Emma Angela Montecchiari & **Juliette Napoli-Jacob**. Università di Trento. Course: Introduction to Computer Programming (Python).

Course Project Evaluation Submission. Summer Session 2022/23. 16-6-2023.

README file – **Text permutation over constrained computational algorithms**

The project deals with text manipulation over constrained computational algorithms. We are doing word replacement on specific word categories in both user-written texts and external data, mainly literacy material.

The idea of word replacement came from the Oulipo Surréaliste experimental group practice, which implemented techniques of constrained composition. They were using computational methods to build a structure in which they could convey their compositional skills and creativity. One of the most famous ones is the N + n technique, which we are actually implementing in this project.

Our aim, however, is not related to creating meaningful text and allowing the user to improve their compositional skills. Instead, we aim to provide a little interactive game through which one can observe a little text manipulation, through those techniques.

About

This project was designed for the class Introduction to Computer Programming, held at University of Trento and taught by Paolo Rota and Jakub Szymanik.
As part of the course evaluation, students were required to work on a Python project of their choice. Inspired by our shared interest in literature, we decided to focus on text processing and design a project involving text manipulation.

From this point, we decided to establish a program that would allow us and users to play with texts and words. We wanted to implement basic text manipulation, such as word replacement. To achieve this, we implemented two primary methods of text manipulation: Noun Switch and the Oulipo technique.

Description

Noun Switch

The Noun Switch is the first implemented word replacement feature in this project. It utilizes pre-built libraries, such as Spacy, to replace all the nouns in a given text (Text A) with the nouns from another text (Text B). The selection of Text A and Text B depends on user input, which can be either a custom text or chosen from a list of classic literature texts. The resulting output is a new text (Text C) with the replaced nouns.

The operation follows a 1:1, 2:2, 3:3 switching algorithm. More specifically we achieve this through extraction and list appending methods. We are using the part-of-speech tagging and tokenization from Spacy.

Example:

- Original text (Text A): We saw a wonderful cat outside of the window, and now my children want to take care of it and of elephants, which we already have.
- Second text (Text B): We all want to go to the circus, since it's a beautiful show. But we just want to applaud the acrobats, not to see animals.
- Permuted text (Text C): We saw a wonderful circus outside of the show, and now my acrobats want to take care of it and of animals, which we already have.

N + n Oulipo technique

The N+n Oulipo technique is the second word replacement feature implemented in this project. As mentioned, Oulipo performed a writing style exploring the limits of composition and language through writing games, involving mathematical computations. In one of the famous technique games a word X (here N) in the text would be replaced by the nth one following it in the dictionary, hence the N+n formula.

In our implementation, we employ a mapping algorithm that takes user input to determine the text and the dictionary. We are using Spacy functionalities such as tokenization and part-of-speech tagging. When it comes to the dictionary, we downloaded the Brown corpus from the NLTK library which we filtered to a sorted list of nouns.

Example:

- Original text: I wish I could go home to see my cat
- Permuted text: I wish I could go homily to see my cataract.

Sentiment Analysis and Word Cloud

In addition to text manipulation, our project incorporates pre-trained machine learning features to perform further analysis on the generated texts. We have implemented Sentiment Analysis Polarity and Word Cloud displaying functionalities.
When it comes to sentiment analysis, we are using the NLTK TextBlob library. From the different features it offers as built on top of NLTK, we chose to focus on sentiment analysis. We are displaying the polarity of the original texts and the one created as positive (1), negative (-1) or neutral (0) through the pre-trained model. We are removing the stop words to diminish noise.

For the word cloud, the library is downloaded so the feature can be easily accessed through the code. We chose to remove stop words here again, as they do not bring any value to the result. We hoped that the most used words for each novel could give a vague representation of the theme, and such themes would be comically changed with the noun switching events.

Text Data and User Interaction

Data comes from Gutenberg project texts. Initially, we attempted to retrieve text data directly from the Gutenberg project using URL retrieval. However, due to access limitations of Italian VPNs, we decided to download the texts locally and store them in a folder within the project repository for safety.
To ensure clean and usable text data, we perform data cleaning operations. They are cleaned up from the footer, through regex expression finding. From all the first part until the very beginning of the actual texts (getting rid of prefaces and indices) through detection of manually defined starting points.
After that, the program interacts with the files locally through os features.

