Asssignment2a, cmsc5707 , 2024-5, 2024.11.08 (Q1-Q13) suggested answers

Description

After you complete the assignment, you may get a mark (probably 0), it is a dummy result and not the real result because the true answers have not been input to the systems yet. The real result will be published 2 weeks or more after the assignment deadline.

Instructions Hard deadline: late submission is not allowed.

Please click on the "save and submit" button at the end of the page after you completed the assignment.
If your answer is not an integer, give it to the nearest 3 decimal places.
If your answer is an integer, add decimal point has no harm. E.g. 3=3.0=3.00=3.000 etc.

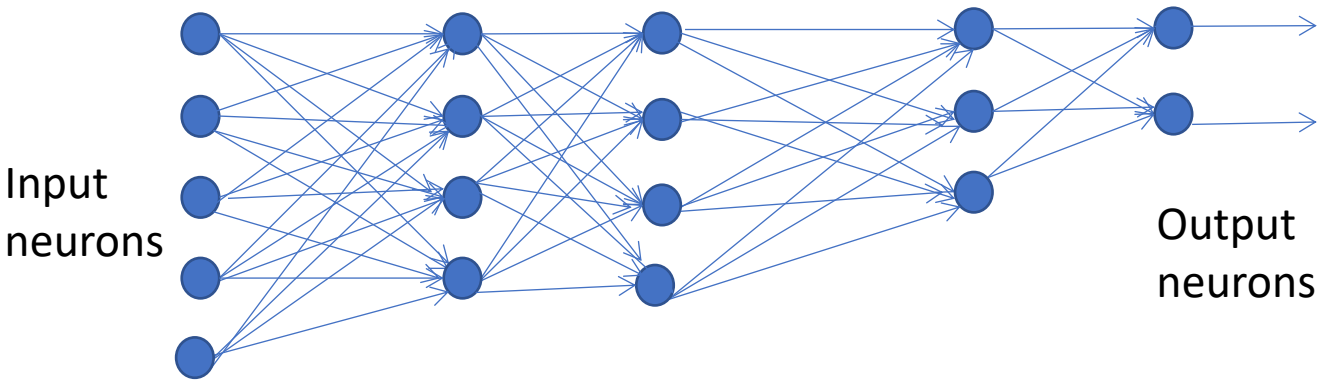
Please complete the assignment before the deadline. Multiple attempts are allowed.

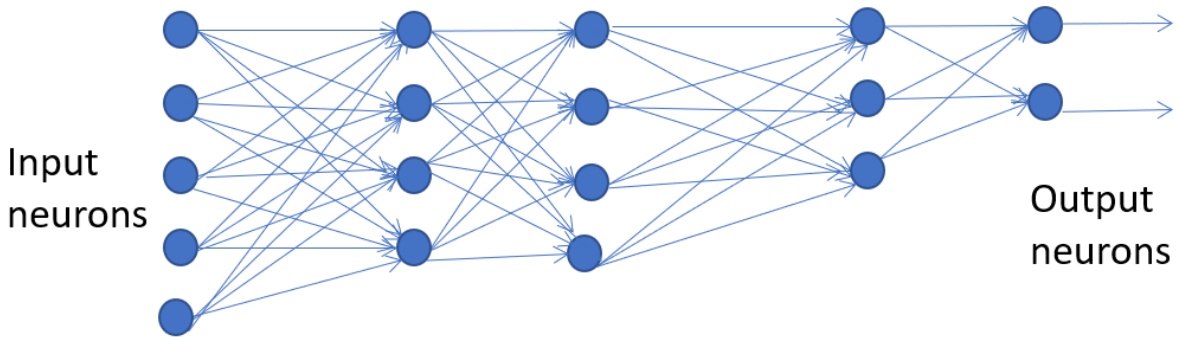
%%% Question3 has some problems during marking, please see the details below.
//

A57074.2.1 : Neural Network model 1

A fully connected neural network is shown below.

How many weights are in this network?





