LANGUAGE DETECTION AND TRANSLATION

Team Members:
19BCE2467 (VISHNU ANILKUMAR NAIR)
19BCE2495 (GLENN VARGHESE GEORGE)

Report submitted for the
Final Project Review of

Course Code: CSE4022 – Natural Language Processing

Slot: G2 + TG2

Professor: Dr. Jayashree J

Deadline: 30th May 2021

1. Introduction:

In our world, there are 7 billion people in different regions and they speak about 6500 languages. With these many languages, it will be hard for people to communicate with others from areas and cultures different from their own. Sure, the people from the same region won't have any problem communicating with each other but it's not the same case for people from different cultures and regions. Humans are very social animals. We crave to have a good community where all types of cultures can come together.

People can study new languages but it will take time and practice. But with so many languages present, it is almost impossible for everyone to learn it all. For this reason, an efficient way of translation was needed. English is the most prominent language in the world. As a result, one might question the importance of translation, and ask, why doesn't everybody just speak English?

The reality, however, is that not everybody can speak English, fewer still are able to speak it well enough to communicate effectively, and perhaps even more importantly: language is much more than the communication of words. It is also an expression of culture, society, and belief. Promoting a universal language, therefore, would likely lead to a loss of the culture and heritage communicated through native languages.

Translation is necessary for the spreading of new information, knowledge, and ideas across the world. It is absolutely necessary to achieve effective communication between different cultures. In the process of spreading new information, translation is something that can change history. It is the only medium by which certain people can know different works that will expand their knowledge of the world.

For this reason, we decided to build a language translator and also a detector with which the user can detect and translate their desired language. With our project, it can be implemented in areas like chat apps where two people can communicate with each other using their native languages. We can also use them for multiplayer games where players from different countries will be able to chat with each other using their native language for better strategic decisions on how to beat the other players in the game. Our project has a wide array of applications it can be embedded into.

2. Literature Review Summary Table

Authors and Year (Reference)	Title (Study)	Concept / Theoretical model/ Framework	Methodology used/ Implementation	Dataset details/ Analysis	Relevant Finding	Limitations / Future Research/ Gaps identified
Indhuja K, Indu M, Sreejith C April - 2014	Text Based Language Identification System for Indian Languages Following Devanagiri Script	To build a text based language identification system for Indian languages following Devanagari script.	Natural language Toolkit (NLTK) in Python language was used for this experimentation. Once the training set was created, the proposed system was used on random test data for classification and identification of unknown content in the digital online text.	For this, initially, separate corpora of approximately 2MB size were created for Hindi, Nepali, Bhojpuri, Marathi and Sanskrit languages.	It can be concluded that trichar, uniword and bichar identify the corresponding language with a greater accuracy around 90% in two pair LID.	It is observed that the differences between the languages within language family and across language family cases are not very drastic. So the prediction wasn't always accurate.
Apple Natural Language Processing team July 2019	Language Identification from Very Short Strings	A neural network-based LID system designed to improve classification when only a few characters have been observed.	They model LID as a character level sequence labeling problem.For each script, the architecture of Figure 1 maps character sub-sequences to languages based on global evidence gathered from a database of documents representative of all languages considered for that script. The recurrent neural network (RNN) paradigm naturally handles the problem of variable-length sequences	They design such an LID system for K=20 Latin script languages. The character inventory consists of around M=250 characters. To enhance discriminative power, they use a two-layer bi-directional variant of the LSTM architecture.	Robust LID from very short strings is highly desirable in multiple NLP pipelines, including in usage of autocorrection lexicons and predictive and multilingual typing, for part-of-speech and named entity tagging, when performing document classification], and as a component of TTS engines	They limited their experiments to relatively shallow LSTM networks rather than newer but computationally more complex architectures such as transformers.

