LANGUAGE IDENTIFICATION OF TEXT

Project report submitted in partial fulfilment of the requirement for the degree of Bachelor of Technology

in

Computer Science and Engineering

By

Aishwary (131316)

Under the supervision of

Ms. Ruhi Mahajan

to

Department of Computer Science & Engineering and Information Technology
Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

Certificate

Candidate's Declaration:

I hereby declare that the work presented in this report entitled " Language Detection of Text" in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat is an authentic record of my own work carried out over a period from July 2016 to May 2017 under the supervision of **Ms. Ruhi Mahajan**, Assistant Professor, Dept. of Computer Science & Engineering, Jaypee University of Information Technology, Waknaghat.

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Aishwary
131316

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Ms. Ruhi Mahajan
Assistant Professor
Dept. of Computer Science & Engineering
Dated:

Acknowledgement

I would like to take the opportunity to thank and express my deep sense of gratitude to my mentor and project guide **Ms. Ruhi Mahajan** for her immense support and valuable guidance without which it would not have been possible to complete my final year project.

I am also obliged to all my faculty members for their valuable support in their respective fields which helped me in completing my project.

TABLE OF CONTENTS

List of Figures:

List of Tables:

List of Graphs:

Abstract

Language Identification refers to the process of detecting the language(s) of the text in the document based on the script used for writing and observing the diacritics particular to a language. This research area has always fascinated researchers as early as 1970 and till now due to varied applications and increased demands of this field. In this work, I address the problem of detecting language of textual documents. I have introduced a method which is able to detect language of text more efficiently and accurately by determining their respective proportions and finding the greatest of them which represents the language of the text. I have demonstrated the performance comparison of three different approaches which are using n-gram approach (word-wise), using n-gram approach (character-wise) and using a combination of word search and stop words detection. My project currently contains language models for 4 languages. On an average the accuracy of my program is about 96.5%.

CHAPTER-1

INTRODUCTION

1.1 Introduction

Language**,** a system of conventional spoken, manual, or written symbols by means of which human beings, as members of a social group and participants in its culture, express themselves. The functions of language include communication, the expression of identity, play, imaginative expression, and emotional release.

Every physiologically and mentally typical person acquires in childhood the ability to make use of a system of communication that comprises a circumscribed set of symbols (e.g., sounds, gestures, or written or typed characters) in both way i.e., as a sender or receiver. By means of these symbols, people are able to impart information, to express feelings and emotions, to influence the activities of others, and to comport themselves with varying degrees of friendliness or hostility toward persons who make use of substantially the same set of symbols.

Different systems of communication constitute different languages; the degree of difference needed to establish a different language cannot be stated exactly. Generally, systems of communication are recognized as different languages if they cannot be understood without specific learning by both parties, though the precise limits of mutual intelligibility are hard to draw and belong on a scale rather than on either side of a definite dividing line. So, here comes the role of Language Translation which implies the role and importance of Language Identification. Language Identification typically acts as a pre-processing stage for both human listeners (i.e. call routing to a proper human operator) and machine systems (i.e. multilingual speech processing systems).

Language Identification or Language Guessing is studied under Natural Language Processing and is the problem of determining which natural language given content is written in. Computational approaches to this problem view it as a special case of text categorization, solved with various statistical methods.

1.2 Problem Statement

People can communicate in over 6900 languages. Information on the internet may seem limitless. However, a person who speaks one language can access only a fraction of the available information. For example, the website Wikipedia contains articles in 262 languages, and as of July 2007, only 23% of the articles were in English. World Press is a website that describes itself as “a nonpartisan magazine whose mission is to foster the international exchange of perspectives and information”. It contains approximately seven million blogs, and only around 36% are in English. Throughout history, Electrical and Computer engineers have built tools, including the telegraph, telephone, and internet router, which have helped people communicate. Computer software which can translate between languages is one such tool. The first step of translating a text is to identify its language. In this project, we are writing a computer program, a tool, which accomplishes this first step.

1.3 Objectives

The primary goal of the project was to accumulate the necessary dictionary data or corpus for comparisons and matching. The objective of this research was to design, develop and demonstrate a system for Language Identification of a text written in selected languages so as to identify and differentiate in the languages in same text. This involves identifying the text in case of single language and the dominant language in case of multiple languages. The main objectives of this project can be summarised as follows:

- Collect the required and necessary research papers.
- Study thoroughly the research papers for available research and study in the topic and various algorithms employed for the implementation
- Select one of the algorithms to form a basic model
- Perform necessary improvements so as to improve the efficiency and working of algorithm
- Collect and identify various dictionary data and corpus for various languages
- Interface for human-computer interaction

1.4 Methodology

Language Identification can be done using two types of techniques:

i. Computational techniques
ii. Non-computational techniques.