Answer1: 54 (ok4)

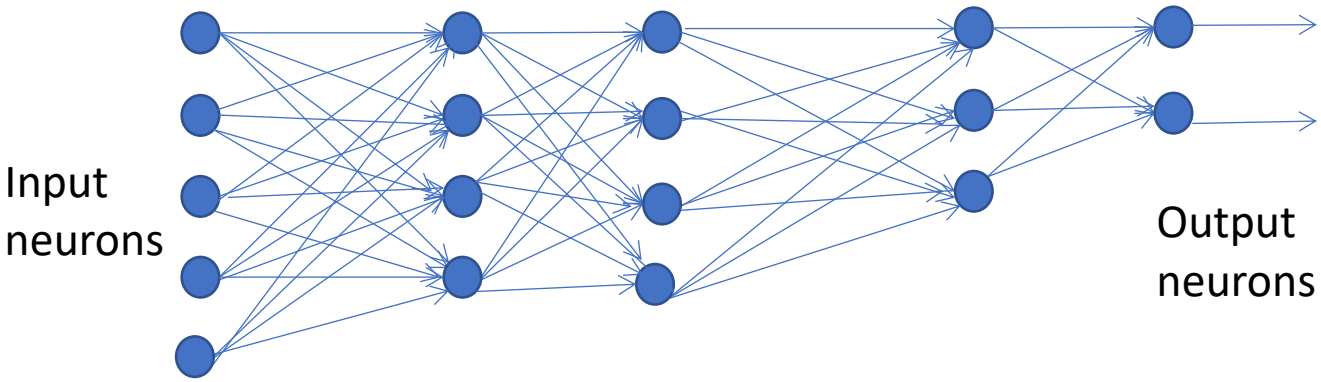
Because all the neurons are fully connected, so number of weights=5*4+4*4+4*3+3*2=54

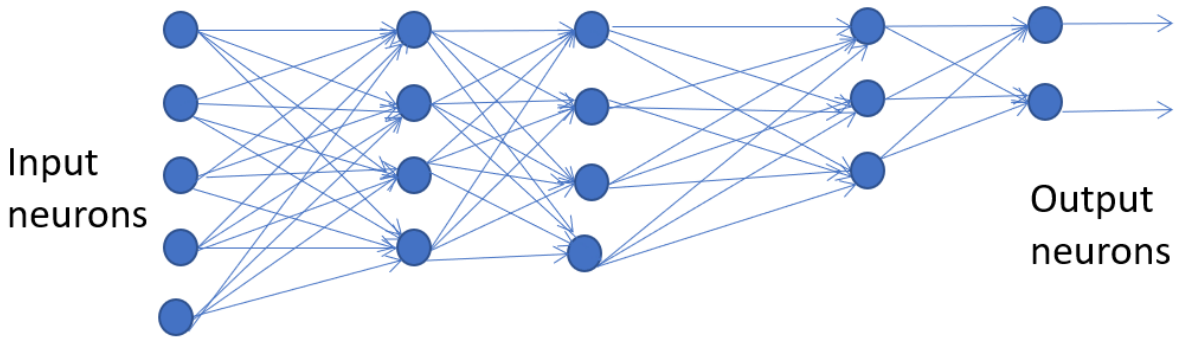
//

A57074.2.2: Neural Network model 2

A fully connected neural network is shown below.

How many biases are in the network?





Answers2: 13 (ok4)

Note: Input neurons have no bias

//

A57074.2.3 : Question ID: Boosting 1

Boosting 1: An Ada-Boost algorithm based on parallel axis weak classifiers is used to build a strong classifier. There are [x] samples, in the training data set, each sample is 2-dimension, 40% are positive (+1) examples and the rest are negative (-1) examples (round off if necessary).

At step t=1, a weak classifier hs( ) which gives the least error is found, using hs( ) 200 samples are incorrectly classified. Calculate the weight (α at time t) of this weak classifier in building the final strong classifier.

Answer3 : (ok4)
X range 2000 – 3000

Since I cannot change the decimal setting for the answer after it is deoplyed (a system problem, or maybe I set it wrongly at the beginning) , so I add the answer range to be +/-2, so all get 10 marks . However, please double check your answer whether you are really answer it correctly.

answer

$$0.5 * \ln(\quad (1 - (200/x)) \quad /(200/x) \ )$$

+/- 0.002

//

A57074.2.4 : Question ID: Boosting 2

An Ada-Boost algorithm based on parallel axis weak classifiers is used to build a strong classifier. There are 2000 samples in the training data set.

At step t=1, a weak classifier hs( ) which gives the least error is selected, using hs( ) 300 samples are incorrectly classified.

Assume D=normalized incorrectly classified weight.