Iwara ARIKPO, Iniobong DICKSON July-2018	Development of an automated English-to-local language translator using Natural Language Processing	The development of an automated English-to-local-language translator as a model to bridge the communication gap in Nigeria and other multilingual settings. Object-oriented methodology was used in the design and implementation of the machine translator. Java technology was used for the development of the software.	At the first layer of the architecture is a natural language processing tool that performs morphological analysis: sentence tokenization, part of-speech tagging, phrase formation and figure of speech tagging. The second layer of the architecture comprises of a grammar generator that is responsible for converting English language structure to target local language structures. At the third layer of the application, results produced by the grammar generator are mapped with matching terms in the bilingual dictionary.	A bilingual dictionary containing 500 words and 25 corpuses was designed to provide optimum direct translations regarding the Efik language.	The system provides unidirectional translations for sentences and text documents. Sentence translations are provided on text entry. Tests were performed on the system in two modes using 15 well-formed English simple and complex sentences.	They have only used Efik and English language, Hence high accuracy.
Ekansh Gupta Rohit Gupta October 2015	Hindi ↔ English Parallel Corpus Generation from Comparable Corpora for Neural Machine Translation	The aim of this project is to develop a technique to produce a high quality parallel sentence-aligned corpus from existing subject/document-aligned comparable corpora (Wikipedia)	The Keras Library was used to implement the RNN encoder-decoder translation model. The model encoded each input sentence (variable length) into a fixed-sized vector of length 512.	The vocabulary size used was 5000 for both Hindi and English.The translator was trained using 25000 sentence pairs on a GPU system with 4GB memory. The train-test split was 10%-90%.	The 600 article pairs were used to generate sentence pairs. About 1500 matching sentence pairs were generated after putting a reasonably high threshold in the aligner parameters for finding similarity.	They have only used Hindi and English language for translation. Pace of parallel corpus generation has been limited due to the slow pace of scraping.

Gerald R. Gendron, Jr. 2015	Natural Language Processing: A Model to Predict a Sequence of Words	This report provides documentation describing the process and decisions used to develop a predictive text model. The model uses natural language processing techniques to accomplish predictive analytics.	They cleaned and transformed corpus, it was tokenized using the tau package in the R programming environmentN-gram with a frequency less than (K:k=5) is an ideal candidate to adjust the count using Good-Turing methods. N-Grams appearing six or more times are seen to be realistic and do not require smoothing.	The dataset is a zip file including blog posts, news articles, and Twitter tweets in four languages (English, German, Finnish, and Russian). T	In this analysis, many observations were made of common relationships between the various sub-corpus elements, the vocabulary, and word types. These enabled better understanding of their impacts on predictive power of a model.	This measure was used to make a decision to limit the data for the model to just blog entries.

3. Objective of the project:

With this project we wanted to create a system to correctly predict and detect a language with high accuracy . Users can input words or even sentences, with the click of a button the language will be predicted and from the choice of the language they want to translate it into, they can find the word or sentence in the desired language.

4. Innovation component in the project:

We have been able to read, detect and translate languages from a dataset of 13328 entries and an accuracy of 90.435%. The dataset consists of l5+ languages including few Indian regional languages such as (Tamil, Malayalam, Kanada, Hindi) .

All of this has been made without the use of the NLTK library and with the use of the MultiNomialNB model.

5. Work done and implementation

Methodology adapted:

This program is constructed with the use of

- **CountVectorizer** tool from the scikit-learn library to break down the given text into words and then modify it into vectors on the basis of its frequency.
- **LabelEncoder** tool from the sklearn library is used to encode the levels of categorical features into numeric values i.e. in this project to give unique records in the language column and count of its corresponding records from the text column.
- **train_test_split** tool from the sklearn library split arrays or matrices into random train and test subsets
- **test_size** has been initialized as 0.20 (size of the test set is 20% and the train set is 80% of the Lang_new_version_2.0 dataset)
- The model chosen for this project is **MultiNomialNB** from the sklearn.naive_bayes library ,The multinomial Naive Bayes classifier is suitable for classification with discrete features (e.g., word counts for text classification) which makes it an apt combination with the CountVectorizer tool.
- The accuracy score, confusion matrix and classification report has been classified from the sklearn.metrics library

Hardware requirements:

Windows 10 , 64-bit , 16 GB RAM / MacOS X

Software requirements:

Anaconda Navigator Jupyter Notebook, pip 20.2.4, python 3.8

Dataset used:

