End Semester Examination

Course Name: Natural Language Processing **Code: CS 563**
Full Marks-80 **Time: 3 hours**

Answer ALL the questions

Make reasonable assumptions as and whenever necessary. You can answer the questions in any sequence. However, answers to all the parts of any particular question should appear together.

SP

(Q1). Consider the following 2 sentences:

a. Ram saw Sita with a telescope which he dropped accidentally
b. Ram saw Sita with a telescope which she dropped accidentally

Part-A

Use the following grammar rules and draw the constituency parse trees:

Structural Rules:

S --> NP VP| S S'
S' -> WH S
NP --> N | N PP | PRON
VP --> V ADV
PP --> P NP

Lexical Rules:

WH --> which | where | who | what | how | why | whom
N --> Ram | Sita
PRON --> he | she
V --> saw | dropped
ADV --> accidentally
P --> with | in | by | to | from

3 + 3 = 6

Part-B:

Draw the Dependency Trees for the 2 sentences starting from the special symbol "Root". You do not have to label the edges. Show the head-modifier dependencies correctly.

3 +3 = 6

concep

(Q2). Remember the "loss" function (actually the "gain" function, since the value is maximized) of the word2vec algorithm in its skip gram version. It maximizes the conditional probability of the context word(s) given the target word. Assume that the constituency and dependency trees of the sentences are somehow available. Modify the "loss" function to incorporate the "knowledge" of parse trees.

4+4=8

SA **(Q3).** Compare and contrast the word2vec, glove and fasttext algorithms for generating word embeddings. **6**

SA **(Q4).** Consider the following training and test instances for sentence-level sentiment analysis.

Training samples

It was very expensive for what you get.	Negative
Certainly not the best camera.	Negative
I thought it was expensive, however, the awesome features justify it.	Positive
You do not need to worry about the backup.	Positive

Test sample

It was not expensive at all.	

Train a feed-forward neural network for sentence-level sentiment analysis. Consider the parameters of the network as follows:

- Hidden layer size: 3 neurons, Activation function: ReLu.
- Output classifier: Softmax
- Loss function: Mean-Squared-Error (MSE)
- Optimizer: Gradient-descent
- Weight matrix: W(*inp,hid*) and W(*hid,out*)
- Epoch: 3 or error becomes zero (whichever is earlier).

SA a. Use bag-of-word (BoW) feature for training your model (you may omit the irrelevant tokens considering its sentiment class. For e.g. you may ignore *it*, *was*,... etc. as irrelevant for the first sentence). **3**

b. Initialize the weight matrices randomly (all weights should not be the same), and update them for each epoch. Show all the computations at each step. **14**

c. After training, predict the sentiment of the test sample utilizing the trained model. **3**

(Q5). Consider the problem of Part-of-Speech (PoS) tagging with four possible tags (N, V, ADJ, DT):

Design a 3-layer feed-forward neural network (**Input**: word embedding representations of context word window of size 3, i.e. embeddings of previous and next word along with the current word; **Output**: PoS tag for the current word)

a. Write down and explain the input and output of each layer with appropriate dimension of weight matrix, input, output and bias term. Also use the suitable non-linear function with proper justification. **7**

b. State with proper mathematical formulations how the output PoS tag can be computed considering the Softmax activation on the last layer of output. **3**

c. Suppose your network predicts each possible PoS tag with equal probability. What will be the cross-entropy error on a single example? **3**

d. It is usual that all the context words are not equally useful to predict the PoS tag of the current word. To tackle this situation, design a strategy to obtain the *weight* (β) for word embedding of each context word. The strategy should be implemented in line with an end-to-end neural network based PoS tagger. State and explain the strategy using appropriate mathematical notations to obtain the weighted word embedding of the context words (***Hint:*** *You may use feed-forward NN to re-compute the embedding*).

Input: $x_1, x_2, x_3, \ldots\ldots x_k$ (Word Embedding for each word)

Output: $\beta_1 x_1, \beta_2 x_2, \beta_3 x_3, \ldots\ldots \beta_k x_k$ (Weighted word embeddings)

7

(Q6). a. Formulate the framework for Statistical Machine Translation (SMT), and explain how its different parameters can be computed. **4**

b. Consider the following (English-Hindi) parallel corpus: **10**

- Election of India: Bharat ka chunaav
- Election of England: England ka chunaav

Suppose the SMT system employs an *IBM1 model* for translation. Show the first three iterations to calculate the translation probability (***Hint:*** *assume uniform probability for initialization; Explain each step*).