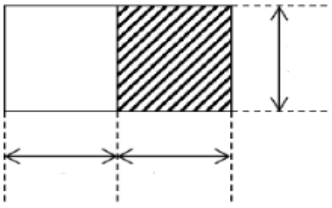
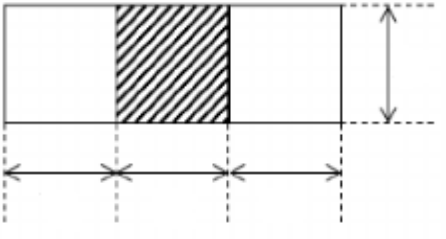
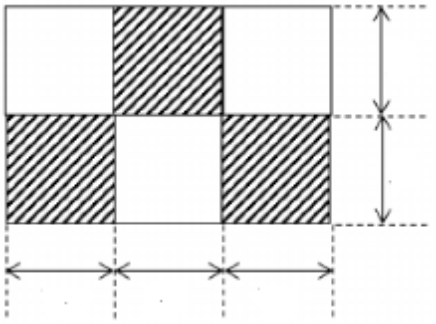
At time t=2, all normalized weights D (time t=2) for incorrectly classified samples should be equal. Find the value of D (time t=2).

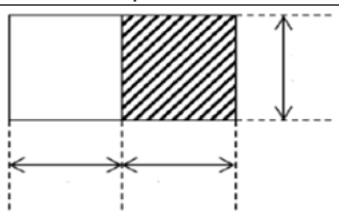
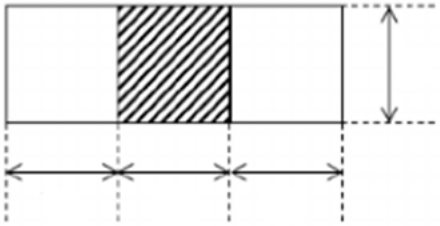
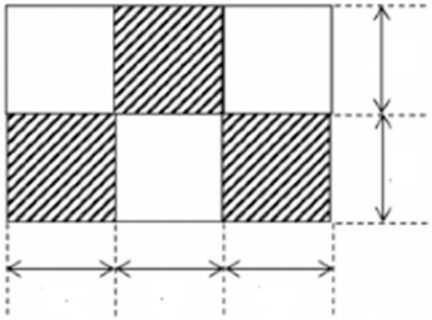
```
Answer4= 0.0167 (ok4)

+/- 0.005

%matlab
clear all
clc
N_total=2000
incorrect_n=30
correct_n=N_total- incorrect_n
Dt1=1/(N_total) %weigth for D(t=1) , all equal
error_rate_t=incorrect_n/N_total

alpha_t=0.5*log((1-error_rate_t)/error_rate_t)
correct_weight=correct_n * Dt1*exp(-alpha_t) %decrease if correct
incorrect_weight=incorrect_n * Dt1*exp(alpha_t)%increase if correct
Z= correct_weight+incorrect_weight % is the normalization factor
Dt2_incorrect_prob =Dt1*exp(alpha_t)/Z
%answer: Dt2_incorrect_prob = 0.0167
```

//

A57074.2.5: object detection

In an object detection system using a detection window with size width=48 pixels, and height = 24 pixels. There are 3 basic feature types as shown below. The feature value is equal to (Sum of pixels in shaded area)- (Sum of pixels in white area). The numbers of different features found in the detection window for type I (one row, each row has two rectangles) , II (one row, each row has three rectangles), III (two rows, each row has three rectangles) are N1,N2,N3 respectively. Find N1+N2+N3.

Type	Feature shape
I	
II	
III	

Type	Feature shape
I	
II	
III	

Answer5: 339744 (ok4)
+/- 0

```
%Solution:
clear %
%clc %don't use 'clc', it clear the screen >> no diaply in command win
win_width=48, win_height=24
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
temp=0; %Type-I feature: block aspect ratio is width=2 units, height=1unit
for nx=1:win_width/2%nx=no. of x pixels in white area. Min =1,max=win_width/2
    for ny=1:win_height%ny=no. of y pixels in white area. Min =1,max=win_height
       number_of_blocks_x=(win_width-2*nx+1);%no.of x Blocks fit in win_width
       number_of_blocks_y=(win_height-ny+1);%no.of y Blocks fit in win_height
       temp=number_of_blocks_x*number_of_blocks_y+temp;
    end
end
N1=temp
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
temp=0;
%Type-II: aspect ratio of the feature block, width=3 units, height=1unit
for nx=1:win_width/3 %nx=no. of x pixels in white area.Min =1,max=win_width/3
    for ny=1:win_height%ny=no. of y pixels in white area.Min=1,max=win_height
       number_of_blocks_x=(win_width-3*nx+1);%no.of x Blocks fit in win_width
       number_of_blocks_y=(win_height-ny+1);%no.of y Blocks fit in win_height
        temp=number_of_blocks_x*number_of_blocks_y+temp;
    end
end
N2=temp %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
temp=0; %-------------------------------------------------------------------------
%type-III: aspect ratio of the feature block, width=3 units, height=2 units
for nx=1:win_width/3%nx=no. of x pixels in white area.Min =1,max=win_width/3
    %ny=no.of y pixels in white area.Min =1,max=win_height/2
```

```
    for ny=1:win_height/2
       number_of_blocks_x=(win_width-3*nx+1);%no.of x Blocks fit in win_width
       number_of_blocks_y=(win_height-2*ny+1);%no.of y .. fit in win_height
        temp=number_of_blocks_x*number_of_blocks_y+temp;
    end
end
N3=temp

disp('N1+N2+N3=')
N1+N2+N3
%       339744
```

////////////////////////////

A57074.2.6: ANN1

For a neural network shown in the diagram.

