# CSI5386: Natural Language Processing

# Assignment 1

## Devansh Kumar: 300374674

## Souleymane Sankara: 300100940

Part 1**: Corpus processing (legal text): tokenization and word counting**

1. **First 20 lines from Output.txt**

CO-BRANDING

AND

ADVERTISING

AGREEMENT

THIS

CO-BRANDING

AND

ADVERTISING

AGREEMENT

(

the

``

Agreement

''

)

is

made

as

of

June

1. How many tokens did you find in the corpus? How many types (unique tokens) did you have? What is the type/token ratio for the corpus? The type/token ratio is defined as the number of types divided by the number of tokens.

|  |  |
| --- | --- |
| Total Tokens | 4,789,386 |
| Unique Tokens | 58,691 |
| Type/Token Ratio | 0.012254 |

1. For each token, print the token and its frequency in a file called tokens.txt (from the most frequent to the least frequent) and include the first 20 lines in your report.

, 240576

the 239941

of 151491

to 127054

and 125196

. 117446

or 102674

) 78092

( 75436

in 73941

\* 67765

any 58848

-- 58711

shall 48423

a 46208

by 41853

be 39157

Agreement 37007

for 35430

this 35215

1. How many tokens appeared only once in the corpus?

**Tokens Appearing Only Once: 24,151**

1. From the list of tokens, extract only words, by excluding punctuation and other symbols, if any. Please pay attention to end of sentence dot (full stops). How many words did you find? List the top 20 most frequent words in your report, with their frequencies. What is the type/token ratio when you use only words (called lexical diversity)?

|  |  |
| --- | --- |
| Total Tokens(Excluding Punctuation) | 3,810,668 |
| Unique Words | 37,022 |
| Type/Token Ratio(Words Only) | 0.009715 |

**Top 20 Most Frequent Words:**

the: 239941

of: 151491

to: 127054

and: 125196

or: 102674

in: 73941

any: 58848

shall: 48423

a: 46208

by: 41853

be: 39157

Agreement: 37007

for: 35430

this: 35215

such: 34814

with: 32571

as: 31625

that: 27281

other: 25137

is: 21533

1. From the list of words, exclude stopwords. List the top 20 most frequent words and their frequencies in your report. You can use this list of stopwords (or any other that you consider adequate). Also compute the type/token ratio when you use only word tokens without stopwords (called lexical density)?

|  |  |
| --- | --- |
| Total Content Words (Excluding Stopwords) | 2,149,290 |
| Unique Content Words | 36,666 |
| Lexical Density | 0.017060 |

**Top 20 Most Frequent Content Words:**

shall: 48423

Agreement: 37007

Party: 20520

may: 13160

Section: 12355

party: 11574

Company: 10848

including: 9582

Product: 8745

use: 7968

provided: 7868

Parties: 7684

time: 7425

set: 6806

written: 6732

applicable: 6471

information: 6330

forth: 6238

right: 6216

rights: 6191

1. Compute all the pairs of two consecutive words (bigrams) (excluding stopwords and punctuation). List the most frequent 20 pairs and their frequencies in your report.

**Most Frequent Bigrams:**

set forth: 6018

Agreement shall: 3505

Confidential Information: 2868

written notice: 2371

Effective Date: 2262

Party shall: 2230

third party: 2143

Third Party: 1916

terms conditions: 1902

prior written: 1807

forth Section: 1687

shall deemed: 1679

without limitation: 1657

time time: 1656

shall mean: 1630

Intellectual Property: 1626

including without: 1462

shall provide: 1430

shall right: 1357

written consent: 1323

|  |  |
| --- | --- |
| # of tokens (b) | 4,789,386 |
| # of types (b) | 58,691 |
| type/token ratio (b) | 0.012254 |
| tokens appeared only once (d) | 24,151 |
| # of words (excluding punctuation) (e) | 3,810,668 |
| type/token ratio (excluding punctuation) (e) | 0.009715 |
| List the top 3 most frequent words and their frequencies (e) | the: 239941 |
| of: 151491 |
| to: 127054 |
| type/token ratio (excluding punctuation and stopwords) (f) | 0.017060 |
| List the top 3 most frequent words and their frequencies (excluding stopwords) (f) | shall: 48423 |
| Agreement: 37007 |
| Party: 20520 |
| List the top 3 most frequent bigrams  and their frequencies (g) | set forth: 6018 |
| Agreement shall: 3505 |
| Confidential Information: 2868 |

**Implementation:**

We used the code provided by “Kevin” on the Brightspace discussion for the purpose of concatenating the 510 .txt files into one.