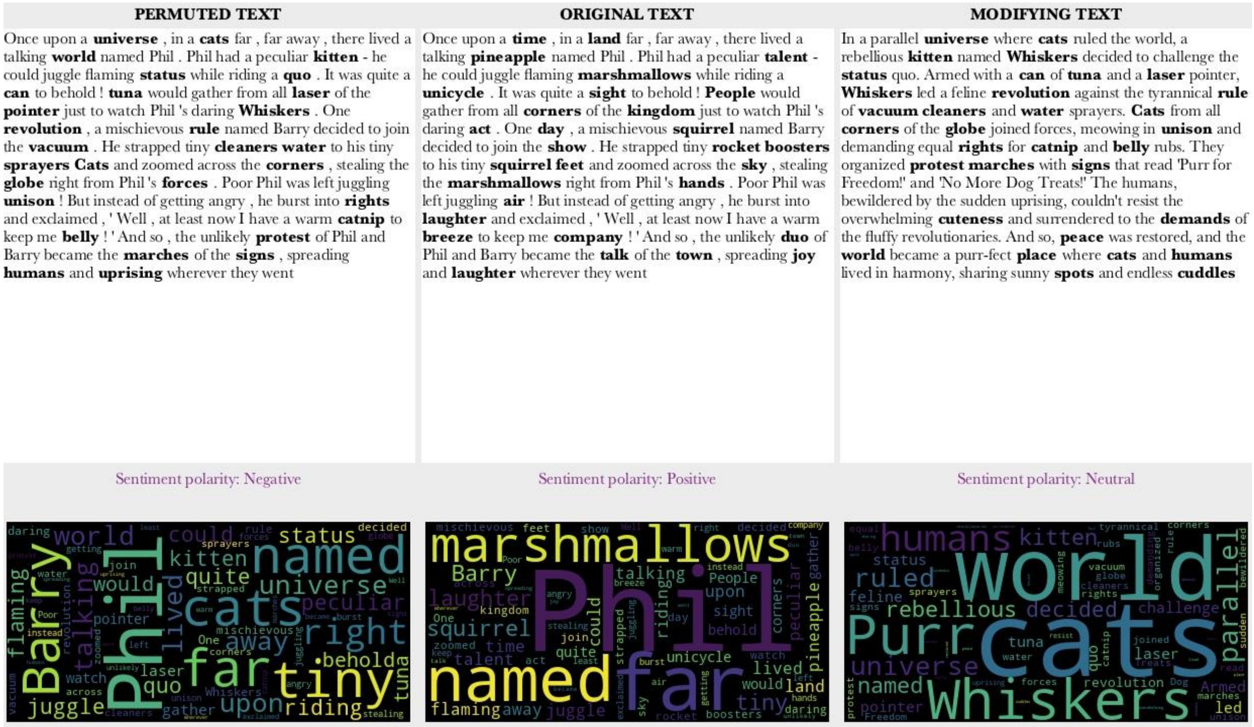
The program provides a simple input-output interface for user interaction. The user can choose between different text inputs and techniques. Currently, the program is written only for English language usage. The input-output is allowed to make selections and observe the manipulated texts.

Results displaying -
Terminal printing and GUI graphical interface

The results are both displayed in a GUI graphical interface and on the terminal.
The GUI graphical interface is implemented using the TKinter library, which allows for the display of text and image frames. The interface consists of three frames for the nouns switch technique: Original, Second, and Permuted texts. For the N+n technique, there are two frames to avoid displaying the dictionary. The texts have the nouns in bold and can be scrolled through with mouse commands. Additionally, under the texts, the sentiment polarity and corresponding word clouds for each text are displayed.

The GUI might require some time to load, especially for longer texts, such as the ones from classic literature. Therefore, for safety and practicality, we also made a terminal printing option. This option prints the texts (two or three depending on the technique used) for their first 1500 characters, with the nouns highlighted in bold.