Weights from input to the first hidden layer are called W.

Weights from the hidden layer to the output layer are called OW.

Biases are called B.

The activation function for neurons is Sigmoid. The parameters are:

X1=0.3, X2=0.2, X3=0.1, X4=0.4, X5=0.5

W1=0.21, W2=0.22, W3=0.31, W4=0.42, W5=0.32,

W6=0.32, W7=0.41, W8=0.42, W9=0.42, W10=0.52

OW1=0.3, OW2=0.2, OW3=0.3, OW4=0.14

B1=0.21, B2=0.3, B3=0.25, B4=0.5

Find Y1+Y2.

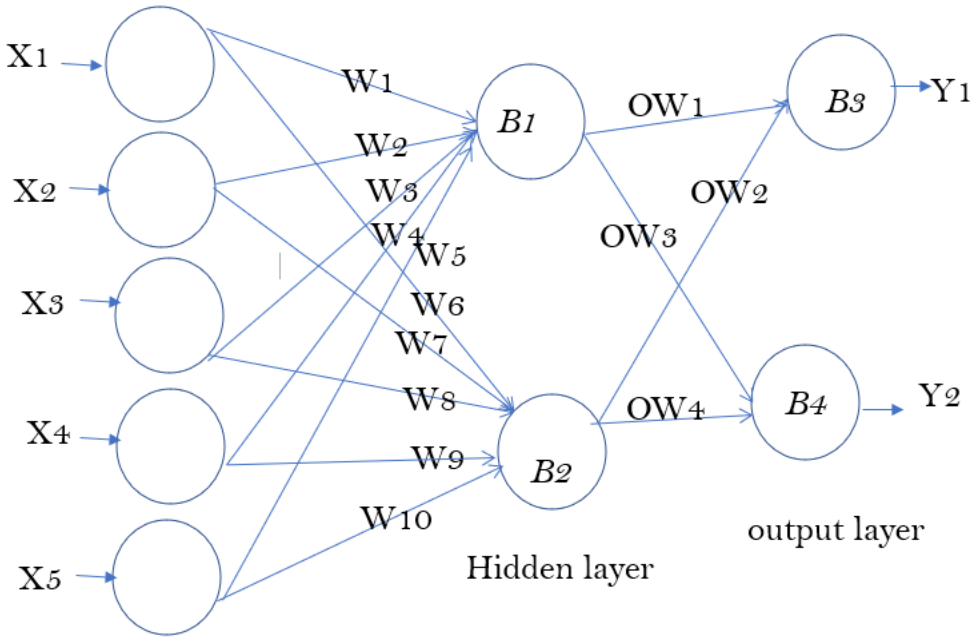


Figure ANN2a

Answer6: 1.3340 (ok4)
+/- 0.05

```
clear
clc
X1=0.3, X2=0.2, X3=0.1, X4=0.4, X5=0.5

W1=0.21, W2=0.22, W3=0.31, W4=0.42, W5=0.32,
W6=0.32, W7=0.41, W8=0.42, W9=0.42, W10=0.52

OW1=0.3, OW2=0.2, OW3=0.3, OW4=0.14

B1=0.21, B2=0.3, B3=0.25,B4=0.5
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
u1=X1*W1+ X2*W2+ X3*W3+ X4*W4+ X5*W5 +B1
out1=1/(1+exp(-(u1)))

u2=X1*W6+ X2*W7+ X3*W8+ X4*W9+ X5*W10 +B2
out2=1/(1+exp(-(u2)))
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
u1=X1*W1+ X2*W2+ X3*W3+ X4*W4+ X5*W5 +B1
out1=1/(1+exp(-(u1)))

u2=X1*W6+ X2*W7+ X3*W8+ X4*W9+ X5*W10 +B2
out2=1/(1+exp(-(u2)))

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
u3=out1*OW1+ out2*OW2 +B3
Y1=1/(1+exp(-(u3)))
%% Y1=0.6445
```

```
'out1*OW3+ out2*OW4 = 0.3017'
out1*OW3+ out2*OW4

u4=out1*OW3+ out2*OW4 +B4

Y2=1/(1+exp(-(u4)))

Y1+Y2

% result is 1.3340
```

//
A57074.2.7 ANN2: Neural network model

In the following diagram, it shows the parameters of a part of a neural network at time k. The activation function of the neurons is sigmoid. The energy to be minimized during training is E = (1/2)*(y-t)^2. such that new_w = old_w + dw, where

dw = -learning_rate*(∂E/∂w), and dw=delta_weight

Assume all the weights will be updated together only after all delta weights (dw) have been calculated for each epoch time k. So, use the current (time k) parameters for this calculation.

