Birla Institute of Technology & Science, Pilani
Work Integrated Learning Programmes Division
First Semester 2023-2024
M.Tech. in AIML

Mid-Semester Test
(EC-2 Regular Paper)

Course No. : AIMLCZG530
Course Title : Natural Language Processing
Nature of Exam : Closed Book
Weightage : 30%
Duration : 2 Hours
Date of Exam : 21-01-2024_FN

No. of Pages = 3
No. of Questions = 5

Note to Students:
1. Please follow all the *Instructions to Candidates* given on the cover page of the answer book.
2. All parts of a question should be answered consecutively. Each answer should start from a fresh page.
3. Assumptions made if any, should be stated clearly at the beginning of your answer.

Question 1. [4 Marks] Introduction

a) Identify the type of NLP application (i.e. Text categorization, Language Modeling, Named Entity Recognition) for each of the functionalities mentioned below. **[2 marks]**

Classifying emails into spam or non-spam
"Look ahead" typing where user is prompted with the next few words to type in an email
Sentiment Analysis
Classifying entities into pre-defined labels

Solution

Classifying emails into spam or non-spam	**Text categorization**
"Look ahead" typing where user is prompted with the next few words to type in an email	**Language Modeling**
Sentiment Analysis	**Text categorization**
Classifying entities into pre-defined labels	**Named entity recognition**

b) Which of the following two sentences has a "Structural ambiguity", and which one has a "Lexical ambiguity" – explain the ambiguities in 1 or 2 sentences? **[2 marks]**

- She saw a girl with a binoculars on the beach
- I saw bats

Solution & Marking Scheme:

- She saw a girl with a binoculars on the beach
  Structural ambiguity. – 1 mark
  Interpretation 1: The girl was on the beach with binoculars and she saw her.
  Interpretation 2:She was on beach with her binoculars and saw a girl.
- I saw bats
  Lexical ambiguity. – 1 mark

Interpretation 1: Bats as mammal
Interpretation 2: Bats as a cricket bats

Question 2. [4 Marks] n –gram language modeling

a) A machine translation system has to decide which of the following is the right sequence of words for a translation:
- I gave a book
- I book gave a

i. Given the following count of words in a corpus, compute the unigram probabilities for each of the above sentences. Indicate which is the more likely sequence of words to be used after translation. **[2 marks]**

ii. Now compute the bigram probabilities for each of the above sentences. Indicate which is the more likely sequence of words. **[2 marks]**

Unigram count matrix

<s>	I	gave	a	book
4	2	2	3	2

Bigram count matrix

		W_n				
		I	gave	a	book	</s>
W_{n-1}	<s>	1	0	0	0	0
	I	0	1	1	0	0
	gave	0	0	1	0	0
	a	0	0	0	2	0
	cherry	0	0	0	0	1

Unigram probabilities:

Unigram probability matrix

<s>	I	gave	a	book
4/13	2/13	2/13	3/13	2/13

P(I gave a book) = P(I) x P(gave) x P(a) x P(book)
= (2/13) x (2/13) x (3/13) x (2/13) = $24/(13^4)$ = 8.4E-4
P(I book gave a) = P(I) x P(book) x P(gave) x P(a)
= (2/13) x (2/13) x (2/13) x (3/13) = 8.4E-4
Since there is no difference in the probabilities of each of the unigrams, there is no way of figuring out which is the right sequence of words for translation.

Bigram probabilities:

		W_n				
		I	gave	a	book	</s>
W_{n-1}	<s>	1/4	0/4	0/4	0/4	0/4
	I	0/2	1/2	1/2	0/2	0/2
	gave	0/2	0/2	1/2	0/2	0/2
	a	0/3	0/3	0/3	2/3	0/3
	book	0/2	0/2	0/2	0/2	1/2

P(I gave a book) = P(I | <s>) x P(gave | I) x P(a | gave) x P(book | a) x P(</s> | book)
= (1/4) x (1/2) x (1/2) x (2/3) x (1/2) = 2/96 = 1/48
P(I book gave a) = P(I | <s>) x P(book | I) x P(gave | book) x P(a | gave) x (P </s> | a)
= (1/4) x (0/2) x (0/2) x (1/2) x (0/3) = 0
The first sequence has a higher probability than the second sequence and will be chosen by the machine translation system.