***Gui displaying example*:**



PERMUTED TEXT

Once upon a **universe** , in a **cats** far , far away , there lived a talking **world** named Phil . Phil had a peculiar **kitten** - he could juggle flaming **status** while riding a **quo** . It was quite a **can** to behold ! **tuna** would gather from all **laser** of the **pointer** just to watch Phil 's daring **Whiskers** . One **revolution** , a mischievous **rule** named Barry decided to join the **vacuum** . He strapped tiny **cleaners water** to his tiny **sprayers Cats** and zoomed across the **corners** , stealing the **globe** right from Phil 's **forces** . Poor Phil was left juggling **unison** ! But instead of getting angry , he burst into **rights** and exclaimed , ' Well , at least now I have a warm **catnip** to keep me **belly** ! ' And so , the unlikely **protest** of Phil and Barry became the **marches** of the **signs** , spreading **humans** and **uprising** wherever they went

ORIGINAL TEXT

Once upon a **time** , in a **land** far , far away , there lived a talking **pineapple** named Phil . Phil had a peculiar **talent** - he could juggle flaming **marshmallows** while riding a **unicycle** . It was quite a **sight** to behold ! **People** would gather from all **corners** of the **kingdom** just to watch Phil 's daring **act** . One **day** , a mischievous **squirrel** named Barry decided to join the **show** . He strapped tiny **rocket boosters** to his tiny **squirrel feet** and zoomed across the **sky** , stealing the **marshmallows** right from Phil 's **hands** . Poor Phil was left juggling **air** ! But instead of getting angry , he burst into **laughter** and exclaimed , ' Well , at least now I have a warm **breeze** to keep me **company** ! ' And so , the unlikely **duo** of Phil and Barry became the **talk** of the **town** , spreading **joy** and **laughter** wherever they went

MODIFYING TEXT

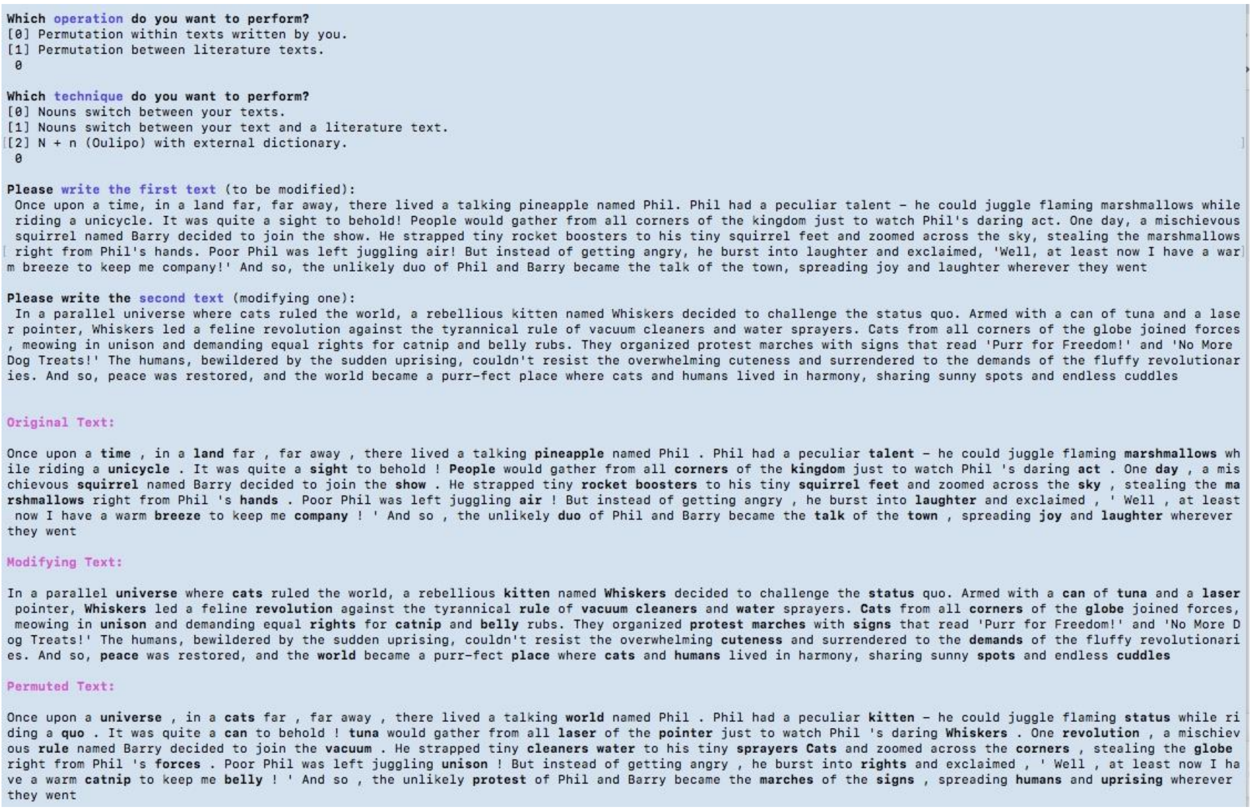
In a parallel **universe** where **cats** ruled the world, a rebellious **kitten** named **Whiskers** decided to challenge the **status** quo. Armed with a **can** of **tuna** and a **laser** pointer, **Whiskers** led a feline **revolution** against the tyrannical **rule** of **vacuum cleaners** and **water** sprayers. **Cats** from all **corners** of the **globe** joined forces, meowing in **unison** and demanding equal **rights** for **catnip** and **belly** rubs. They organized **protest marches** with **signs** that read 'Purr for Freedom!' and 'No More Dog Treats!' The humans, bewildered by the sudden uprising, couldn't resist the overwhelming **cuteness** and surrendered to the **demands** of the fluffy revolutionaries. And so, **peace** was restored, and the **world** became a purr-fect **place** where **cats** and **humans** lived in harmony, sharing sunny **spots** and endless **cuddles**