The current parameters at time k are:

learning_rate=0.6,

x1=0.5, x2=0.8, x3=0.7,

wh1=0.24, wh2=0.43, wh3=0.52,

ow11=0.12, ow12=0.42, ow13=0.1,

bh1=0.33, bh2=0.21, bh3 = 0.15,

s1=0.3, s2=0.2, s3=0.1 %are sensitivities

Find new wh1 at time k+1.

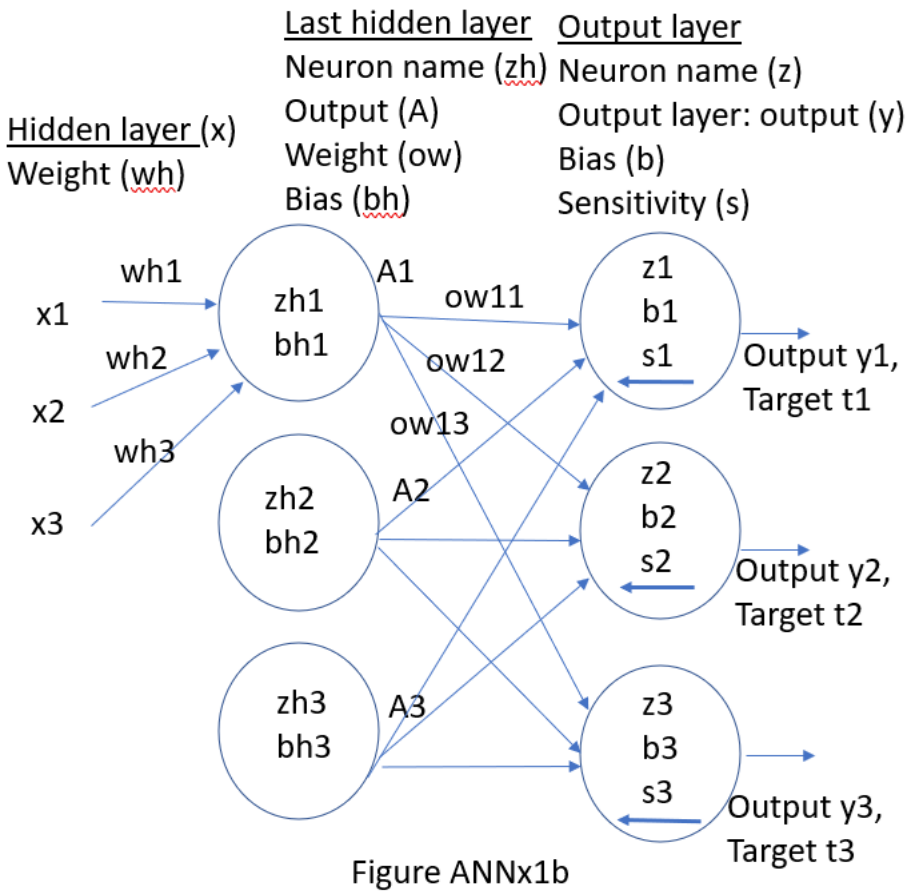


Figure ANNx1b

Answer7: 0.2329 (ok4)

+/ 0.002 is acceptable

```
%%matlab,
clear
clc
learning_rate=0.6
x1=0.5, x2=0.8, x3=0.7
wh1=0.24, wh2=0.43, wh3=0.52
ow11=0.12, ow12=0.42, ow13=0.1 ;%p(i),
bh1=0.33, bh2=0.21, bh3 = 0.15
s1=0.3, s2=0.2, s3=0.1 %are sensitivities
%Find A1 first
uh1=x1*wh1+x2*wh2+x3*wh3;
A1=1/(1+exp(-(uh1+bh1)));
term2=A1*(1-A1);
term3=x1;
s1_ow11_add_s2_ow12=s1*ow11+ s2*ow12; %0.604
result=wh1 -learning_rate*(s1*ow11+s2*ow12+s3*ow13)*term2*term3
%0.2329
```

//////////////////////

A57071.2.8 : convolution

A=[11, 22

23, 18];

