**Week 3 - Lab Exercise: Text Visualisation and Analysis using spaCy**

**Objective:**

In this lab, you will use spaCy to process and clean text data, then visualise word frequencies and word clouds using Seaborn and WordCloud libraries.

**Prerequisites:**

* Jupyter Notebook is set up and running.
* The following Python libraries are installed: spacy, seaborn, matplotlib, and wordcloud.
* Note: spacy may not work with Python 3.13, so you may need to downgrade to 3.12
* The en\_core\_web\_sm model is downloaded for spaCy using the command:

!python -m spacy download en\_core\_web\_sm

**Instructions:**

**Step 1: Load the spaCy Model and Process Text**

Insert the code below into your Jupyter Notebook to load the spaCy English model and process two sample texts.

import spacy

# Load the spaCy English model

nlp = spacy.load("en\_core\_web\_sm")

# Sample text 1

text1 = "Hanoi is a beautiful city with a rich culture. The traffic can be overwhelming, but the food is amazing!"

# Process text with spaCy

doc = nlp(text1)

tokens = [token.text for token in doc]

print(len(tokens), " tokens")

print(tokens)

# Sample text 2

text2 = "Okay, so, like, you know, today—well, actually, uh, I mean, technically, it's, um, February 18th, 2025, right? And, well, Mo, yeah, that’s Mo El-Haj, uh, is, you know, kinda showing us, or, well, more like trying to show us how to, um, I guess, properly visualize, or should I say analyse, text, you know, like, online, at, umm, the, uh, VinUniversity, which, by the way, is in, uh, Hanoi, Vietnam, you know? I mean, honestly, I didn’t even realise, but, well, here we are, sitting, listening, and watching, kinda waiting for something, and then, boom! He’s like, ‘Hey, let’s clean some text,’ and, I mean, obviously, you know, we have to do it right, otherwise, well, it just doesn’t make sense, right? But, like, I dunno, there’s, like, a lot of unnecessary words and, um, extra spaces, and also, weird punctuation—like this! Or… maybe this? You know what I mean? And, uh, that’s why, well, uh, stopwords, yeah, those have to go, and lemmas, too, like, totally important. So, yeah, here we are, watching Mo, at VinUniversity, doing his thing, and, uh, yeah, that’s what’s happening, I guess?"

# Process text with spaCy

doc = nlp(text2)

tokens = [token.text for token in doc]

print(len(tokens), " tokens")

print(tokens)

**Step 2: Visualise Word Frequencies and Word Clouds**

from collections import Counter

import seaborn as sns

import matplotlib.pyplot as plt

# Function to plot word frequencies

def plot\_word\_frequencies(tokens, top\_n=10):

word\_freq = Counter(tokens)

sns.barplot(x=list(word\_freq.keys())[:top\_n], y=list(word\_freq.values())[:top\_n])

plt.xticks(rotation=45)

plt.show()

from wordcloud import WordCloud

# Function to generate a word cloud

def generate\_wordcloud(text):

wordcloud = WordCloud().generate(text)

plt.figure(figsize=(6, 6))

plt.imshow(wordcloud, interpolation="bilinear")

plt.axis("off")

plt.show()

# Generate visualisations for text1

generate\_wordcloud(text1)

plot\_word\_frequencies(tokens)

**Step 3: Remove Stopwords and Punctuation**

tokens\_no\_stop\_no\_punct = [token.text for token in doc if not token.is\_stop and not token.is\_punct]

print(len(tokens\_no\_stop\_no\_punct), " tokens")

print(tokens\_no\_stop\_no\_punct)

# Clean text for visualisation

clean\_text = " ".join(tokens\_no\_stop\_no\_punct)

generate\_wordcloud(clean\_text)

plot\_word\_frequencies(tokens\_no\_stop\_no\_punct)

**Step 4: Filter Tokens by Length**

tokens\_no\_stop\_no\_punct = [token.text for token in doc if not token.is\_stop and not token.is\_punct and len(token.text) > 2]

print(len(tokens\_no\_stop\_no\_punct), "tokens")

print(tokens\_no\_stop\_no\_punct)

clean\_text = " ".join(tokens\_no\_stop\_no\_punct)

generate\_wordcloud(clean\_text)

plot\_word\_frequencies(tokens\_no\_stop\_no\_punct)

**Step 5: Custom Stopwords and Number Removal**

custom\_stopwords = {"know", "like"}

tokens\_no\_stop\_no\_punct = [

token.text for token in doc

if not token.is\_stop and not token.is\_punct and len(token.text) > 3 and token.text.lower() not in custom\_stopwords and not token.like\_num

]

print(len(tokens\_no\_stop\_no\_punct), "tokens")

print(tokens\_no\_stop\_no\_punct)

clean\_text = " ".join(tokens\_no\_stop\_no\_punct)

generate\_wordcloud(clean\_text)

plot\_word\_frequencies(tokens\_no\_stop\_no\_punct)

**Task:**

1. Run each code block sequentially.
2. Observe how the number of tokens decreases as you remove stopwords, punctuation, and short words.
3. Compare the visualisations and analyse how cleaning the text improves the clarity of key words.

**Submission:**

* Save your Jupyter Notebook as spacy\_text\_visualisation\_lab.ipynb.
* Submit your notebook via canvus.

✅ **Lab complete! If you haven’t changed the code by trying different examples, please do so. If you have done that, now move to the Lab Assignment.**