**List of files**

***Text\_to\_json.ipynb***

This file create a corpus from a plain text I need two file one for Amharic and one for English it should be equal number of line.

***IBM1.ipynb***

This file create the statistical table and maximum probability. This file produce two file AbebeBesoBela max probability.json( translated word with max probability) and AbebeBesoBela statisticas overall.json (statistical probability table)

***FinalSearch.ipynb***

This file used to translate the user input Amharic text to English with AbebeBesoBela statisticas overall.json and return translated value

***FinalSearch-On overall.ipynb***

It translate from max probability file which is AbebeBesoBela max probability.json

**am\_tr\_pair\_small.json**

bilingual corpus

**AbebeBesoBela max probability.json**

Source language with target language with max probability

**AbebeBesoBela statistics overall.json**

Source language with target language with all word by word probability

**Programing language**

python

**Software requirements**

**Oprating system :-** Window 11

Environment :-Anaconda

Development tool:- Jupiter lab version 3.2.1 and VS code

**Package and library requirements**

Nltk #language module

Difflib # sequence matching

Defaultdict # object creation

Re # to use regular expressions

Codecs # to support non English text using asci and unicode

**System work flow**

1. I need Bi lingual source and target corpus pre processed and saved on the same file system. the corpus need to save as a Jason file
2. I have a text to Jason converter that automatically reads two files which is the source text and target text then it's much line by line and produce Jason file that can used as a corpus.
3. the system read the bilingual corpus and for each words from the source language run the probability against the target language each word is from source language to each word this from the target language. Here I exclude English prepositional words like “the,and,a,be,is,an……
4. It run the maximum probability after the initial statistical calculation executed
5. what's the statistical probability table and the maximum probability result he served as a surprise to some fire that help us to evaluate those two outputs
6. I have two searching algorithms Ill for statistical probability and other for maximum probability
7. a client give a source language which is an American text on the search algorithm and the search algorithm search from the maximum probability if I are run the maximum probability algorithm if I run the other algorithm so that the algorithm run on the statistical probability and try to get the maximum probability value as a final result

**Function list and there purpose**

**IBM1.ipynb**

**variable declaration**

source language = ar

translation language=tr

Iteration number=2

English prepositions[“”]

file=”corpus.json”

**Function list**

**def** fileOpener(FILEZ): this function used to open the given file and return the Jason data

**def** word\_tag(corpus): this function is the corpus as an inputs and it's split every words put it onto the corpus object array

**def** ibm1\_EM(word\_tags): it iteratively assign every word with 0 probability then with other iteration every source word assign with target word every source word with target word. One loop run to calculate p(e|f) other loop for every sentence p(f) finally it assign probability value for every word return with object.

**def** maximum\_prob(current\_prob): I calculate the maximum probability value to assign for the source word.

**def** alignment(prob\_result, word\_tags): it try to produce a word alignment (at this time I didn’t use it but I attempted to local alignment which indicate which word come after which word probability)

**def** JsonSaver(jsonName,Content): this used to save the probability table

**def** amharic\_english(): to compile all fragmented function and give a single run up

**def** main(): to initiate execution

**FinalSearch.ipynb**

**Variable declaration**

FILE=”” #corpus

**Function list**

**def** fileOpener(): it open the corpus file

**def** sortmeout(e): it use the corpus and check the maximum probability and return the word with high probability value

**def** charSequencSimilarity(str1,str2): checking the word similarity

**def** searchMe(sw): searching every input word and return the result

**def** breakDown(zi): it broke down the user input and get a translation for every word

**def** alighneLanguageModel(en,am):it get the language model and solve the subject object relation for the language( I give a simple rule manually ) in the real life scenario this information get from local alignment function

**How the system run**

Our system calculate a word level statistical probability unlike the phrase level statistical machine translation word level translation will be more suitable for Amharic because Amharic has a high level of inflation on word level so that its better to calculate on word level with the expense of computational cost.

For simplicity I advice to use a simple sentence for our corpus in our test I use a small amount of corpus but in the real life scenario for better result its expected to feed a large corpus.

In our system creating translation table consume a lot of computational resources and time to reduce this problem I propose other alternative algorithms with first extract similar words from source text which is Amharic text and collect translated language together and execute statistical probability which reduce time end resource.

Our model produce two kind of table one a translation table with all probabilities list and the other with maximum probability.

I have two searching tool one for Max probability and the other for all possible table