Sentiment polarity: Negative

Sentiment polarity: Positive

Sentiment polarity: Neutral

Terminal outputting example:



```
Which operation do you want to perform?
[0] Permutation within texts written by you.
[1] Permutation between literature texts.
 0

Which technique do you want to perform?
[0] Nouns switch between your texts.
[1] Nouns switch between your text and a literature text.
[2] N + n (Oulipo) with external dictionary.
 0

Please write the first text (to be modified):
 Once upon a time, in a land far, far away, there lived a talking pineapple named Phil. Phil had a peculiar talent - he could juggle flaming marshmallows while
 riding a unicycle. It was quite a sight to behold! People would gather from all corners of the kingdom just to watch Phil's daring act. One day, a mischievous
 squirrel named Barry decided to join the show. He strapped tiny rocket boosters to his tiny squirrel feet and zoomed across the sky, stealing the marshmallows
 right from Phil's hands. Poor Phil was left juggling air! But instead of getting angry, he burst into laughter and exclaimed, 'Well, at least now I have a war
m breeze to keep me company!' And so, the unlikely duo of Phil and Barry became the talk of the town, spreading joy and laughter wherever they went

Please write the second text (modifying one):
 In a parallel universe where cats ruled the world, a rebellious kitten named Whiskers decided to challenge the status quo. Armed with a can of tuna and a lase
r pointer, Whiskers led a feline revolution against the tyrannical rule of vacuum cleaners and water sprayers. Cats from all corners of the globe joined forces
, meowing in unison and demanding equal rights for catnip and belly rubs. They organized protest marches with signs that read 'Purr for Freedom!' and 'No More
Dog Treats!' The humans, bewildered by the sudden uprising, couldn't resist the overwhelming cuteness and surrendered to the demands of the fluffy revolutionar
ies. And so, peace was restored, and the world became a purr-fect place where cats and humans lived in harmony, sharing sunny spots and endless cuddles


Original Text:

Once upon a time , in a land far , far away , there lived a talking pineapple named Phil . Phil had a peculiar talent - he could juggle flaming marshmallows wh
ile riding a unicycle . It was quite a sight to behold ! People would gather from all corners of the kingdom just to watch Phil 's daring act . One day , a mis
chievous squirrel named Barry decided to join the show . He strapped tiny rocket boosters to his tiny squirrel feet and zoomed across the sky , stealing the ma
rshmallows right from Phil 's hands . Poor Phil was left juggling air ! But instead of getting angry , he burst into laughter and exclaimed , ' Well , at least
 now I have a warm breeze to keep me company ! ' And so , the unlikely duo of Phil and Barry became the talk of the town , spreading joy and laughter wherever
they went

Modifying Text:

In a parallel universe where cats ruled the world, a rebellious kitten named Whiskers decided to challenge the status quo. Armed with a can of tuna and a laser
 pointer, Whiskers led a feline revolution against the tyrannical rule of vacuum cleaners and water sprayers. Cats from all corners of the globe joined forces,
 meowing in unison and demanding equal rights for catnip and belly rubs. They organized protest marches with signs that read 'Purr for Freedom!' and 'No More D
og Treats!' The humans, bewildered by the sudden uprising, couldn't resist the overwhelming cuteness and surrendered to the demands of the fluffy revolutionari
es. And so, peace was restored, and the world became a purr-fect place where cats and humans lived in harmony, sharing sunny spots and endless cuddles

Permuted Text:

Once upon a universe , in a cats far , far away , there lived a talking world named Phil . Phil had a peculiar kitten - he could juggle flaming status while ri
ding a quo . It was quite a can to behold ! tuna would gather from all laser of the pointer just to watch Phil 's daring Whiskers . One revolution , a mischiev
ous rule named Barry decided to join the vacuum . He strapped tiny cleaners water to his tiny sprayers Cats and zoomed across the corners , stealing the globe
right from Phil 's forces . Poor Phil was left juggling unison ! But instead of getting angry , he burst into rights and exclaimed , ' Well , at least now I ha
ve a warm catnip to keep me belly ! ' And so , the unlikely protest of Phil and Barry became the marches of the signs , spreading humans and uprising wherever
they went
```

