

**DSO 560 – Text Analytics & Natural Language Processing**

**Instructor: Yu Chen**

**Final Exam (Practice)**

# Due Tuesday, May 10th, 8:30pm PST, 90 minutes

# No exams will be accepted past 8:35pm PST

**Instructions:**

* **WRITE ALL ANSWERS ON SEPARATE SHEETS OF PAPER**
* **SCAN EACH PAGE (AS A PDF OR IMAGE) AND SEND TO ME AND THE TA VIA SLACK.**
* **DO ALL SECTIONS.**

# ONCE YOU SUBMIT YOUR EXAM, YOU CAN LEAVE CLASS

# SHOW ALL WORK TO RECEIVE PARTIAL CREDIT

**Short Answer (5 pts, recommended 30 minutes)**

***Pick 5 of the short answer questions below to answer. Write no more than 2 sentences in your explanation. Each question is 1pt: 0.5pts for the correct answer and 0.5pts for a correct explanation.***

***Use the following word2vec embedding table for Q1-Q2***

|  |  |  |  |
| --- | --- | --- | --- |
| **Word** |  |  |  |
| **A** | **-1.0** | **2.0** | **1.0** |
| **B** | **1.0** | **0** | **1.0** |
| **C** | **2.0** | **-2.0** | **0** |
| **D** | **1.0** | **-1.0** | **0** |
| **NULL/UNKNOWN** | **0** | **0** | **0** |

1. *Which pair of words would have the highest cosine similarity score with each other? Why?*

C and D. If you compute their cosine similarity, you’ll get

1. *Write what the input sequence to a sequential model would look like for the document “A B A D” and sequence length 5.*

The sequence would look like the following. Notice that the 5th sequence is the NULL token:

[

[-1.0, 2.0, 1.0],

[1.0, 0.0, 1.0],

[-1.0, 2.0, 1.0],

[1.0, -1.0, 0.0],

[0.0, 0.0, 0.0]

]

1. *Write a regex with named capture groups that extracts the username and domain (username@domain) from an email address for only .com, .biz, and .net addresses.*

See [here](https://regexr.com/6kr1r).

1. *Provide an example of when an* ***InvalidContinuationByte*** *error could occur.*

Let’s say you had the Chinese character 车. Using a [site to look up the binary](https://unicode-table.com/en/8F66/), UTF-8, it would be

11101000 10111101 10100110

This is 3 bytes in UTF-8, since its Unicode codepoint is very large (15252902). UTF-8 indicates that the character is 3 bytes using continuation bytes.

Table

Description automatically generated

So here is the same bits with the continuation bits highlighted:

**1110**1000 **10**111101 **10**100110

Let’s say something weird happened when saving the data to disk. Now, the

**1110**1000 **11**111101 **10**100110

The yellow highlighted bits are not expected. The 1110 tells the computer to expect 2 more bytes that each start with 10, but the next byte starts with 11. This will raise an **InvalidContinuationByte** error.

1. ***What is one way a Naïve Bayes model could deal with a test document containing a token it has not seen before during training?***

***One technique is using Laplace smoothing (see from the notebook notes):***

***Text, letter

Description automatically generated***

1. *What is the unicode codepoint for the character* ***A****? What is its binary representation?*

Its Unicode codepoint is 65. Its binary representation would be 01000001.

*Provide the binary encoding for the character 🥰 in UTF8. Identify which bits are continuation bytes.*

This is similar to the above question – you can look up the [Unicode codepoint for this character on this site, for instance](https://unicode-table.com/en/1F970/), and then get its UTF-8 binary:

11110000 **10**011111 **10**100101 **10**110000

The 1111 to start indicates that this character is a sequence of 4 bytes.

***The following sections will be condensed, and only 2 of the following sections will appear. You can look through prior years’ exams – not much has changed / will change:***

|  |
| --- |
| **Naïve Bayes (3 pts, recommended 15 minutes)**  **Vectorization and Similarity (3 pts, recommended 15 minutes)**  **N-Gram Language Models (3 pts, recommended 15 minutes)** |

**True/False (5 pts, recommended 20 minutes)**

Pick 5 of the statements below, indicate if it is true or false. **In both cases (true or false), explain your reasoning in a brief sentence. Each question is worth 1pt: 0.5pts for the correct answer, 0.5pts for explanation.**

1. *The documents “John loves cats” and “cats love John” would have the same vector if we used TF-IDF or word count vectorization.*

True. Because TF-IDF / word counts are BOW (bag of words) models that don’t consider the sequence of the tokens, the frequencies would match.

1. In a Hidden Markov Model, the word we observe at sequence step 2 is determined by the word we observe at sequence step 1.

*False. The word we observe at sequence 2 is determined entirely by the hidden state at sequence 2. Note that there is NO arrow from the word at sequence 1 to the word at sequence 2:*

*Diagram

Description automatically generated*

1. If you want to make sure you keep human-readable tokens, you should pick lemmatization over stemming.

*True. Lemmatization will preserve the token as actual words. Stemming will often reduce the original word into its base form.*

1. According to Zipf Law, there is a relatively uniform distribution of word counts across all the words in a corpus’ vocabulary.

*False. Zipf Law says there is an extremely uneven distribution of counts across all words, that more closely reflects an exponential distribution:*

Diagram

Description automatically generated with medium confidence

1. Using a longer sequence length for your RNN models will reduce the chance of vanishing/exploding gradients.

*False. Using longer sequence lengths will make your RNN model more prone to vanishing/exploding gradients, since the weight matrices will be multiplied with themselves more times (the part in the circle below is the weight matrix w being multiplied by itself):*

*Diagram

Description automatically generated*

1. LSTMs calculate all of their hidden states across all sequences at once.

*False. LSTMs are sequential models like RNNs. They calculate each hidden state iteratively, sequence step by sequence step. Transformer models calculate all hidden states at the same time.*