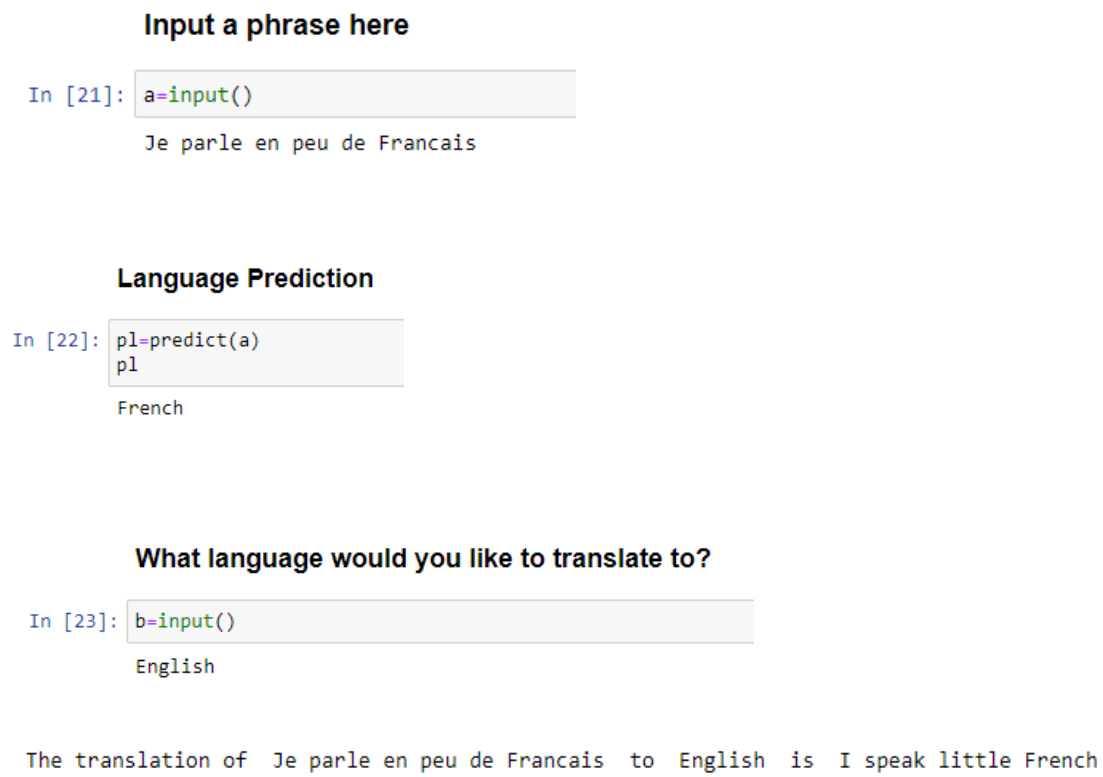
The dataset used for this project Lang_new_version_2.0, comprises **15+** languages and comprises **13000+** entries is partly unique and partly used from various sites such as kaggle and lexicalcomputing.com .

Tools used

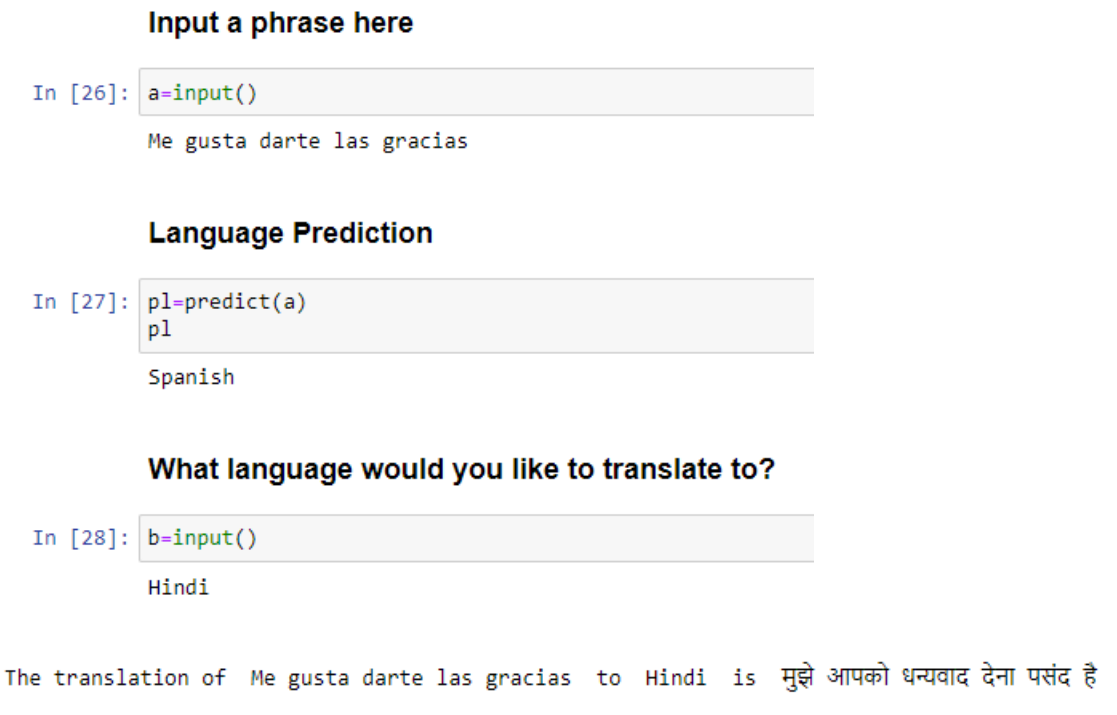
- Jupyter Notebook
- Excel for the dataset
- pandas for data manipulation and analysis.
- numpy for adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.
- re for using regular expression functions
- seaborn and matplotlib for drawing attractive and informative statistical graphics.
- deep_translator to import the google_translator api for the translation process

Screenshot and Demo:

Predicting French and Translating it into English.



