PDF Text Converter and Generation Example

Mt. SAC CISB 63 Final Project - Fall 2023

Russell Luna

https://github.com/rmoon64/CISB63_Final/ (https://github.com/rmoon64/CISB63_Final/)

Summary/Explanation of Project

The goal of this project was to convert a PDF file into a TXT file and generate text using the extracted content. It utilized various natural language processing techniques, such as part-of-speech tagging (POS), named entity recognition (NER), translation, frequency distributions, word clouds, and tokenization. First, I utilized PyPDF2 to extract the text from the PDF. Then, tokenize the extracted sentences into individual words to identify the most frequently used words and determine the significance of sentences for generating word clouds. To enhance entity visualization, I used displacy, which helped visualize entities such as organizations. Finally, we used Matplotlib to plot the graph for our frequency distribution (fdist).

Import libraries

In [1]:

```
import tensorflow as tf
import re
import pandas as pd
import PyPDF2
import numpy as np
import nltk
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from nltk.tokenize import word_tokenize
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.corpus import stopwords

# Extract text
def convert_pdf_to_txt(pdf_path, txt_path):
    start_page = 17
    end_page = 35

    with open(pdf_path, 'rb') as pdf_file:
        pdf_reader = PyPDF2.PdfFileReader(pdf_file)
        text = ''
        for page_num in range(start_page - 1, min(end_page, pdf_reader.num
            page = pdf_reader.getPage(page_num)
            text += page.extractText()

    with open(txt_path, 'w', encoding='utf-8') as txt_file:
        txt_file.write(text)

# Add paths for PDF and text files
pdf_path = 'Maxwell Maltz - Psycho-Cybernetics_ Updated and Expanded Paper
txt_path = 'Output.txt'

# Call the function to convert the PDF to text
convert_pdf_to_txt(pdf_path, txt_path)
```

```
WARNING:tensorflow:From C:\Anaconda\Lib\site-packages\keras\src\losses.p
y:2976: The name tf.losses.sparse_softmax_cross_entropy is deprecated. P
lease use tf.compat.v1.losses.sparse_softmax_cross_entropy instead.
```

In [2]:

```
# Load the text
file_path = 'Output.txt'
with open(file_path, 'r', encoding='utf-8') as file:
    text = file.read()
```

In [3]:

```
# Tokenize the text
tokenizer = Tokenizer()
tokenizer.fit_on_texts([text])
total_words = len(tokenizer.word_index) + 1
```

In [4]:

```
# Data Transformation
txt_path = 'Output.txt'

# Read the text file into a DataFrame
df = pd.read_csv(txt_path, sep='\t')

# Use describe() to get statistical information about the DataFrame
print(df.describe())

# Use head() to preview the first five rows of the DataFrame
print(df.head())

# Use info() to get information about the DataFrame's structure and data t
print(df.info())
```

```
                                        General Principles
count                                                  595
unique                                                 595
top     The self-image is the key to human personality...
freq                                                     1
                                 General Principles
0  The self-image is the key to human personality...
1  Change the self-image and you change the perso...
2  But more than this: The self-image sets the bo...
3  accomplishment. It defines what you can and ca...
4  self-image and you expand the “area of the pos...
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 595 entries, 0 to 594
Data columns (total 1 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   General Principles  595 non-null    object
dtypes: object(1)
memory usage: 4.8+ KB
None
```

Create input sequences and corresponding labels

steps:

1. Initialize an Empty List
2. Tokenize the Text
3. Create N-gram Sequences

In [5]:

```
# Create input sequences and corresponding labels
input_sequences = []
for line in text.split('\n'):
    token_list = tokenizer.texts_to_sequences([line])[0]
    for i in range(1, len(token_list)):
        n_gram_sequence = token_list[:i+1]
        input_sequences.append(n_gram_sequence)
```

Data preparation

Let's prepare the input data by ensuring that all sequences have the same length, separating input sequences (X) from output labels (y), and one-hot encoding the output labels for training a language model.

steps:

1. Calculate Maximum Sequence Length
2. Pad Sequences
3. Create Input (X) and Output (y) Sequences
4. One-Hot Encode the Output (y)

X, y = input_sequences[:,:-1],input_sequences[:,-1]

- The X variable represents the input sequences, which include all elements of each sequence except the last one.
- The y variable represents the labels, which are the last elements of each sequence.
- X represents the input sequences for training the language model.
- input_sequences[:, :-1] selects all elements in each row of input_sequences except for the last one.
- X consists of the first to second-to-last elements of each sequence in input_sequences.
- Each row of X corresponds to an input sequence that the model will use to predict the next word.