Question 3. [4 Marks]

One of the entries from a restaurant review website has the following statement: “The gulab jamoon is really quite good here”. Use this statement as one of the entries in the training dataset, and the word embeddings for each word in this statement as given in the Table below:

	dim_1	dim_2	dim_3
the	0.1	0.2	0.3
gulab	0.4	0.5	0.6
jamoon	0.7	0.8	0.9
is	0.2	0.1	0.4
really	0.5	0.7	0.2
quite	0.9	0.6	0.3
good	0.3	0.8	0.5
here	0.6	0.1	0.7

Explain how you will set-up each of the following:

a. A Neural Network based sentiment classifier, with a hidden layer consisting of 3 nodes, that can flag the restaurant / food reviews into one of the following classes i) GOOD ii) BAD iii) AVERAGE. **[2 marks]**

b. A Neural Network based “word predictor” that uses three context words and a 4 node hidden layer to predict the immediately following word. Assume that the context words are “jamoon is really”, and network has to be trained for the word that follows, “quite” **[2 marks]**

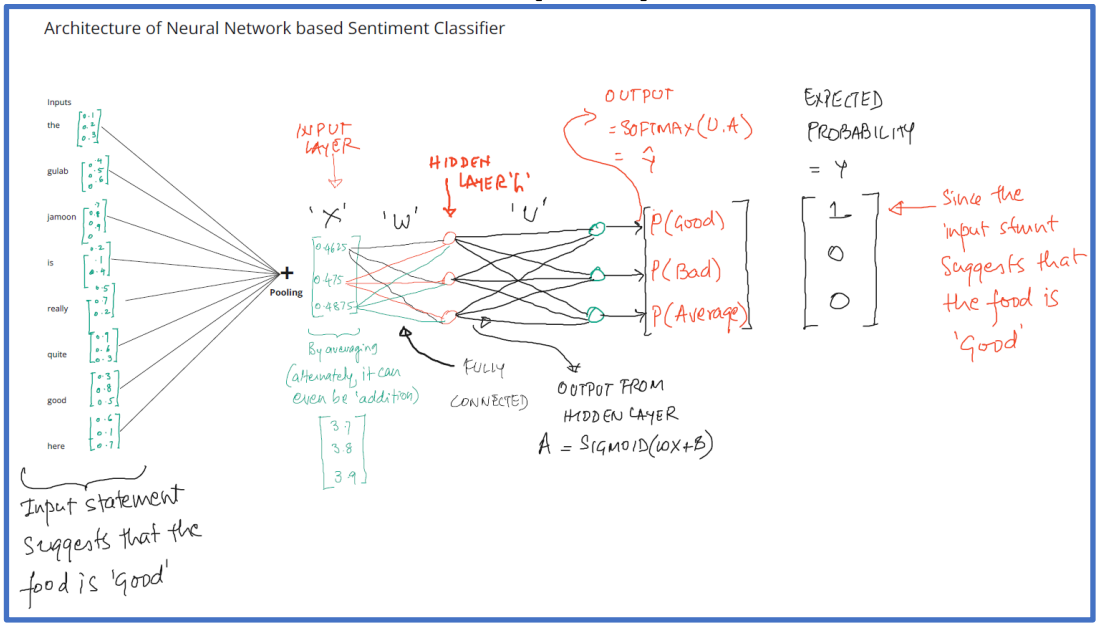
Your answer the question should contain the following:

i. The architecture diagram of the Neural Network, clearly indicating the input layer (using, appropriately, values of the word embedding’s given above), and schematics of all the other required intermediate steps / layers / connections and the outputs generated.
ii. Names of the activation function(s) used in the intermediate and output layers
iii. The expected output (y)

Note: You are NOT required to calculate the values of the weights and the outputs.