Predicting Spanish and translating it into Hindi.



6. Results and discussion

We have 17 languages and their corresponding unique words are shown below.

Counting the unique records in the "Language" coloumn

```
In [6]: df["Language"].value_counts()
```

```
Out[6]: English       1884
        French        1516
        Spanish       1319
        Italian       1198
        German         971
        Portugeese     739
        Russian        692
        Sweedish       676
        Malayalam      594
        Hindi          552
        Dutch          546
        Arabic         536
        Turkish        474
        Tamil          469
        Danish         428
        Kannada        369
        Greek          365
        Name: Language, dtype: int64
```

The confusion matrix displayed here gives information on how the languages are being compared and the similarity of one language with another .

example: the characters in the english language have most similarity with english as compared to any other language followed by Italian and portugeese

Confusion Matrix

```
In [16]: cm = confusion_matrix(y_test, y_pred)
         print ("Confusion Matrix : \n\n",cm)
```

```
Confusion Matrix :

 [[113   0   0   1   0   1   0   0   0   0   0   0   0   0   0   0   0]
 [  0  69   0   3   0   0   0   0   1   0   0   0   0   0   2   0   0]
 [  0   0 104   2   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0 386   3   4   0   0   4   0   0   0   0   2   0   0   0]
 [  0   0   1  17 271   0   0   0   0   0   0   1   0   4   0   0   0]
 [  0   0   5  28   9 138   0   0   2   0   0   0   0   3   3   0   0]
 [  0   0   0   2   0   1  71   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0  59   0   0   0  60   0   0   0   0   0   0   0   0   0]
 [  0   0   0  24   4   0   0   0 209   0   0   2   0  11   0   0   0]
 [  0   0   0   3   0   1   0   0   0  81   0   0   0   0   0   0   0]
 [  0   0   0   3   0   0   0   0   0   0 103   0   0   0   0   0   0]
 [  0   0   0   1   1   0   0   0   0   0   0 138   0   3   0   0   0]
 [  0   0   0   1   0   1   0   0   0   0   0   0 128   0   0   0   0]
 [  0   0   0  20   6   1   0   0   3   0   0   5   0 206   0   0   0]
 [  0   1   0   2   0   0   0   0   0   0   0   0   0   0 152   0   0]
 [  0   0   0   1   0   0   0   0   0   0   0   0   0   0   0  89   0]
 [  0   0   0   3   1   4   0   0   0   0   0   1   0   0   0   0  87]]
```