In [6]:

```
max_sequence_length = max([len(x) for x in input_sequences])
input_sequences = pad_sequences(input_sequences, maxlen=max_sequence_lengt
X, y = input_sequences[:,:-1],input_sequences[:,-1]
y = tf.keras.utils.to_categorical(y, num_classes=total_words)
```

In [7]:

```
# Build the LSTM model
model = Sequential()
model.add(Embedding(total_words, 50, input_length=max_sequence_length-1))
# model.add(LSTM(150, return_sequences = True))
model.add(LSTM(100))
# model.add(Dense(150, activation = 'relu'))
model.add(Dense(total_words, activation='softmax'))

model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=[

model.summary()
```

```
WARNING:tensorflow:From C:\Anaconda\Lib\site-packages\keras\src\backe
nd.py:873: The name tf.get_default_graph is deprecated. Please use t
f.compat.v1.get_default_graph instead.

WARNING:tensorflow:From C:\Anaconda\Lib\site-packages\keras\src\optim
izers\__init__.py:309: The name tf.train.Optimizer is deprecated. Ple
ase use tf.compat.v1.train.Optimizer instead.

Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 53, 50)            81500

 lstm (LSTM)                 (None, 100)               60400

 dense (Dense)               (None, 1630)              164630

=================================================================
```