B=[4,6

9,8];

C= conv2(A,B) %conv2 is 2D convolution

For the convolution function, all overlapped and non-overlapped cases are included.
Find the sum of all elements of C.

```
%answer8: 1998 (ok4)
A=[11, 22
   23, 18];
B=[4,6
   9,8];
C= conv2(A,B) %conv2 is 2D convolution
sum(sum(C))
% ans =       1998
```

////////////////////////////////

A57071.2.9: CNN1 Convolution neural network

The resolution of an input image is 97x97. A Convolution Neural Network CNN uses a kernel = 5x5, step size = 3, zero padding = p to generate a feature map of size DxD. Note: D must be an integer.
Find the value of D+p.

```
Answer9: 35 (ok4)
+/- 0
```

```
%Find the value of D.
clear
N=97 %window size NxN
m=5   % kernel size mxm
```

```
s=3    %step size =s


p=0   %zero padding :p=0
D=(((N-m+2*p)/s)+1)
D+p


p=1   %zero padding :p=1
D=(((N-m+2*p)/s)+1)
D+p


p=2   %zero padding :p=2
D=(((N-m+2*p)/s)+1)
D+p
%Answer: D=33,p=2,D+p=35
```

//

A57071.2.10: CNN2 Convolution neural network

The input layer to the first convolution layer feature maps (C1) of a convolution neural network CNN is shown in the attached figure.

Input = 84 x 84 pixels

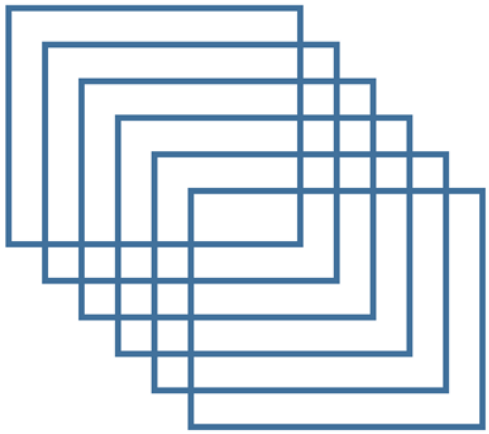
kernel size=5x5

Step size=1

For this part of the neural network, the number of weights is W and the number of biases required is B.

Find W+B.

Input is NxN



6 feature maps in C1

Answer10: 156 (ok4)

+/- 0

Weights for each feature map=K*K=5*5

There are 6 feature maps, so there are W=6*5*5

Each feature map has one bias, so there are B=6 biases.

W+B=6*5*5 + 6= 156

This calculation is independent of the input image size.

//

A57074.2.11: Evaluation of machine learning algorithms 1

In a system, the performance is as follows.

True positives (TP) =100

True Negatives (TN)=60

False Positives (FP) =51

False Negatives (FN)=32

If the accuracy is A, Precision is P, Recall is R. Find A+P+R.

%%

Answer11: 2.0783 (ok4)
+/- 0.005

```
clear
clc
TP =100
TN=60
FP =51
FN=32
%Answer:
A  = (TP+TN)/(TP+TN+FP+FN)
P  = TP/(TP+FP)
R= TP/(TP+FN)
A+P+R
%2.0783
```

//////////////////////

A57074.2.12: Evaluation of machine learning algorithms

There are 200 apples in the picture. The system can extract 150 apples, of which 20 are incorrect. Calculate the precision of the system.

answer12 = 0.866 (0k4)

+/- 0.005

Answer:

So the formula is TP/(TP+FP)=(150-20)/(150)=0.8667

The answer is independent of the number of apples (200) in the picture

////////////////////////////////

A57074.2.13: Batching

To train a neural network you are given 60000 training samples and 10000 testing samples.

Using training scheme 1 of applying the full batch method, the total iterations for 10 epochs is X1.

Using another training scheme 2 of applying the mini-batch method, the mini-batch size is 500, the total iterations for 10 epochs is X2.

Find X1+X2.

Answer: 1210 (ok4 ok4)

+/- 1

X1=10*1

X2=10*(60000/500)=1200.

Testing samples are not involved in the calculation.

//

CMSC57074 assignment 2, (241108a). For year 24-25, Q14-Q17

////////////////////////////////////

A57074.2.14: adaboost
In this assignment, we use the notations as given in the lecture notes. You may calculate the result with the help of a calculator or computer.

A set of training data (X) and their classes (Y) as shown below.

X=[u v] is the coordinates of the sample.