Requirements

We implemented it over a Python 3.11 version and with a Conda (23.1.0) environment.

The requirements are mainly over libraries installation:

1. os: Provides a way to interact with the operating system.
2. re: Offers support for regular expressions.
3. spacy: Natural language processing functionalities. (Version 3.5.0)
4. nltk: Natural language processing utilities and datasets. (Version 3.7)
5. textblob: Enables sentiment analysis (Version 0.17.1)
6. wordcloud: Used for generating word clouds. (Version 1.9.2)
7. colorama: Enables printing colored text in the terminal. (Version 0.4.6)
8. tkinter: Used for creating a graphical user interface (GUI). (Version 8.6.12)
9. PIL: Used to deal with image objects (Version 9.3.0)

We used pip to install the libraries which were not already inside the Conda environment. The Conda environment isn't strictly required but it's necessary to have all the libraries installed.

Regarding the additional resources, we are retrieving the 'brown' corpus and 'stopwords' from NLTK:

```
nltk.download('brown')
nltk.download('stopwords')
```

We create a set of English stopwords through the retrieval of the NLTK corpus:

```
stopwords = set(stopwords.words('english'))
```

To use the Spacy model, it's important to have 'en_core_web_sm' installed and downloaded. We specified it in the code:

```
from spacy.cli import download
spacy_model_name = 'en_core_web_sm'
if spacy_model_name not in spacy.util.get_installed_models():
   download(spacy_model_name)
```

However, if something it's not working, we would also suggest trying to run this in the terminal:

```
python -m spacy download en_core_web_md
```

Furthermore, we load the Spacy model with certain functions disabled to reduce computational load. We also adjust the tokenization maximum length for easier file retrieval:

```
nlp = spacy.load("en_core_web_md", disable=['ner', 'parser', 'textcat'])
nlp.max_length = 30000000
```

It's important to check if the text files folder is properly downloaded inside the project folder. In fact, we retrieve the texts locally using the path 'Gutenberg' stored in the project repository itself.

Warnings

Part of Speech Tag accuracy not at top-level:

Since part of speech tagging is better performing on big amounts of data, the nouns recognition might not always be accurate.

We tried to deal with this issue using Spacy, which was better on small texts than NLTK. We used the medium size Spacy corpus of English web data to train it, lighter than the large one and better performing than the small size.

Time Warning:

Tokenization is a computationally expensive process, and certain operations may require more time to execute.. To deal with this problem we have disabled some tokenization functionality which were not specific for our purposes: such as named entity recognition, dependency parsing and text categorization.

This helps improve the processing time. However, when working with larger texts, such as literature text data, the noun switch operation may still take longer. We kindly ask for your patience and understanding during these processing times.

GUI displaying:

Please note that the graphical user interface features implemented using Tkinter may introduce some performance overhead. While Tkinter provides certain functionalities that other libraries may not have, it is not the fastest library for GUI development. are heavy and the Tkinter library isn't the fastest one. To mitigate potential problems, we have also included output results in the terminal.

Future developments

The project might be improved with other functionalities and improvements of the existing ones:

- Tokenization accuracy improvements;
- Multilingual Support;
- Words category switching and a more complex system of user interaction. In general, more complex text analysis and the possibility to choose between dictionaries.
- Contextual and grammatical improvements. The switching operation might be adjusted on semantic context and grammar features, such as word gender and plurality. It could generally be improved to be transformed from a simple text manipulation game to a little composition machine.