In [8]:

```
# Train the model
model.fit(X, y, epochs=100, verbose=1)
```

```
Epoch 1/100
WARNING:tensorflow:From C:\Anaconda\Lib\site-packages\keras\src\utils\tf
_utils.py:492: The name tf.ragged.RaggedTensorValue is deprecated. Pleas
e use tf.compat.v1.ragged.RaggedTensorValue instead.

WARNING:tensorflow:From C:\Anaconda\Lib\site-packages\keras\src\engine\b
ase_layer_utils.py:384: The name tf.executing_eagerly_outside_functions
is deprecated. Please use tf.compat.v1.executing_eagerly_outside_functio
ns instead.

251/251 [==============================] - 7s 19ms/step - loss: 6.0528 -
accuracy: 0.0362
Epoch 2/100
251/251 [==============================] - 5s 19ms/step - loss: 5.3567 -
accuracy: 0.0655
Epoch 3/100
251/251 [==============================] - 5s 19ms/step - loss: 5.1777 -
accuracy: 0.0736
Epoch 4/100
251/251 [==============================] - 5s 19ms/step - loss: 5.0743 -
accuracy: 0.0884
Epoch 5/100
251/251 [==============================] - 5s 19ms/step - loss: 4.9795 -
accuracy: 0.0896
Epoch 6/100
251/251 [==============================] - 5s 19ms/step - loss: 4.9084 -
accuracy: 0.0970
Epoch 7/100
251/251 [==============================] - 5s 18ms/step - loss: 4.8402 -
accuracy: 0.1018
Epoch 8/100
251/251 [==============================] - 5s 18ms/step - loss: 4.7625 -
accuracy: 0.1074
Epoch 9/100
251/251 [==============================] - 5s 18ms/step - loss: 4.6828 -
accuracy: 0.1092
Epoch 10/100
251/251 [==============================] - 5s 18ms/step - loss: 4.5966 -
accuracy: 0.1156
Epoch 11/100
251/251 [==============================] - 5s 18ms/step - loss: 4.5070 -
accuracy: 0.1234
Epoch 12/100
251/251 [==============================] - 5s 18ms/step - loss: 4.4096 -
accuracy: 0.1338
Epoch 13/100
251/251 [==============================] - 5s 18ms/step - loss: 4.3117 -
accuracy: 0.1413
Epoch 14/100
251/251 [==============================] - 5s 18ms/step - loss: 4.2095 -
accuracy: 0.1503
Epoch 15/100
251/251 [==============================] - 5s 18ms/step - loss: 4.1050 -
accuracy: 0.1674
Epoch 16/100
251/251 [==============================] - 5s 18ms/step - loss: 4.0052 -
accuracy: 0.1709
```

```
Epoch 17/100
251/251 [==============================] - 5s 18ms/step - loss: 3.9055 -
accuracy: 0.1834
Epoch 18/100
251/251 [==============================] - 5s 19ms/step - loss: 3.8082 -
accuracy: 0.1926
Epoch 19/100
251/251 [==============================] - 5s 19ms/step - loss: 3.7105 -
accuracy: 0.2075
Epoch 20/100
251/251 [==============================] - 5s 19ms/step - loss: 3.6158 -
accuracy: 0.2177
Epoch 21/100
251/251 [==============================] - 5s 19ms/step - loss: 3.5205 -
accuracy: 0.2271
Epoch 22/100
251/251 [==============================] - 5s 19ms/step - loss: 3.4255 -
accuracy: 0.2392
Epoch 23/100
251/251 [==============================] - 5s 19ms/step - loss: 3.3353 -
accuracy: 0.2474
Epoch 24/100
251/251 [==============================] - 5s 21ms/step - loss: 3.2428 -
accuracy: 0.2635
Epoch 25/100
251/251 [==============================] - 5s 21ms/step - loss: 3.1492 -
accuracy: 0.2790
Epoch 26/100
251/251 [==============================] - 5s 21ms/step - loss: 3.0605 -
accuracy: 0.2952
Epoch 27/100
251/251 [==============================] - 5s 21ms/step - loss: 2.9671 -
accuracy: 0.3155
Epoch 28/100
251/251 [==============================] - 5s 21ms/step - loss: 2.8806 -
accuracy: 0.3335
Epoch 29/100
251/251 [==============================] - 5s 21ms/step - loss: 2.7943 -
accuracy: 0.3459
Epoch 30/100
251/251 [==============================] - 5s 21ms/step - loss: 2.7075 -
accuracy: 0.3695
Epoch 31/100
251/251 [==============================] - 5s 21ms/step - loss: 2.6226 -
accuracy: 0.3913
Epoch 32/100
251/251 [==============================] - 5s 22ms/step - loss: 2.5431 -
accuracy: 0.4065
Epoch 33/100
251/251 [==============================] - 5s 22ms/step - loss: 2.4640 -
accuracy: 0.4246
Epoch 34/100
251/251 [==============================] - 6s 22ms/step - loss: 2.3915 -
accuracy: 0.4518
Epoch 35/100
251/251 [==============================] - 6s 22ms/step - loss: 2.3156 -
accuracy: 0.4615
```

```
Epoch 36/100
251/251 [==============================] - 6s 22ms/step - loss: 2.2439 -
accuracy: 0.4733
Epoch 37/100
251/251 [==============================] - 6s 23ms/step - loss: 2.1775 -
accuracy: 0.4920
Epoch 38/100
251/251 [==============================] - 6s 23ms/step - loss: 2.1103 -
accuracy: 0.5026
Epoch 39/100
251/251 [==============================] - 6s 23ms/step - loss: 2.0448 -
accuracy: 0.5149
Epoch 40/100
251/251 [==============================] - 6s 23ms/step - loss: 1.9867 -
accuracy: 0.5300
Epoch 41/100
251/251 [==============================] - 6s 23ms/step - loss: 1.9273 -
accuracy: 0.5441
Epoch 42/100
251/251 [==============================] - 6s 23ms/step - loss: 1.8724 -
accuracy: 0.5564
Epoch 43/100
251/251 [==============================] - 6s 23ms/step - loss: 1.8196 -
accuracy: 0.5673
Epoch 44/100
251/251 [==============================] - 6s 24ms/step - loss: 1.7687 -
accuracy: 0.5828
Epoch 45/100
251/251 [==============================] - 6s 24ms/step - loss: 1.7207 -
accuracy: 0.5917
Epoch 46/100
251/251 [==============================] - 6s 24ms/step - loss: 1.6703 -
accuracy: 0.6059
Epoch 47/100
251/251 [==============================] - 6s 24ms/step - loss: 1.6234 -
accuracy: 0.6136
Epoch 48/100
251/251 [==============================] - 6s 25ms/step - loss: 1.5813 -
accuracy: 0.6213
Epoch 49/100
251/251 [==============================] - 6s 25ms/step - loss: 1.5376 -
accuracy: 0.6319
Epoch 50/100
251/251 [==============================] - 6s 25ms/step - loss: 1.4980 -
accuracy: 0.6399
Epoch 51/100
251/251 [==============================] - 6s 25ms/step - loss: 1.4588 -
accuracy: 0.6467
Epoch 52/100
251/251 [==============================] - 7s 26ms/step - loss: 1.4191 -
accuracy: 0.6566
Epoch 53/100
251/251 [==============================] - 7s 26ms/step - loss: 1.3836 -
accuracy: 0.6648
Epoch 54/100
251/251 [==============================] - 7s 27ms/step - loss: 1.3490 -
accuracy: 0.6763
```

```
Epoch 55/100
251/251 [==============================] - 7s 29ms/step - loss: 1.3136 -
accuracy: 0.6837
Epoch 56/100
251/251 [==============================] - 7s 27ms/step - loss: 1.2812 -
accuracy: 0.6901
Epoch 57/100
251/251 [==============================] - 7s 28ms/step - loss: 1.2484 -
accuracy: 0.6988
Epoch 58/100
251/251 [==============================] - 7s 26ms/step - loss: 1.2189 -
accuracy: 0.7015
Epoch 59/100
251/251 [==============================] - 7s 27ms/step - loss: 1.1881 -
accuracy: 0.7083
Epoch 60/100
251/251 [==============================] - 7s 26ms/step - loss: 1.1610 -
accuracy: 0.7181
Epoch 61/100
251/251 [==============================] - 7s 27ms/step - loss: 1.1346 -
accuracy: 0.7201
Epoch 62/100
251/251 [==============================] - 7s 27ms/step - loss: 1.1042 -
accuracy: 0.7286
Epoch 63/100
251/251 [==============================] - 7s 28ms/step - loss: 1.0793 -
accuracy: 0.7352
Epoch 64/100
251/251 [==============================] - 7s 28ms/step - loss: 1.0561 -
accuracy: 0.7392
Epoch 65/100
251/251 [==============================] - 7s 28ms/step - loss: 1.0299 -
accuracy: 0.7434
Epoch 66/100
251/251 [==============================] - 7s 28ms/step - loss: 1.0087 -
accuracy: 0.7514
Epoch 67/100
251/251 [==============================] - 7s 29ms/step - loss: 0.9863 -
accuracy: 0.7549
Epoch 68/100
251/251 [==============================] - 7s 29ms/step - loss: 0.9642 -
accuracy: 0.7612
Epoch 69/100
251/251 [==============================] - 7s 28ms/step - loss: 0.9454 -
accuracy: 0.7677
Epoch 70/100
251/251 [==============================] - 7s 29ms/step - loss: 0.9246 -
accuracy: 0.7709
Epoch 71/100
251/251 [==============================] - 7s 29ms/step - loss: 0.9038 -
accuracy: 0.7758
Epoch 72/100
251/251 [==============================] - 7s 29ms/step - loss: 0.8864 -
accuracy: 0.7774
Epoch 73/100
251/251 [==============================] - 7s 29ms/step - loss: 0.8675 -
accuracy: 0.7793
```

```
Epoch 74/100
251/251 [==============================] - 7s 29ms/step - loss: 0.8506 -
accuracy: 0.7849
Epoch 75/100
251/251 [==============================] - 7s 29ms/step - loss: 0.8346 -
accuracy: 0.7906
Epoch 76/100
251/251 [==============================] - 7s 28ms/step - loss: 0.8190 -
accuracy: 0.7906
Epoch 77/100
251/251 [==============================] - 7s 28ms/step - loss: 0.8026 -
accuracy: 0.7937
Epoch 78/100
251/251 [==============================] - 7s 28ms/step - loss: 0.7881 -
accuracy: 0.7956
Epoch 79/100
251/251 [==============================] - 7s 28ms/step - loss: 0.7739 -
accuracy: 0.7976
Epoch 80/100
251/251 [==============================] - 7s 28ms/step - loss: 0.7590 -
accuracy: 0.8048
Epoch 81/100
251/251 [==============================] - 7s 28ms/step - loss: 0.7479 -
accuracy: 0.8016
Epoch 82/100
251/251 [==============================] - 7s 28ms/step - loss: 0.7343 -
accuracy: 0.8082
Epoch 83/100
251/251 [==============================] - 7s 28ms/step - loss: 0.7213 -
accuracy: 0.8119
Epoch 84/100
251/251 [==============================] - 7s 28ms/step - loss: 0.7100 -
accuracy: 0.8130
Epoch 85/100
251/251 [==============================] - 7s 28ms/step - loss: 0.6998 -
accuracy: 0.8128
Epoch 86/100
251/251 [==============================] - 7s 28ms/step - loss: 0.6895 -
accuracy: 0.8154
Epoch 87/100
251/251 [==============================] - 7s 28ms/step - loss: 0.6768 -
accuracy: 0.8178
Epoch 88/100
251/251 [==============================] - 7s 28ms/step - loss: 0.6673 -
accuracy: 0.8161
Epoch 89/100
251/251 [==============================] - 7s 28ms/step - loss: 0.6577 -
accuracy: 0.8239
Epoch 90/100
251/251 [==============================] - 7s 28ms/step - loss: 0.6484 -
accuracy: 0.8232
Epoch 91/100
251/251 [==============================] - 7s 28ms/step - loss: 0.6393 -
accuracy: 0.8244
Epoch 92/100
251/251 [==============================] - 7s 29ms/step - loss: 0.6300 -
accuracy: 0.8277
```

```
Epoch 93/100
251/251 [==============================] - 7s 28ms/step - loss: 0.6215 -
accuracy: 0.8265
Epoch 94/100
251/251 [==============================] - 8s 32ms/step - loss: 0.6124 -
accuracy: 0.8311
Epoch 95/100
251/251 [==============================] - 7s 30ms/step - loss: 0.6042 -
accuracy: 0.8322
Epoch 96/100
251/251 [==============================] - 7s 29ms/step - loss: 0.5945 -
accuracy: 0.8311
Epoch 97/100
251/251 [==============================] - 7s 29ms/step - loss: 0.5902 -
accuracy: 0.8310
Epoch 98/100
251/251 [==============================] - 7s 29ms/step - loss: 0.5822 -
accuracy: 0.8346
Epoch 99/100
251/251 [==============================] - 7s 29ms/step - loss: 0.5756 -
accuracy: 0.8351
Epoch 100/100
251/251 [==============================] - 7s 29ms/step - loss: 0.5677 -
accuracy: 0.8382
```