X has 2 classes Y: “pos” and “neg”

neg=[16 38
2 54
32 4
42 10
30 42];

pos=[22 39
4 33
-22 -25
-37 -31
-23 -48];

Just after the calculation of step t =2, the highest normalized weight D is D_high, the lowest normalized weight D is D_low. find D_high - D_low.

Answe14: = 0.214 (ok4)

+/- 0.005

D_high =0.250

D_low =0.036

D_high- D_low= 0.250-0.036=0.214

At time 2

t= 2,i= 1, err =0.125,alpha=0.973,D_current(1)=0.062, correct_i(1)=-1.000, D_next(1)=0.250

t= 2,i= 2, err =0.125,alpha=0.973,D_current(2)=0.062, correct_i(2)=-1.000, D_next(2)=0.250

t= 2,i= 3, err =0.125,alpha=0.973,D_current(3)=0.062, correct_i(3)=1.000, D_next(3)=0.036

t= 2,i= 4, err =0.125,alpha=0.973,D_current(4)=0.062, correct_i(4)=1.000, D_next(4)=0.036

t= 2,i= 5, err =0.125,alpha=0.973,D_current(5)=0.062, correct_i(5)=1.000, D_next(5)=0.036

t= 2,i= 6, err =0.125,alpha=0.973,D_current(6)=0.250, correct_i(6)=1.000, D_next(6)=0.143

t= 2,i= 7, err =0.125,alpha=0.973,D_current(7)=0.250, correct_i(7)=1.000, D_next(7)=0.143

t= 2,i= 8, err =0.125,alpha=0.973,D_current(8)=0.062, correct_i(8)=1.000, D_next(8)=0.036

t= 2,i= 9, err =0.125,alpha=0.973,D_current(9)=0.062, correct_i(9)=1.000, D_next(9)=0.036

t= 2,i=10, err =0.125,alpha=0.973,D_current(10)=0.062, correct_i(10)=1.000, D_next(10)=0.036

//

A57074.2.15: Neural network training
Run the following code in COLAB

https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/keras/overfit_and_underfit.ipynb

and answer the following questions. Partial credit will be given for correct answers.

References :

https://colab.research.google.com/ (how to use colab)

https://www.tensorflow.org/datasets/catalog/higgs

Which of the following statements are true?

Question num	Question	Answer: T=True, F=false
1	Each sample in the Higgs data set has 28 features.	T.
2	The first feature of a sample is the class label	T.

3	BUFFER_SIZE+ BATCH_SIZE+ N_TRAIN +BUFFER_SIZE+BATCH_SIZE + STEPS_PER_EPOCH=32000.	F. because print(BUFFER_SIZE+ BATCH_SIZE+ N_TRAIN +BUFFER_SIZE+BATCH_SIZE)=31000
4	In the “tiny” model, the number of biases=17.	T. Because biases=16+1
5	During training the learning rate is kept stable.	F. because the learning rate is inversely proportional to epoch.
6	Without weight regularization, the “tiny” model can avoid overfitting compared to other models under the same testing environment.	T. because…” typically, only the "Tiny" model manages to avoid overfitting altogether,..”
7	L2 regularization adds the absolute value of the weight coefficients, while L1 regularization adds the square of the value of the weight coefficients to the total loss during weight update.	F. It is the reverse.
8	L1 regularization performs better to reduce overfitting than L2 regularization.	F.
9	The “dropout” regularization method can improve overfitting.	T.
10	In this experiment the best performance in term of reducing overfitting is Combined L2 + dropout.	T.

Ok4

//

A57074.2.16: CNN Colab, ch9 programming

Run the CNN system found in this link

https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/images/cnn.ipynb

Based on this CNN system, after the step “Add Dense layers on top” is executed, if the total number of weights used is W and biases is B. Find W + B of this network.

Hints: Please read these documents for more information

tensorflow - what is the default activation function of dense layer in keras - Stack Overflow/

machine learning - What is the role of "Flatten" in Keras? - Stack Overflow

Answer15 : 122570 (ok4 ok4)

+/- 0

Model.summary (overall)

Model: "sequential"

Layer (type)	Output Shape	Param # (W+B)

Model: "sequential"

Layer (type)	Output Shape	Param #

- Layer 1: conv2d (Conv2D) (None, 30, 30, 32) 896
- Layer2: max_pooling2d (MaxPooling2D) (None, 15, 15, 32) 0
- Layer3: conv2d_1 (Conv2D) (None, 13, 13, 64) 18496
- Layer4: max_pooling2d_1 (MaxPooling2D) (None, 6, 6, 64) 0
- layer5: conv2d_2 (Conv2D) (None, 4, 4, 64) 36928
- layer6: flatten (Flatten) (None, 1024) 0
- layer7: dense (Dense) (None, 64) 65600
- layer8: dense_1 (Dense) (None, 10) 650