The accuracy found out while predicting the languages is found to be 90.435108%

Accuracy Score

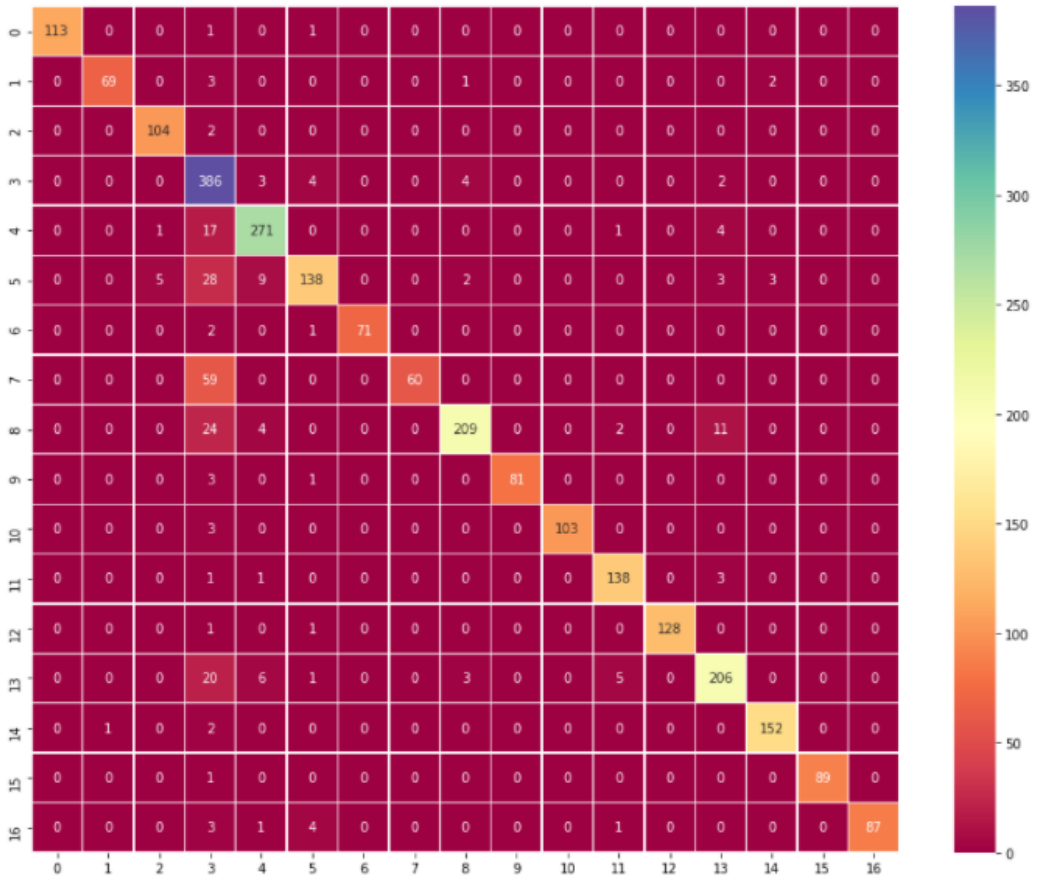
```
In [16]: ac = accuracy_score(y_test, y_pred)
         print("The accuracy comes to : ",ac)

         The accuracy comes to :  0.904351087771943
```

Here the confusion matrix has been plotted onto a heat map where the spectral colourmap has been used, so the block having the highest similarity to a language has been marked a shade of blue ((3,3),386) and the least being marked with a shade of dark red. (0)

Plotting the confusion matrix onto a heatmap

```
In [18]: plt.figure(figsize=(15,12))
         sns.heatmap(cm, annot = True,fmt="d",cmap="Spectral", linewidths=.3 )
         plt.show()
```



Here the classification report has been displayed from where we can learn the precision , recall, f1-score and its corresponding macro average ,weighted average and support value & and the total accuracy.
we understand from this that 20% of the dataset consisting of 13328 records = 2666 has been taken as the entries in the test set

Precision= $\frac{TP}{(TP+FP)}$.

Recall = $\frac{TP}{TP+FN}$

f1-score = $2 \cdot \frac{precision \cdot recall}{precision + recall}$

TP - True Positive **, FP -** False Positive **, FN -** False Negative

Classification Report

```
In [17]: cr = classification_report(y_test, y_pred)
         print ("Classification Report : \n\n",cr)
```

```
Classification Report :

              precision    recall  f1-score   support

           0       1.00      0.98      0.99       115
           1       0.99      0.92      0.95        75
           2       0.95      0.98      0.96       106
           3       0.69      0.97      0.81       399
           4       0.92      0.92      0.92       294
           5       0.91      0.73      0.81       188
           6       1.00      0.96      0.98        74
           7       1.00      0.50      0.67       119
           8       0.95      0.84      0.89       250
           9       1.00      0.95      0.98        85
          10       1.00      0.97      0.99       106
          11       0.94      0.97      0.95       143
          12       1.00      0.98      0.99       130
          13       0.90      0.85      0.88       241
          14       0.97      0.98      0.97       155
          15       1.00      0.99      0.99        90
          16       1.00      0.91      0.95        96

    accuracy                           0.90      2666
   macro avg       0.95      0.91      0.92      2666
weighted avg       0.92      0.90      0.90      2666
```

From the Demo shown, we have correctly predicted the language and translated it to the language we wanted.

7. References

- ➢ Indhuja, K., Indu, M., Sreejith, C., Sreekrishnapuram, P., & Raj, P. R. (2014). Text based language identification system for indian languages following devanagari script. International Journal of Engineering, 3(4).
- ➢ https://machinelearning.apple.com/research/language-identification-from-very-short-strings
- ➢ ARIKPO, I., & DICKSON, I. Development of an automated English-to-local-language translator using Natural Language Processing.
- ➢ https://www.cse.iitk.ac.in/users/cs671/2015/_submissions/egupta/project/report.pdf

➢ Gendron, G. R., & Gendron, G. R. (2015). Natural Language Processing: A Model to Predict a Sequence of Words. *MODSIM World*, *2015*, 1.
➢ Stackoverflow.com
➢ medium.com
➢ towardsdatascience.com