Out[8]:
```
<keras.src.callbacks.History at 0x19e8eddde50>
```

Function to generate text

Let's create a function generate_text that uses a trained language model to generate a sequence of words based on a given seed text.

Steps:

1. Function Definition
2. Tokenize the Seed Text
3. Pad Tokenized Sequence
4. Predict the Next Word
5. Map Index to Word
6. Update Seed Text
7. Return Generated Text

Note: generate_text is a function that takes four parameters:

- seed_text: The initial text or seed for generating the sequence.
- next_words: The number of words to generate.
- model: The trained language model.
- max_sequence_length: The maximum length of the input sequence.

In [9]:

```
# Function to generate text
def generate_text(seed_text, next_words, model, max_sequence_length):
    for _ in range(next_words):
        token_list = tokenizer.texts_to_sequences([seed_text])[0] # tokeni
        token_list = pad_sequences([token_list], maxlen=max_sequence_lengt

        #The model is used to predict the index of the next word in the se
        predicted = np.argmax(model.predict(token_list, verbose=0))

        # The predicted index is then mapped back to the corresponding wor
        output_word = ""
        for word, index in tokenizer.word_index.items():
            if index == predicted:
                output_word = word
                break

        # The predicted word is appended to the seed text, creating an upd
        seed_text += " " + output_word
    return seed_text
```

Generate text

- generate_text is a function that takes four parameters: a seed text ("Call me Ishmael"), the number of words to generate (20), the trained model (model), and the maximum sequence length (max_sequence_length). The function generates text by iteratively predicting the next word and appending it to the seed text.