SOLUTIONS

Neural Network Based Sentiment Classifier [2 marks]

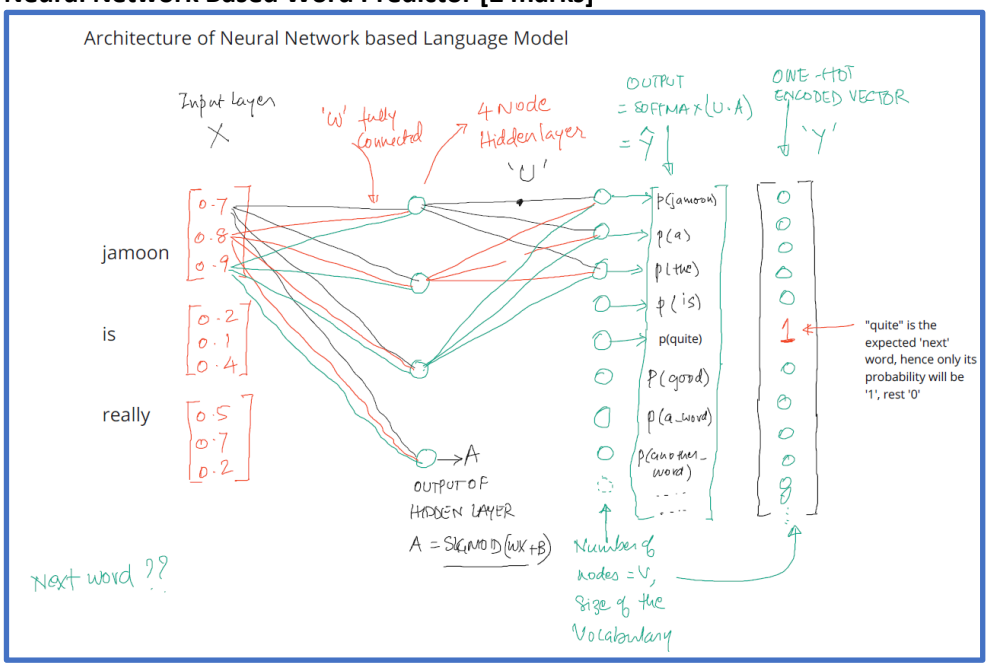


Important points to be looked for in the answer:

- Pooling of embeddings of all the words in the sentence into a single vector. Pooling can be done using either the sum or the mean of all the embeddings
- Ensure that the hidden layer has 3 nodes

- Ensure that the activation function is mentioned w.r.t the output of the hidden layer. It can be either SIGMOID or RELU
- Ensure that 3 nodes are shown in the output layer, and the activation function is SOFTMAX
- Ensure that the probability values in the **y** vector are correctly mentioned – GOOD should have an associated probability value of '1', other two should have '0'

Neural Network Based Word Predictor [2 marks]



Important points to be looked for in the answer:

- Input layer is formed by concatenating the embeddings of the 3 context words "jamoon is really"
- Ensure 4 nodes are shown in the hidden layer.
- Output from the hidden layer can be either SOFTMAX or RELU
- It is mentioned that the output layer has as many entries as the vocabulary
- Activation function at the output layer is SOFTMAX

In the **y** vector, only the word **quite** h

Question 4. (4 Marks)

Consider the following statements:

a) The cat saw the dog.

b) The dog barked at the cat.

