Translator - Read Me

Install all the requirements using the below code

```
!pip install -r requirements.txt
```

Both CLI and Jupyter versions of the code are given. To run the CLI version follow the below mentioned approach.

```
python  \path_to_file\NLP_Project_Final.py
```

Introduction

The objective is to convert any words that are minimum in a given document. The words can either be American or British depending on the document.

A document can consist both American and British words. If the document were to be published as a journal or publication, this can issue. This is why the problem has to be solved.

So to solve this issue, One can use either Machine Learning (NLP) or Text processing or both. For this evaluation, I will be considering it as a Text Processing.

Explanation

First we need to consider that text processing will have an affect on the document. To treat this problem I leveraged docxtext and python-docx library.

To count the word we must have a list of British and American words. So, a custom word list is created. Then both the lists are compared with the document to get the count of British and American words in the document.

Any word that are matched from the lists must also have the original format.

Example: The word Aluminum can exist as aluminum, Aluminum and ALUMINUM.

So, Separate list are then created for each of these versions. Then converted back to a dict format and their consecutive words are kept.

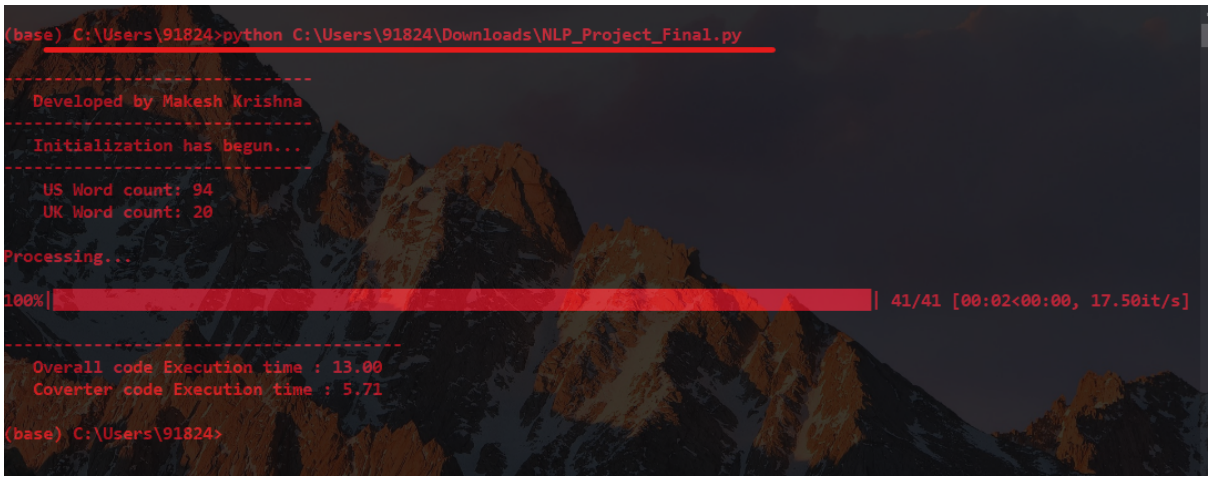
After getting the distribution for both American and British words. Based on the distribution the lesser value is then transformed.

Replacement for these words in a document while keeping the format same is an issue that can be solved using the above mentioned library. Words that are present in both paragraph and table can be changed with the help of these library.

Conclusion

Then converter function is used to convert the words with the help of dict values which has least count in the given document

Following image gives us how the code is executed in CLI .



Red Annotated line is how the execution is typed in

Validation

After manually checking. All but 2 words (Aluminum, Utilization) are not transformed properly.

Still needed to look in this issue

Names such as Washington and Molt (an author) are transformed

How can this be improved ?

1. This also can be treated as a Machine Learning (NLP) problem. Obviously a lot of data is required to transform British to American or vice-versa. Collecting both British and American word corpuses and then using the collected data to train a classifier to identify words based on Names, Places or Spelling.
2. As Docs for python does not meant to store a full Docx with Images, headers,..., but only contains the inner content of the document. So there's no simple way to do this.

   However, there are methods to do just this.

   All docx file can be unzipped. Here is how a typical file looks like:

```
+--docProps
|  +  app.xml
|  \  core.xml
+  res.log
+--word //this folder contains most of the files that control the content of the document
|  +  document.xml //Is the actual content of the document
|  +  endnotes.xml
|  +  fontTable.xml
|  +  footer1.xml //Containst the elements in the footer of the document
|  +  footnotes.xml
|  +--media //This folder contains all images embedded in the word
|  |  \  image1.jpeg
|  +  settings.xml
|  +  styles.xml
|  +  stylesWithEffects.xml
|  +--theme
|  |  \  theme1.xml
|  +  webSettings.xml
|  \--_rels
|     \  document.xml.rels //this document tells word where the images are situated
+  [Content_Types].xml
\--_rels
   \  .rels
```

   Recommendation :

   1. get the content of the document with **opendocx function*
   2. Replace the document.xml with the **advReplace** method
   3. Open the docx as a zip, and replace the document.xml content's by the new xml content.
   4. Close and output the zipped file (renaming it to output.docx)

3. We can also use the Text processing approaches mentioned above to create a Hybrid solution to transform Text.