In [44]:

```
# Generate text
generated_text = generate_text("Our self-image", 15, model, max_sequence_l
print(generated_text)
```

```
Our self-image prescribes the limits for the accomplishment of any exper
ience and a vicious or a beneficent
```

Adding WordCloud

In [46]:

```
import matplotlib.pyplot as plt
from wordcloud import WordCloud

# Generate a word cloud
wordcloud = WordCloud(width=800, height=400, background_color="white").ger

# Display the word cloud using matplotlib
plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis("off")
plt.show()
```

Adding Textblob

In [12]:

```
!pip install textblob
```

```
Requirement already satisfied: textblob in c:\anaconda\lib\site-packages
(0.17.1)
Requirement already satisfied: nltk>=3.1 in c:\anaconda\lib\site-package
s (from textblob) (3.8.1)
Requirement already satisfied: click in c:\anaconda\lib\site-packages (f
rom nltk>=3.1->textblob) (8.0.4)
Requirement already satisfied: joblib in c:\anaconda\lib\site-packages
(from nltk>=3.1->textblob) (1.2.0)
Requirement already satisfied: regex>=2021.8.3 in c:\anaconda\lib\site-p
ackages (from nltk>=3.1->textblob) (2022.7.9)
Requirement already satisfied: tqdm in c:\anaconda\lib\site-packages (fr
om nltk>=3.1->textblob) (4.65.0)
Requirement already satisfied: colorama in c:\anaconda\lib\site-packages
(from click->nltk>=3.1->textblob) (0.4.6)
```

In [13]:

```python
from textblob import TextBlob
```

In [55]:

```python
# Create a TextBlob object with the text
blob = TextBlob(generated_text)

# Translate the text to Spanish
translated_blob = blob.translate('en', 'es')

# Print the translated text
print(translated_blob)
```

```
Nuestra autoimagen prescribe los límites para el logro de cualquier expe
riencia y una viciosa o beneficiosa
```

Adding SpaCy

In [15]:

```
!pip install -U spacy
```

```
Requirement already satisfied: spacy in c:\anaconda\lib\site-packages
(3.7.2)
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in c:\anacond
a\lib\site-packages (from spacy) (3.0.12)
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in c:\anacond
a\lib\site-packages (from spacy) (1.0.5)
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in c:\anaconda
\lib\site-packages (from spacy) (1.0.10)
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in c:\anaconda\lib\si
te-packages (from spacy) (2.0.8)
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in c:\anaconda\lib
\site-packages (from spacy) (3.0.9)
Requirement already satisfied: thinc<8.3.0,>=8.1.8 in c:\anaconda\lib\si
te-packages (from spacy) (8.2.1)
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in c:\anaconda\lib\s
ite-packages (from spacy) (1.1.2)
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in c:\anaconda\lib\si
te-packages (from spacy) (2.4.8)
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in c:\anaconda\li
b\site-packages (from spacy) (2.0.10)
Requirement already satisfied: weasel<0.4.0,>=0.1.0 in c:\anaconda\lib\s
ite-packages (from spacy) (0.3.4)
Requirement already satisfied: typer<0.10.0,>=0.3.0 in c:\anaconda\lib\s
ite-packages (from spacy) (0.9.0)
Requirement already satisfied: smart-open<7.0.0,>=5.2.1 in c:\anaconda\l
ib\site-packages (from spacy) (5.2.1)
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in c:\anaconda\lib\si
te-packages (from spacy) (4.65.0)
Requirement already satisfied: requests<3.0.0,>=2.13.0 in c:\anaconda\li
b\site-packages (from spacy) (2.31.0)
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<3.0.0,>=1.7.4 in
c:\anaconda\lib\site-packages (from spacy) (1.10.8)
Requirement already satisfied: jinja2 in c:\anaconda\lib\site-packages
(from spacy) (3.1.2)
Requirement already satisfied: setuptools in c:\anaconda\lib\site-packag
es (from spacy) (68.0.0)
Requirement already satisfied: packaging>=20.0 in c:\anaconda\lib\site-p
ackages (from spacy) (23.1)
Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in c:\anaconda\li
b\site-packages (from spacy) (3.3.0)
Requirement already satisfied: numpy>=1.19.0 in c:\anaconda\lib\site-pac
kages (from spacy) (1.24.3)
Requirement already satisfied: typing-extensions>=4.2.0 in c:\anaconda\l
ib\site-packages (from pydantic!=1.8,!=1.8.1,<3.0.0,>=1.7.4->spacy) (4.
7.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\anaconda\l
ib\site-packages (from requests<3.0.0,>=2.13.0->spacy) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\anaconda\lib\site-pack
ages (from requests<3.0.0,>=2.13.0->spacy) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\anaconda\lib\sit
e-packages (from requests<3.0.0,>=2.13.0->spacy) (1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in c:\anaconda\lib\sit
e-packages (from requests<3.0.0,>=2.13.0->spacy) (2023.11.17)
Requirement already satisfied: blis<0.8.0,>=0.7.8 in c:\anaconda\lib\sit
e-packages (from thinc<8.3.0,>=8.1.8->spacy) (0.7.11)
Requirement already satisfied: confection<1.0.0,>=0.0.1 in c:\anaconda\l
ib\site-packages (from thinc<8.3.0,>=8.1.8->spacy) (0.1.4)
```

```
Requirement already satisfied: colorama in c:\anaconda\lib\site-packages
(from tqdm<5.0.0,>=4.38.0->spacy) (0.4.6)
Requirement already satisfied: click<9.0.0,>=7.1.1 in c:\anaconda\lib\si
te-packages (from typer<0.10.0,>=0.3.0->spacy) (8.0.4)
Requirement already satisfied: cloudpathlib<0.17.0,>=0.7.0 in c:\anacond
a\lib\site-packages (from weasel<0.4.0,>=0.1.0->spacy) (0.16.0)
Requirement already satisfied: MarkupSafe>=2.0 in c:\anaconda\lib\site-p
ackages (from jinja2->spacy) (2.1.1)
```

