Text, logo

Description automatically generated with medium confidence

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

**Instructor: Yu Chen**

**Final Exam**

# Due Tuesday, May 10th, 8:35pm PST, 90 minutes

# No exams will be accepted past 8:40pm PST

**Instructions:**

* **WRITE/TYPE ALL ANSWERS ON SEPARATE PAPER OR DOCUMENT**
* **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

**COLLABORATING, SHARING, OR DISCUSSING THIS EXAM/ITS CONTENTS WITH ANYONE BEFORE MAY 11th, 2022 IS AN ACADEMIC INTEGRITY VIOLATION**.

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

***Pick 5 of the short answer questions below to answer. Write no more than 3 sentences in your explanation. Each question is 1pt: 0.5pts for the correct answer and 0.5pts for a correct explanation.***

1. Explain what the window size of a skipgram/CBOW model controls and when you might increase/decrease it.

**The window size determines how many tokens to the left/right of the target word you’d include in your training samples.**

**If you feel like longer-range dependencies between words are relevant, you would increase the window size. For example, perhaps in more formal speech/text. However, if you are working with text that has very loose connections to surrounding words (just short phrases), then your window size can be set lower.**

1. Provide an example of two documents with different text that would have a cosine similarity of 1.

**Any document where the ratio of the tokens are the same will have the same cosine similarity. For instance,**

* ***love love movie movie***
* ***love movie***

1. You are doing analysis on current events reporting and want to capture the frequency of references to the current U.S. president, Joe Biden. You find that he is frequently referred to as “Biden”, “Joe Biden”, “President Biden”, or “the President” in news articles. Write an efficient regex pattern that captures all references to Joe Biden (no explanation needed).

**Use this** [**example**](https://regexr.com/6l4su)**. Each of the 4 representations is worth 0.25 pts.**

1. You want to write your own TF-IDF implementation that more heavily weights rare tokens (tokens that very rarely appear in the documents of the corpus). Which of the following TF/IDF functions would you select?

|  |  |  |
| --- | --- | --- |
| **Option A** | **Option B** | **Option C** |
| TF = n(t,d) x 2  IDF = 1 + | TF = n(t,d)  IDF = 1 + | TF =  IDF = 2 x |

**You would pick Option C, since its IDF function doubles the score of the IDF portion and divides the score of the TF portion by 2. This will result in more of the final TF-IDF score being contributed by the IDF portion.**

1. Suppose you are using a Hidden Markov Model to classify named entities (PERSON, PLACE, NON\_NAMED\_ENTITY are your labels) on text. Identify and explain
   1. What are your observed states?
   2. What are your hidden states?
   3. What would the values in your transition matrix represent?
   4. What would the values in your emission matrix represent?

**Observed state would be the actual tokens themselves. Hidden states would be the labels (PERSON, PLACE, NON\_NAMED\_ENTITY). The transition matrix would contain the probabilities of transitioning from one label to another (eg. PERSON 🡪 PLACE). The emission matrix would contain the conditional probabilities of seeing a word given a hidden state (label), for ex. P(“Biden” | PERSON).**

1. What type of text would be more likely to suffer more from vanishing/exploding gradients with an RNN, all else being equal?
   1. Tweets
   2. BBC news articles
   3. SMS text messages

**BBC News articles, since they are much longer on average than tweets or SMS text messages. Since RNNs are sequential models, the longer the sequence length of the documents, the more likely it will suffer from vanishing/exploding gradients all else being equal.**

**Vectorization and Similarity (3 pts, recommended 20 minutes)**

You work as a data scientist working at Nordstrom. Your company has conducted several consumer research surveys, with consumers filling in open-response questions about the outfit combinations they would be most willing to spend money on items. Here is what 3 customers wrote:

**Customer A:** retro woven skirt loose-fit  
**Customer B:** casual one-piece shirts casual  
**Customer C.** casual woven loose-fit one-piece

Assume you perform text preprocessing via lemmatization.