We started out with using the popular spaCy tokenizer for tokenizing, it was taking a lot of time mainly because when the spacy tokenizes it also simultaneously processes the “parser”, “ner” and “tagger”, once we disabled and also set the Max\_chunk size as 500000 because of the following error  
“"""text of length 26807133 exceeds maximum of 1000000. The parser and NER models require roughly 1GB of temporary memory per 100,000 characters in the input. This means long texts may cause memory allocation errors. If you're not using the parser or NER, it's probably safe to increase the nlp.max\_length limit. The limit is in number of characters, so you can check whether your inputs are too long by checking len(text).

"""” we got our results in around 3 mins 47 seconds.

However, the spacy tokenizer gave us a a lot of whitespaces in the final output and also broke down terms that add more value when together like “Co-branding”. Example of spacy tokenizer is submitted as file output\_spacy.txt. Therefore we decided to use **the “NLTK” tokenizer** which worked really well.

**Analysis:**

The corpus consists of 4,789,386 tokens and 58,691 unique tokens, resulting in a low type/token ratio (0.012254), indicating high redundancy and repetition. The most frequent tokens include common function words like "the," "of," and "to," highlighting the dominance of stopwords. However, after filtering out punctuation and symbols, the number of words reduces to 3,810,668, with a slightly improved lexical diversity of 0.009715. Removing stopwords(downloaded from the nltk library) further isolates content words (2,149,290 tokens), showing a lexical density of 0.017060

Part 2 Report: **Evaluation of pre-trained sentence embedding models**

1. Sentence Embedding models
   1. All-MiniLM-L6-v2

* Maps a sentence or short paragraph to 384-dimensional dense vector
* Built with 6 transformers layers
* Based on Bert architecture
  1. All-MPNet-Base-V2
* Maps sentences and paragraphs into 768-deimensional dense vector
* Fined-tuned from pre-trained Microsoft MPNet-Base model
* Built with 12 layers
  1. T5-Large
* A variant of the T5 model family, which include T5-small, T5-medium and T5-large
* Used 24 layers
* Maps sentences and paragraphs into 1024-dimensional vector
  1. MSMARCO-RoBERTa-Base-v2
* Produce embedding in a 768-dimensional dense vector space
  1. All-DistilRoBERTa-v1
* Maps text into 768-dimensional vector space

1. Performance Evaluation

To compute similarity, we used cosine similarity as the metric. The results were then scaled using the formula:  
This approach allowed us to derive a performance score ranging from 0 to 5

1. Results

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Datasets | SE1 | SE2 | SE3 | SE4 | SE5 | Best score |
| STS2016.gs.answer-answer.txt | 0.725 | 0.748 | 0.840 | 0.719 | 0.732 | 0.840 |
| STS2016.gs.headlines.txt | 0.796 | 0.843 | 0.837 | 0.782 | 0.828 | 0.843 |
| STS2016.gs.plagiarism.txt | 0.829 | 0.823 | 0.864 | 0.804 | 0.849 | 0.864 |
| STS2016.gs.postediting.txt | 0.857 | 0.880 | 0.880 | 0.834 | 0.885 | 0.885 |
| STS2016.gs.question-question.txt | 0.805 | 0.821 | 0.805 | 0.784 | 0.794 | 0.821 |

1. Discussion

T5-Large (SE3), a larger model, gives strong performance achieving then the best score in 2 of the 5 trials. For the answer-answer dataset it outperforms all other models by significant margin. Although, we can notice that All-MPNet-Base-v2 (SE2) which is a smaller model gives competitive performance with also the best score in 2 cases out of the 5. This means that SE2 can be an alternative, especially in scenarios requiring smaller computational characteristics.

SE1 ((All-MiniLM-L6-v2) and SE5 (All-DistilRoBERTa-v1) shows promising results despite their smaller architecture. They offer a good trade-off between accuracy and efficiency

**Contribution:**

|  |  |
| --- | --- |
| Devansh Kumar | Part 1, Report |
| Souleymane Sankara | Part 2, Report |

**References:**

Nltk: <https://www.nltk.org/>

Huggingface sentence\_transformers: <https://huggingface.co/sentence-transformers>

Pytorch: <https://pypi.org/project/torch/>

Sentence Embedding Models: <https://huggingface.co/sentence-transformers>