In [16]:

```
!python -m spacy download en_core_web_sm
```

```
Collecting en-core-web-sm==3.7.1
  Downloading https://github.com/explosion/spacy-models/releases/downloa
d/en_core_web_sm-3.7.1/en_core_web_sm-3.7.1-py3-none-any.whl (https://gi
thub.com/explosion/spacy-models/releases/download/en_core_web_sm-3.7.1/e
n_core_web_sm-3.7.1-py3-none-any.whl) (12.8 MB)
     ---------------------------------------- 0.0/12.8 MB ? eta -:--:--
     ---------------------------------------- 0.0/12.8 MB ? eta -:--:--
     ---------------------------------------- 0.1/12.8 MB 656.4 kB/s eta
0:00:20
      ---------------------------------------- 0.3/12.8 MB 2.0 MB/s eta
0:00:07
     -- ------------------------------------- 0.7/12.8 MB 3.8 MB/s eta
0:00:04
     ---- ----------------------------------- 1.5/12.8 MB 7.0 MB/s eta
0:00:02
     ------- -------------------------------- 2.4/12.8 MB 8.8 MB/s eta
0:00:02
     --------- ------------------------------ 3.1/12.8 MB 10.0 MB/s eta
0:00:01
     ------------ --------------------------- 4.1/12.8 MB 10.8 MB/s eta
0:00:01
     --------------- ------------------------ 4.9/12.8 MB 11.9 MB/s eta
0:00:01
     ------------------ --------------------- 6.0/12.8 MB 13.1 MB/s eta
0:00:01
     --------------------- ------------------ 7.0/12.8 MB 13.9 MB/s eta
0:00:01
     ----------------------- ---------------- 7.6/12.8 MB 13.9 MB/s eta
0:00:01
     ------------------------- -------------- 8.3/12.8 MB 14.0 MB/s eta
0:00:01
     ----------------------------- ---------- 9.4/12.8 MB 14.7 MB/s eta
0:00:01
     -------------------------------- ------ 10.8/12.8 MB 20.5 MB/s eta
0:00:01
     ------------------------------------ - 12.2/12.8 MB 21.8 MB/s eta
0:00:01
     ------------------------------------  12.8/12.8 MB 22.6 MB/s eta
0:00:01
     ---------------------------------------- 12.8/12.8 MB 20.4 MB/s eta
0:00:00
Requirement already satisfied: spacy<3.8.0,>=3.7.2 in c:\anaconda\lib\si
te-packages (from en-core-web-sm==3.7.1) (3.7.2)
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in c:\anacond
a\lib\site-packages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1)
(3.0.12)
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in c:\anacond
a\lib\site-packages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1)
(1.0.5)
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in c:\anaconda
\lib\site-packages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (1.
0.10)
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in c:\anaconda\lib\si
te-packages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (2.0.8)
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in c:\anaconda\lib
\site-packages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (3.0.9)
Requirement already satisfied: thinc<8.3.0,>=8.1.8 in c:\anaconda\lib\si
```

```
te-packages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (8.2.1)
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in c:\anaconda\lib\s
ite-packages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (1.1.2)
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in c:\anaconda\lib\si
te-packages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (2.4.8)
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in c:\anaconda\li
b\site-packages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (2.0.1
0)
Requirement already satisfied: weasel<0.4.0,>=0.1.0 in c:\anaconda\lib\s
ite-packages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (0.3.4)
Requirement already satisfied: typer<0.10.0,>=0.3.0 in c:\anaconda\lib\s
ite-packages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (0.9.0)
Requirement already satisfied: smart-open<7.0.0,>=5.2.1 in c:\anaconda\l
ib\site-packages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (5.2.
1)
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in c:\anaconda\lib\si
te-packages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (4.65.0)
Requirement already satisfied: requests<3.0.0,>=2.13.0 in c:\anaconda\li
b\site-packages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (2.31.
0)
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<3.0.0,>=1.7.4 in
c:\anaconda\lib\site-packages (from spacy<3.8.0,>=3.7.2->en-core-web-sm=
=3.7.1) (1.10.8)
Requirement already satisfied: jinja2 in c:\anaconda\lib\site-packages
(from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (3.1.2)
Requirement already satisfied: setuptools in c:\anaconda\lib\site-packag
es (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (68.0.0)
Requirement already satisfied: packaging>=20.0 in c:\anaconda\lib\site-p
ackages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (23.1)
Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in c:\anaconda\li
b\site-packages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (3.3.
0)
Requirement already satisfied: numpy>=1.19.0 in c:\anaconda\lib\site-pac
kages (from spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (1.24.3)
Requirement already satisfied: typing-extensions>=4.2.0 in c:\anaconda\l
ib\site-packages (from pydantic!=1.8,!=1.8.1,<3.0.0,>=1.7.4->spacy<3.8.
0,>=3.7.2->en-core-web-sm==3.7.1) (4.7.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\anaconda\l
ib\site-packages (from requests<3.0.0,>=2.13.0->spacy<3.8.0,>=3.7.2->en-
core-web-sm==3.7.1) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\anaconda\lib\site-pack
ages (from requests<3.0.0,>=2.13.0->spacy<3.8.0,>=3.7.2->en-core-web-sm=
=3.7.1) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\anaconda\lib\sit
e-packages (from requests<3.0.0,>=2.13.0->spacy<3.8.0,>=3.7.2->en-core-w
eb-sm==3.7.1) (1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in c:\anaconda\lib\sit
e-packages (from requests<3.0.0,>=2.13.0->spacy<3.8.0,>=3.7.2->en-core-w
eb-sm==3.7.1) (2023.11.17)
Requirement already satisfied: blis<0.8.0,>=0.7.8 in c:\anaconda\lib\sit
e-packages (from thinc<8.3.0,>=8.1.8->spacy<3.8.0,>=3.7.2->en-core-web-s
m==3.7.1) (0.7.11)
Requirement already satisfied: confection<1.0.0,>=0.0.1 in c:\anaconda\l
ib\site-packages (from thinc<8.3.0,>=8.1.8->spacy<3.8.0,>=3.7.2->en-core
-web-sm==3.7.1) (0.1.4)
Requirement already satisfied: colorama in c:\anaconda\lib\site-packages
(from tqdm<5.0.0,>=4.38.0->spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1)
```

```
(0.4.6)
Requirement already satisfied: click<9.0.0,>=7.1.1 in c:\anaconda\lib\si
te-packages (from typer<0.10.0,>=0.3.0->spacy<3.8.0,>=3.7.2->en-core-web
-sm==3.7.1) (8.0.4)
Requirement already satisfied: cloudpathlib<0.17.0,>=0.7.0 in c:\anacond
a\lib\site-packages (from weasel<0.4.0,>=0.1.0->spacy<3.8.0,>=3.7.2->en-
core-web-sm==3.7.1) (0.16.0)
Requirement already satisfied: MarkupSafe>=2.0 in c:\anaconda\lib\site-p
ackages (from jinja2->spacy<3.8.0,>=3.7.2->en-core-web-sm==3.7.1) (2.1.
1)
[+] Download and installation successful
You can now load the package via spacy.load('en_core_web_sm')
```