Computational techniques is a statistical technique and a large dataset is required to train for each language. Statistical technique is segregated into two steps:

i. Training part
ii. Classification part.

In the training part, the feature extraction from the given training dataset, known as corpus, is done. In the classification part, the similarity measure between the training profile and the testing profile is found out and the most similar language is known as the language of the document. Non-computational techniques require that researcher must have extensive knowledge about the language to-be-identified such as diacritics and symbols, most frequent words used, character combinations etc. The approach we have opted for is Computational technique.

Different researchers have focused on different aspects of language identification such as identification of monolingual documents, multilingual documents, short documents, long documents, search engine queries, microblog posts, printed documents etc. But my main focus was on texts with monolingual context also computing the proportion of various languages available to achieve the task.

1.5 Organization

This project report will include a concise and thorough content on the various research done in order to understand the various algorithms in the topic. Moreover it will also include the various steps taken in order to attain the completion of projects. Following is the series in which the various research and steps are taken:

- Chapter-1 deals with the brief overview of the topic opted. It gives an introduction to the topic. Then it moves to the problem statement of the project along with the objectives that are needed to be attained. Then it moves on to the methodology that needs to be opted. And finally, it discusses about how the content is organised in rest of the report.
- Chapter-2 discusses the summarised overview of the various research papers taken into consideration along with important facts and figures completed with the required references utilised in those research papers
- Chapter-3 discusses the system development. In it various tools required are discussed along with their brief details. Further the development model is discussed which is to be employed during the development of project.
- Chapter-4 gives the comparison and contrast between different methods used till the current date and there performance comparisons on various grounds.
- Chapter-5 gives the conclusion that has been derived from the research and results obtained along with the various applications of my project.

CHAPTER-2
LITERATURE SURVEY

[1] An N-Gram-and-Wikipedia Joint Approach to Natural Language Identification, by Xi Yang, Wenxin Liang

The main contributions of their work are summarized as follows:

- Effective identification of multiple languages present in multilingual document based on their proposed approach.
- Their method can detect upto 250 languages greatly exceeding the milestone set by current practices in this area.
- Their method employs only one single language corpus. Thereby completely avoiding the need to collect multiple corpora for different languages
- They were the first researchers to bring in Wikipedia as a huge corpus in field of Natural Language Identification.

The core of the proposed N-Gram-and-Wikipedia joint approach can be divided into two distinct steps:

(1) Dividing the text into separate units based on the language they are written in using the segmentation algorithm which is based on concept of N-Gram

(2) Language of each unit is to be identified using the multiple language versions of Wikipedia articles.

The text is divided into separate units by calculating similarity of each unit with the local English corpus based on N-gram frequency statistics, and then segregating each unit accordingly.

They particularly used local English corpus as a base for the calculation of distance between every to-be-identified paragraph. And, thereby the degree of language difference between two adjacent paragraphs is calculated by finding their absolute subtraction of distance.

Larger the difference indicated the higher probability that the two are in different languages. In short, the English corpus itself was not employed to identify languages, but was thought of as a benchmark to measure and quantify the language properties of paragraphs, making mathematical comparison and the subsequent language identification between adjacent paragraphs possible.

After it, from each unit a segment of words are obtained to be used as a search keyword for domain Wikipedia.org using Google. Regularity in Wikipedia URLs made it possible to know the language used in that specific page. Hence, by studying the URL obtained language of each keyword is determined leading to determination of original text.

