - \sigma(t))$.

Denote the weight that connects the i-th unit in FC1 to Z by W_i^F and the bias for Z by b^F. Denote the weight that connects the j-th unit in MEDIANPOOL1 to the i-th unit in FC1 by W_{ji}^M and the bias of the i-th unit in FC1 by b_i^M. Let $W^C = [W_1^C, W_2^C]$ and the bias for the CONV1 layer be b^C.

A unit in a median-pooling layer outputs the median value of the neurons in its receptive field (i.e., the neurons connected to the unit).

Part (a) [4 MARKS]

How many parameters (i.e., values that specify how the network computes its output) are there in this network? Briefly show your work.

Use this page for rough work—clearly indicate any section(s) to be marked.

Let the training set inputs be $X = [X^{(1)}, X^{(2)}, ..., X^{(N)}]$ and the expected outputs be $Y = [Y^{(1)}, Y^{(2)}, ..., Y^{(N)}]$.

Let the outputs of the layers in the network be denoted using $c(X^{(i)})$, $m(X^{(i)})$, $f(X^{(i)})$, and $z(X^{(i)})$ for the CONV1, MEDIANPOOL1, FC1, and Z layers, respectively (you may use notation such as z_i, f_j, etc.). You may use those without explicitly telling us how to compute them.

The cost function is

$$cost(X, Y) = \sum_n cost(X^{(n)}, Y^{(n)}) = \sum_n (-Y^{(n)} \log(z(X^{(n)})) - (1 - Y^{(n)}) \log(1 - z(X^{(n)})).$$

Part (b) [8 MARKS]

Compute $\partial Cost/\partial W^M_{ji}$, for the *entire training set.* Show the details of the computation. Use Backpropagation to obtain the final answer: **show how you would compute the gradients layer-by-layer.** You may not use matrix multiplication in your answer.

Use this page for rough work—clearly indicate any section(s) to be marked.

Part (c) [8 MARKS]

Compute $\partial Cost/\partial W_1^C$, for the *entire training set*. Show the details of the computation. Note: the padding is significant. Use Backpropagation to obtain the final answer: **show how you would compute the gradients layer-by-layer**. You may not use matrix multiplication in your answer.

Use this page for rough work—clearly indicate any section(s) to be marked.

CONT'D...

Question 4. [10 MARKS]

Describe how to learn word2vec vectors using negative sampling. Be specific. Use pseudocode. You do not need to compute any gradients, but you do need to specify which gradients need to be computed.

Use this page for rough work—clearly indicate any section(s) to be marked.

CONT'D...

Question 5. [10 MARKS]

A Mixture of Gaussians model is defined using

```
mus = np.array([[0, 5], [1, 1]])
sigmas = np.array([  [[1, 0],
                      [0, 2]],

                    [[2, 0],
                     [0, 3]]])

pis = np.array([0.2, 0.8])
```

Write code to generate ten datapoints using the model. To generate random numbers, you may *only* use the following function, which returns *one* float.

```
def rnorm(loc, scale):
   """Return an a sample from the normal
   distribution N(mu=loc, sigma=scale)
   loc and scale are both floats"""
```

Use this page for rough work—clearly indicate any section(s) to be marked.

Question 6. [15 MARKS]

Write code that uses the Metropolis algorithm to fit a linear regression model to the data (x_{raw}, y), and to then output the predictions using this model for the data x_new. Your code's output should be the predictions made for x_{new}. You can use the supplied functions. Assume that the data is generated using $y \sim N(ax_{raw} + b, \sigma^2)$ for $\sigma^2 = 4$, and that the prior for the unknown parameters a and b is $N(0, 1)$. You should use a Guassian distribution as the proposal distribution. The probability density function of a univariate Gaussian distribution is $\frac{1}{\sqrt{2\pi\sigma^2}} \exp(\frac{-(x-\mu)^2}{2\sigma^2})$. **Annotate the code to show what you are doing.**

```
x_raw.shape == (20,)
x  = vstack((    ones_like(x_raw),
                 x_raw,
           ))
y.shape == (20,)

x_new.shape == (30,)

def loglik(x, mu, sigma):
    return sum(-.5*log(2*pi*sigma**2)-(x-mu)**2/(2*sigma**2))

def rnorm(loc, scale, size):
   """Return an array of size independent samples from the normal
   distribution N(mu=loc, sigma=scale)"""
```

Use this page for rough work—clearly indicate any section(s) to be marked.

CONT'D...

Question 7. [20 MARKS]

We would like to use REINFORCE to train an agent that plays Rock Paper Scissors against the computer. The game is played as follows: both the agent and the computer pick an action from the set `{0, 1, 2}`. The reward is $+1$ if the tuple of `(agent, computer)` actions is one of `(0, 1)`, `(1, 2)`, or `(2, 0)`. The reward is -1 if the tuple of `(agent, computer)` actions is one of `(1, 0)`, `(2, 1)`, or `(0, 2)`. The reward is 0 otherwise. (For simplicity, we substitute the integers 0, 1, 2 for Rock, Paper, and Scissors from the familiar game.)