Total params: 122570 (478.79 KB)
Trainable params: 122570 (478.79 KB)
Non-trainable params: 0 (0.00 Byte)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%layer 1
% We can compare our calculations with the above model.sumary
% Layer1: from 3 inputs of 32x32, and kernel 3x3, each output filter size(32-
3+1)^2=30x30.
% W1=Weights used = n_input*(kenel_size)*n_output_filters=3*(3*3)*32=864
% B1=biases used (one for each kernel used)= n_output_filters=32
% Hence W1+B1=864+32=896 (Param #)
W1=3*(3*3)*32 %=864
B1=32
W1+B1 %=896
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Layer 2
%Layer2: max pooling has no weight/bias, but the output size =1/2 of input, so output
=15x15
%W2=B2=0
W2=0,B2=0 % no  weights/bias
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%Layer3
%Layer3: input is 15x15 using kernel 3x3, so output is (15-3+1)x(15-3+1)=13x13
%W3= weights used =n_input_fillters*(kernel_size)*n_output_filters=32*(3x3)*64=18432
%B3= biases used =n_output_filters=64
%W3+B3=18432+64=18496

W3= 32*(3*3)*64 %=18432
B3= 64
W3+B3%=18432+64=18496
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Layer4
%Layer4: max_pooling2d_1 (MaxPooling2D)   (None, 6, 6, 64)          0
%W4=B5=0 (no weights/bias for max polling), output size is halved, rounded off to 6x6
```

```
W4=0, B4=0 % (no weights/bias for max polling), output size is halved, rounded off to
6x6
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%layer5: input is 6x6, kernel is 3x3. Output size is (6-3+1)x(6-3+1) =4x4
%layer5: input is 6x6, kernel is 3x3. Output size is (6-3+1)x(6-3+1) =4x4
%W5=n_input_filters*(kernel_size)*n_output_filters=64*(3*3)*64=36864
%B5= n_output_filters=64. Because Each output has a bias.

W5=64*(3*3)*64 %=36864
B5=64 %. Because Each output has a bias.
W5+B5 %= %=36864+64=36928
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Layer6: flatten has no weight /bias, n_output=4*4*64=1024 ( a vector of 1024
elements).
%Because each input filter is of size 4x4 (see layer5), there are 64 filters in
layer5.
W6=0,B6=0
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Layer 7(dense layer/ fully connected): each input is connected to an output by a
weight
%N_input=1x1024, n_output=1x64. Each output has a bias, Bias= number of output filters
%W7=1024*64=65536
%B7=64
%W7+B7=65600

W7=1024*64 %=65536
B7=64
W7+B7 %=65600
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Layer8: is dense layer (fully connected),
%W8=n_input*n_output=64*10=640
%B8=n_output=10=10
%W8+B8=650

W8=64*10 %=640
B8=10
W8+B8 %=650
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
WW=W1+W2+W3+W4+W5+W6+W7+W8
BB=B1+B2+B3+B4+B5+B6+B7+B8
"% WW+BB ,just to verify, should be 122570 as shown in the result of the program "
WW+BB %      122570
```

//

A57074.2.17: Transfer learning, ch9 programming

In transfer learning the processing steps are:

1) Obtain the pre-trained model
2) Create a base model
3) Freeze layers
4) Add new trainable layers
5) Train the new layers on the dataset
6) Improve the model via fine-tuning

Study the process of transfer learning by running the demo code at

https://colab.research.google.com/github/keras-team/keras-io/blob/master/guides/ipynb/transfer_learning.ipynb

Just after the step “Build a model”.

Total number of parameters = x1

Number of trainable parameters. = x2

Just after the step “Do a round of fine-tuning of the entire model”

Total number of parameters = x3

Number of trainable parameters. = x4

Note: In the final tuning step there are non-trainable parameters which are used for data normalization.

Find x1+x2+x3+x4.

Answer: 62542208 (ok4)
+/- 0

% %after Build a model
% Total params: 20,863,529 (79.59 MB)
% Trainable params: 2,049 (8.00 KB)
% Non-trainable params: 20,861,480 (79.58 MB)
x1=20863529
x2=2049

% //////////////////
% after :Do a round of fine-tuning of the entire model
%
% Total params: 20,867,629 (79.60 MB)
% Trainable params: 20,809,001 (79.38 MB)
% Non-trainable params: 54,528 (213.00 KB)
% Optimizer params: 4,100 (16.02 KB)
% Fitting the end-to-end model
% ////////////////////////
x3=20867629
x4=20809001

x1+x2+x3+x4
%=62542208