Convert all the words to lowercase and WITHOUT further pre-processing the sentences (i.e. DO NOT remove stop words / lemmatization / stemming / etc.) carry out the following tasks:

1.	Create a vocabulary from the sentences	1
2.	Establish the Term Frequency and Document Frequency	1
3.	Establish the TF-IDF vector for each document	2

Note:

For Term Frequency use the simple formula (word frequency)/(sentence length)

Wherever required, use 'Natural Logarithm' in your calculations

SOLUTION

1. Vocabulary from the sentences [1]

V = [the,cat,saw,dog,barked,at]

2. Term Frequency and Document Frequency [1]

Term Frequency Calculation

" The cat saw the dog.”	" The dog barked at the cat."
• the: 2/5 • cat: 1/5 • saw: 1/5 • dog: 1/5	• the: 2/6 • dog: 1/6 • barked: 1/6 • at: 1/6 • cat: 1/6

Document Frequency (DF) Calculation

- the: 2
- cat: 2
- saw: 1
- dog: 2
- barked:1
- at:1

3. TF-IDF Calculation for each statement [1]

- $\text{IDF}(t) = \log(\frac{N}{df(t)})$
- N is the total number of documents (sentences) = 2
- $df(t)$ is the number of documents containing the term t

Note: Use ‘Natural Logarithm’ in your calculations

TF-IDF Scores

" The cat saw the dog.”	" The dog barked at the cat."
the: (2/5) * log(2/2) cat: (1/5) * log(2/2) saw: (1/5) * log(2/1) dog: (1/5) * log(2/2)	the: (2/6) * log(2/2) dog: (1/6) * log(2/2) barked: (1/6) * log(2/1) at: (1/6) * log(2/1) cat: (1/6) * log(2/2)

TF-IDF Vectors

	the	cat	saw	dog	barked	at
Sentence 1	0	0	0.138629	0	0	0
Sentence 2	0	0	0	0	0.115525	0.115525

Question 5.

Consider the following sentence:

"I bank on my best friend to accompany me to the bank located near the river bank."

Your task is to train a classifier such that, given the tuple ("bank","located") where "bank" is the target word and "located" is the candidate context word, the classifier returns the probability that "located" is a real context word for "bank".

- Provide the updated Input weight matrix for the Target Word after one iteration of the Word2Vec algorithm

Support your answer with detailed steps and rationale on the logic and computation. **[5 marks]**

The following additional information are provided:

- Use Word2Vec with Skip Gram Classifier with a Single Hidden Layer

- Negative Sampling words have been specified for you and they are "Purple", "Rain" and
- The One Hot Encoded Input Vectors are:

Bank	[1 0 0 0 0]
Located	[0 1 0 0 0]
Purple	[0 0 1 0 0]
Rain	[0 0 0 1 0]

d. Initial Embedding Matrix for the Single Hidden Layer

Bank	0.1	0.2	0.3
Located	0.2	0.3	0.4
Purple	0.3	0.4	0.5
Rain	0.4	0.5	0.5

e. Initial Embedding Matrix for the Output Layer

Bank	0.2	0.3	0.4
Located	0.3	0.4	0.5
Purple	0.4	0.5	0.6
Rain	0.5	0.4	0.6

Learning Rate = 0.05
Activation Function is Sigmoid

Solution:

Step 1 – Forward Propagation (Hidden Layer) [1 mark]

- The One Hot Encoded Input Matrix: (I)

  [10000
  01000
  00100
  00010]

- Initial Embedding Matrix (W_{input})

  [0.1 0.2 0.3
  0.2 0.3 0.4
  0.3 0.4 0.5
  0.4 0.5 0.5]

- Hidden Layer (h) for Target word “bank” = $W^{T}_{input} * I$

  = [0.1
  0.2
  0.3]

	Context Embedding			Input word embedding	dot product	Sigmoid()	t	Sig - t	
Located	0.3	0.4	0.5	0.1	0.26	0.564636292	1	-0.435363708	**Positive Context Word**
Purple	0.4	0.5	0.6	0.2	0.32	0.420675748	0	0.420675748	**Negative Word**
Rain	0.5	0.4	0.6	0.3	0.31	0.423114739	0	0.423114739	**Negative Word**
Context Embedding				Sig - t	C*(Sig-t)				
Located	0.3	0.4	0.5	-0.435363708	-0.130609112	-0.174145483	-0.217681854		**Positive Context Word**
Purple	0.4	0.5	0.6	0.420675748	0.168270299	0.210337874	0.252405449		**Negative Word**
Rain	0.5	0.4	0.6	0.423114739	0.211557369	0.169245896	0.253868843		**Negative Word**
				Derivative of log loss with respect to input embedding	0.249218556	0.205438286	0.288592438		
				LR	0.05				
				Update step	0.012460928	0.010271914	0.014429622		
				Updated Embedd	**0.087539072**	**0.1897281**	**0.2855704**		
				Ans (3 decimal	**0.088**	**0.190**	**0.286**		