In [17]:

```
text = "A boy who was dropped from one college because of poor grades ente
```

In [18]:

```
# Perform standard imports
import spacy
nlp = spacy.load('en_core_web_sm')
```

In [19]:

```
# Import the displaCy library
from spacy import displacy
```

In [20]:

```
doc = nlp(text)
displacy.render(doc, style='ent', jupyter=True)
```

A boy who was dropped from one **CARDINAL** college because of poor grades entered Columbia **ORG** and became a straight “A” student. A girl who had flunked Latin **NORP** four times, after three **CARDINAL** talks with the school counselor, finished with a grade of 84 **CARDINAL** . A boy who was told by a testing bureau that he had no aptitude for English **LANGUAGE** won an honorable mention the next year **DATE** for a literary prize. The trouble with these students was not that they were dumb or lacking in basic aptitudes.

Customizing Colors and Effects

- You can also pass background color and gradient options:

In [21]:

```
colors = {'ORG': 'linear-gradient(135deg, green 20%, cyan)', 'CARDINAL': '

options = {'ents': ['ORG', 'CARDINAL'], 'colors':colors}

displacy.render(doc, style='ent', jupyter=True, options=options)
```

A boy who was dropped from one CARDINAL college because of poor grades entered Columbia ORG and became a straight “A” student. A girl who had flunked Latin four times, after three CARDINAL talks with the school counselor, finished with a grade of 84 CARDINAL . A boy who was told by a testing bureau that he had no aptitude for English won an honorable mention the next year for a literary prize. The trouble with these students was not that they were dumb or lacking in basic aptitudes.