The computer is using an unknown strategy. For a computer action c_{t-1}, taken at time $t-1$, the policy function that defines the probability of agent action a_t is

$$\pi(a_t = a|c_{t-1}) = p_{a,c_{t-1}}.$$

That is, the policy function is parametrized using 9 coefficients. You may use the function `rps(act)` as follows:

```
computer_act, reward = rps(act)
```

The function takes in the agent's action, and returns the computer's action and the reward the agent gets (so that you do no need to compute the reward yourself). For reference, the Policy Gradient Theorem is:

$$\nabla\eta(\theta) = \sum_s d_\pi(s) \sum_a q_\pi(s,a)\nabla_\theta(a|s,\theta).$$

The REINFORCE algorithm is as follows:

```
Repeat
    Generate an episode $S_0, A_0, R_1, ..., S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot,\theta)$
        For each step of the episode $t = 0, ..., T-1$:
        $G_t \leftarrow$ return from step $t$
        $\theta \leftarrow \theta + \alpha\gamma^t G_t \nabla_\theta \log \pi(A_t|S_t,\theta)$
```

Write pseudocode to use REINFORCE to learn the parameters of the policy function. Make clear how you obtained that pseudocode. You *must* provide all the details of the computation of each variable, and you *must* provide all the necessary derivations in your answer. You do not need to justify the REINFORCE algorithm itself. Please start your answer on the next page. **Neatness and logical structure count!.**

Use this page for rough work—clearly indicate any section(s) to be marked.

Use this page for rough work—clearly indicate any section(s) to be marked.

CONT'D...

This page was intentionally left blank

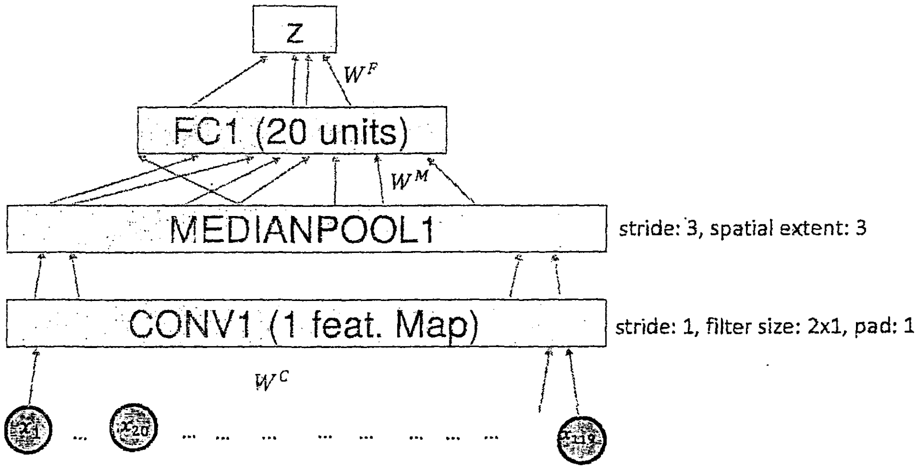
Use this page for rough work—clearly indicate any section(s) to be marked.

CONT'D...

This page was intentionally left blank

PLEASE WRITE NOTHING ON THIS PAGE

CSC411 Exam Aid Sheet, APRIL 2017
For reference only



The network takes in an input of dimension 119×1, and its output is of dimension 1×1. The network consists of the input layer X (with a 0-pad of witdth 1), a convolutional layer CONV1 which consists of one feature map with a 2×1 filter which uses the *ReLU* nonlinearity, a median-pooling layer MEDIANPOOL1, a fully-connected layer FC1 which uses a *ReLU* nonlinearity, and an output layer Z of size 1×1, which is fully connected to the FC1 layer and uses a *sigmoid* nonlinearity. Recall that $\sigma'(t) = \sigma(t)(1 - \sigma(t))$.

Denote the weight that connects the i-th unit in FC1 to Z by W_i^F and the bias for Z by b^F. Denote the weight that connects the j-th unit in MEDIANPOOL1 to the i-th unit in FC1 by W_{ji}^M and the bias of the i-th unit in FC1 by b_i^M. Let $W^C = [W_1^C, W_2^C]$ and the bias for the CONV1 layer be b^C.

A unit in a median-pooling layer outputs the median value of the neurons in its receptive field (i.e., the neurons connected to the unit).

$$\nabla\eta(\theta) = \sum_s d_\pi(s) \sum_a q_\pi(s,a)\nabla_\theta(a|s,\theta).$$

The REINFORCE algorithm is as follows:

```
Repeat
    Generate an episode $S_0, A_0, R_1, ...; S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot,\theta)$
        For each step of the episode $t = 0, ..., T-1$:
        $G_t \leftarrow$ return from step $t$
        $\theta \leftarrow \theta + \alpha\gamma^t G_t \nabla_\theta \log\pi(A_t|S_t,\theta)$
```