1. Generate **TF-IDF document vectors** (you may write them as a matrix or table). Calculate IDF for each of the words, then term frequency (TF) for each of document – word combinations (**1pt**). Use the following term frequency and inverse document frequency functions:

n(t,d) 🡪 the number of times token t appears in document d

df(t) 🡪 the number of documents token t appears in

|  |  |
| --- | --- |
| TF = n(t,d) | IDF = 1 + |

1. A new customer has entered his preferences: **woven casual shirt.** Assuming **TF-IDF vectorized** documents and cosine similarity, is this new customer’s preferences more similar to Customer A or Customer B? (**1pt**)
2. Assume now that a colleague has trained the following 3-dimension **word2vec** word embeddings on the open-ended survey responses. The results are below. (**1pt**)

|  |  |  |  |
| --- | --- | --- | --- |
| **OOV (unknown/out of vocabulary)** | 0 | 0 | 0 |
| **Casual** | -2 | 2 | -1 |
| **Retro** | 1 | -2 | 0 |
| **One-piece** | -1 | 0 | 1 |
| **Woven** | 3 | -2 | -1 |
| **Skirt** | 1 | -2 | 0 |
| **Loose-fit** | -1 | 3 | -1 |

Based on these embedding vectors and using Euclidean distance as your distance measure, is **casual** more similar to the token **loose-fit** or the token **formal-wear**? Calculate each pair’s distance and show your work.

**Naïve Bayes (2 pts, recommended 10 minutes)**

You work at NBC Universal as a data analyst are analyzing social media comments to gauge how much interest there is to see an upcoming TV show. After seeing the pilot, several users indicated their interest/lack of interest along with open text comments.

**Interested Documents**

|  |
| --- |
| 1. Silly but fun and funny 2. Seems Funny in a stupid wholesome way 3. Fun, silly, and |

**Not Interested Documents**

|  |
| --- |
| 1. So stupid 2. Seems silly 3. Not funny at all, garbage |

**Stopwords to remove**: *to, but, in, a, and*

You **do not need to perform stemming or lemmatization, and can disregard punctuation / case-sensitivity** (ie. *Can’t = can’t*).

1. What are the prior probabilities? **(0.5 pts)**
2. A new comment is posted: **Seems funny and silly but stupid.** Assume a Naïve Bayes model with conditional independence and unigram tokens. Calculate the posterior probabilities (**1.5pts**)

**True/False (5 pts, recommended 30 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/real-life example.**

1. After performing Latent Semantic Analysis on our text dataset for topic modelling, we can use the decomposed matrices to gauge the relative “strength” of each topic.

**True. Latent Semantic Analysis is essentially SVD (Singular Value Decomposition). We can use the values on the diagonal of the middle matrix (which is r x r) to get the “strength” or importance of each topic (latent dimension).**

1. After dimensionality reduction using PCA, the number of reduced dimensions are usually far less and are now highly correlated with each other.

**False. The number of dimensions after PCA is usually much less than the original number of dimensions, but these dimensions will no longer have correlations with each other (they are de-correlated).**

1. Adding word boundaries to a regex pattern, for example r’\bboy\b’ will improve precision during information retrieval.

**True. If you don’t use word boundaries for many patterns, such as r‘boy’, you’ll end up with many false positives, like “boycott” or “cowboy”. False positives will decrease your precision metric.**

1. Using fuzzy matching via libraries like fuzzywuzzy, we would find that semantically similar words like “canine” and “dog” have extremely high similarity scores.

**False. Word2vec / Glove word embeddings might show high degree of similarity of these words, but fuzzy matching uses edit distance. Since “canine” and “dog” are nothing alike in terms of spelling, they’d have a very low similarity.**

1. Unlike word2vec, GloVe embeddings will change depending on the context of the word in the document.

**False. Both word2vec and GloVe are static embeddings. Attention-based transformer models such as BERT with adjust the final embedding based on position/context.**

1. Compared to RNNs, LSTMs have architectures that better allow modeling of longer-range dependencies between tokens across many sequence steps.

**True. In addition to a hidden state, LSTMs have “memory” vectors that decide how much to remember/forget from longer-term relationships.**

1. Because it is a variable-length encoding scheme, UTF8 uses continuation bytes to indicate that a current byte is part of a longer sequence of bytes, since characters can be more than 1 byte (8 bits long).

**True. For higher Unicode codepoints, UTF8 will need to use more bytes to represent the character. Whenever it requires more than 1 byte to represent the character, it will use continuation bytes.**