POS

In [22]:

```
import nltk
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\RL\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\RL\AppData\Roaming\nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
```

Out[22]:

```
True
```

In [47]:

```
words = word_tokenize(generated_text)
```

In [48]:
```
nltk.pos_tag(words)
```

Out[48]:
```
[('Our', 'PRP$'),
 ('self-image', 'NN'),
 ('prescribes', 'VBZ'),
 ('the', 'DT'),
 ('limits', 'NNS'),
 ('for', 'IN'),
 ('the', 'DT'),
 ('accomplishment', 'NN'),
 ('of', 'IN'),
 ('any', 'DT'),
 ('experience', 'NN'),
 ('and', 'CC'),
 ('a', 'DT'),
 ('vicious', 'JJ'),
 ('or', 'CC'),
 ('a', 'DT'),
 ('beneficent', 'NN')]
```

In [49]:
```
#How many words are there? :
print (len(words))
```

```
17
```

In [50]:
```
#Import required libraries :
from nltk.probability import FreqDist
```

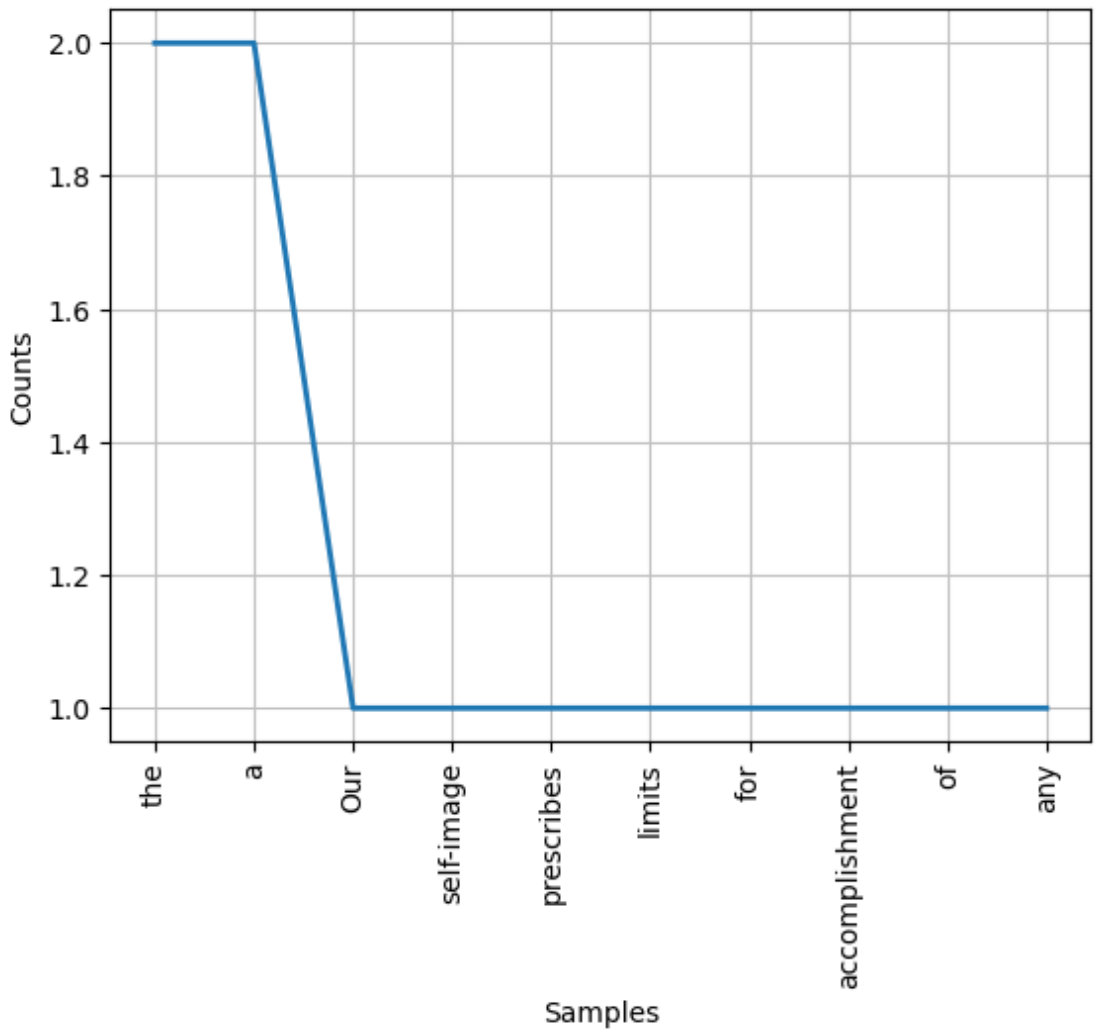
In [51]:
```
#Find the frequency :
fdist = FreqDist(words)
```

In [52]:
```
#Print 10 most common words :
fdist.most_common(10)
```

Out[52]:
```
[('the', 2),
 ('a', 2),
 ('Our', 1),
 ('self-image', 1),
 ('prescribes', 1),
 ('limits', 1),
 ('for', 1),
 ('accomplishment', 1),
 ('of', 1),
 ('any', 1)]
```

In [53]:
```
#Plot the graph for fdist :
import matplotlib.pyplot as plt
%matplotlib inline
```

In [54]:
```
fdist.plot(10)
```



Out[54]: <Axes: xlabel='Samples', ylabel='Counts'>

Conclusion

In general, I believe the program performed quite well. From reading the book, I could tell the program was making a lot of inferences from the text but because it was taking from the general language flow and tone the author used, I'll consider it successful. Another improvement from the midterm was that the format, when compared to the original document, wasn't heavily altered and remained true to the original PDF file's.

Maltz, M. (2015). Psycho-Cybernetics: Updated and Expanded. Penguin.