Step 2 – Forward Propagation (Sigmoid Output Layer) [2 mark]

W_{output} (context) *for (located,purple,rain)*

[0.3 0.4 0.5
0.4 0.5 0.6
0.5 0.4 0.6]

Output Layer = W_{output}t * h

= [0.26
0.32
0.31]

Applying Sigmoid Activation,

For Positive Samples: $\sigma(x) = \frac{1}{1+e^{-x}}$

For Negative Samples: $\sigma(x) = \frac{1}{1+e^{x}}$

= [0.5646
0.4207
0.4231]

Step 3 – Prediction Error [1 mark]

Prediction Error = – 1-hot encoded vector for context

= [0.5646 [1 [-0.4354
0.4207 -- 0 = 0.4207
0.4231] 0] 0.4231]

Backward Propagation (computing W_{input}) step:

Derivative of Loss with respect to Input Word Embeddings for the target word “bank”:

C * (Sig-t) =

[0.2492	**0.2054**	**0.2886]**

(W_{input})

Step 4 - Updated Weight Matrix by applying Learning Rate [1 mark]

Learning Rate = 0.05 (given)

W^{new}_{input} = [0.1 0.2 0.3] – 0.05 * W_{input} =

0.088	**0.190**	**0.286**

This the updated Input weight matrix for the Target Word "bank" after one iteration of the Word2Vec algorithm

Question 6. [5 Marks]

a) Using HMM tagger to disambiguate the POS tag for the word "chase" in the following sentence, given the transition probabilities and emission probabilities below: **[5 marks]**

- "Cut to the chase"

Emission probabilities

	The	Cut	to	chase
VB	0	0.5	0	0.5
TO	0	0	1	0
NN	0	0.5	0	0.5
Det	1	0	0	0

- Transition probabilities

	VB	TO	NN	Det
<s>	0.2	0.01	0.2	0.6
VB	0.0	0.35	0.47	0.70
TO	0.83	0	0.47	0.4
NN	0.40	0.2	0.2	0.2
Det	0.12	0.0	0.23	0

SOLUTION

Cut to the chase

Possible taggings are:

i. VB □ TO □ Det □ NN
ii. VB □ TO □ Det □ VB
iii. NN □ TO □ Det □ NN
iv. NN □ TO □ Det □ VB

We are interested in disambiguating only the word "chase" in the above phrase. Hence the computations will be:

P(NN|Det) = 0.23
P(VB|Det) = 0.12
P(chase | NN) = 0.5
P(chase | VB) = 0.5

For tagging (i): P(NN|Det) x P(chase|NN) = 0.23 x 0.5 = 0.115
For tagging (ii): P(VB|Det) x P(chase|VB) = 0.12 x 0.5 = 0.06

Hence, tagging (i) is the preferred POS tag, i.e.
VB □ TO □ Det □ NN

Question 7. [5 Marks]

Fill up the Viterbi table for the sentence – '**I will'**. The tag transition probabilities and word emission probabilities, for the corpus used, are given below:

Tag transition probabilities	MD	VB	PRP
MD	0.05	0.5	0.001
VB	0.007	0	0.01
PRP	0.91	0.01	0.0001
START	0.01	0.49	0.5

Word emission	I	will
MD	0	0.7
VB	0	0
PRP	1	0

Viterbi Table	I	will
VB		
MD		
PRP		

PRP: PERSONAL PRONOUN

MD:MODAL
VB:VERB BASE FORM

Answer:

Viterbi Table	I	will
VB	0	0
MD	0	0.3185
PRP	0.5	0