The code implementation is based on the work done by William Cavnar and John Trenkle("N-Gram-Based Text Categorization",1994). The dataset was obtained from https://tatoeba.org/. 30 languages were identified as having sufficient data and a maximum of 25000 sentences were chosen from the dataset for each language to train the ngram-language model.

For each language, the corresponding sentences were cleaned and parsed to obtain only the letters,special symbols and apostrophes. Punctuation marks and other symbols were discarded or replaced using the parser. Each word was broken down into constituent bigrams and trigrams and their frequency count was updated. Finally, the top 20000 most occurring ngrams were taken and stored in a dictionary corresponding to the language by rank. Rank here indicates the number of occurrences of each ngram from maximum to minimum.The dictionary generated was stored in a pickle file for further use in any program. All of this is computed in preprocess.py in src folder.

The test1 directory files were generated by taking 50 sentences from each language and appending them together to construct a document like file. The test2 directory files consist of only 6 languages, but are comparable to the size of a normal document. The test2 directory files were extracted from an nltk corpus called "genesis".

Similar approach was used as mentioned above on each test1 and test2 file to break the words of each language into bigrams and trigrams in lang_identify.py. Top 20000 most occurring ngrams were taken and ranked. Each ngram of the test files was then compared with the corresponding ngram in the stored pickle file for all languages to check for similarity in ranking between the two. Absolute difference between the rank of the two ngrams is then added to a distance variable. If the ngram does not exist in the pickle file, then a large penalty is added to the distance variable. The language with the least distance value is predicted as the language the test document is written in.