Their proposed approach is shown through figure underneath:

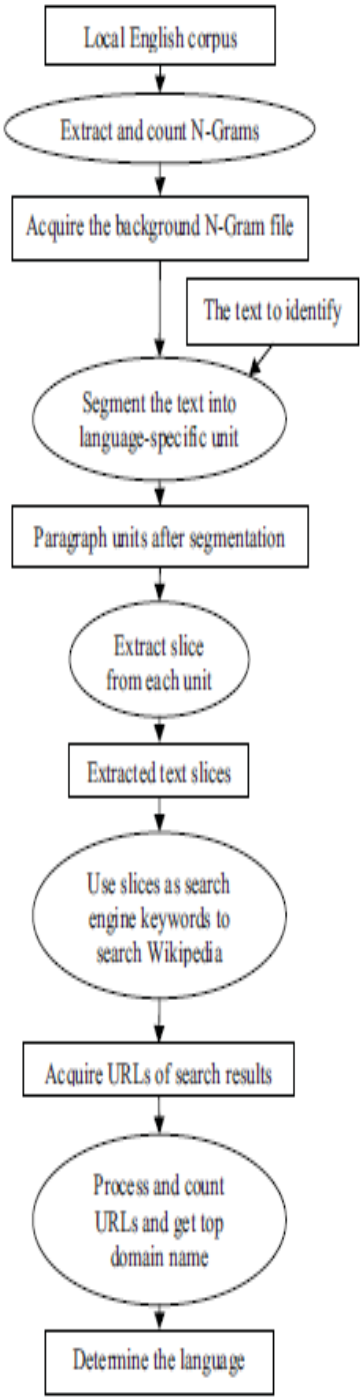


Figure 1. Data Flow of the Proposed Approach

The corpus is needed to undergo the pre-processing procedures before being further employed to remove unnecessary text information which includes special characters, punctuations, numbers and unnecessary spaces. Leaving behind a corpus comprised only of continuous English characters. Hence, this step made the required preparation to accurately extract N-gram in next step.

They extracted the most frequently occurring N-grams ranging from N = 1 to 5, so as to obtain intrinsic characteristics of the language of text and hence computing distance by comparing the two texts using most frequent N-Grams occurring in both texts'.

It is not easy to judge the sufficient number of N-grams required to be extracted from a piece of text but how to acquire N-gram is outlined here briefly:
Firstly, whole text is needed to be traversed to extract N-grams (character-wise) ranging from N=1 to 5, and then Hash tables are used to record the N-grams and corresponding counters. Counter within the Hash tables are incremented, when put into N-grams of same value. Hence, accurate track is maintained of N-gram count, occurring within the local corpus.
Subsequently, the N-grams are sorted in descending order by the counter value, and the rankings are allocated in place of counter value.

These steps helps in establishing an N-gram frequency statistics for the local English Corpus. And this data is used as a benchmark to judge the text to be identified for similarity discrepancies.

Statistical techniques are employed for segmentation into language-specific text and uses N-gram to divide text into language specific units.
Firstly, distance is calculated between local English corpus and each para in the text to be identified by adding up the absolute differences of the two ranking of corresponding n-grams in N-gram model of English corpus and text.

Now it is known that the two neighbouring paras are in two different languages, but their corresponding languages are not known. And here comes the role of Wikipedia and Google for the determination of their corresponding languages.

Now, from every paragraph a slice of seven words is extracted randomly and is searched in Wikipedia.org domain being used as a Google search keyword.

The URL can be constructed as follows:
"http://www.google.com/search?q="+ string + "site%3Awikipedia.org&ie=utf 8&aq=t&rls=org.mozilla:en- THEY:official&client=firefox-a&num=100"

In this URL "string" refers to the slice being used as search keyword. This URL will return many search results which will be further analysed and judged based on unique code present in every Wikipedia article's URL pointing to the language it is written in.

For example, the code "de" in the following Wikipedia URL:
"http://de.wikipedia.org/wiki/Heinrich_Heine"
shows that this is an article written in German.

Hence, they can extract the required code from the most frequently occurring Wikipedia link and accordingly judge the language of the paragraph.

]2] Comparing methods for language identification,
by Muntsa Padró and Llúıs Padró

This paper talks about and compares three different statistical techniques of Language Identification and studies their influence on some basic parameters. Parameters which are analysed includes the amount of text to be classified, optimum size of the training dataset and the languages that can be distinguished by the system.

They employed all three statistical technique on same data for training and testing. These techniques were based on:

- Markov Models
- n-gram text categorisation
- comparison of Trigram Frequency Vectors

All the above techniques are briefly introduced ahead. Figure 1 shows general architecture of systems of all techniques being used. All the techniques that are applied are statistical and a set of predetermined language is required to work with them. When the system is trained for one language, the information stored in it is in a particular way. Each system creates a statistical model of each language it has been trained for. During classification, unknown text is compared with every language model and a similarity measure is computed to obtain a closest language, which is termed as the language of the text. This language modelling and the similarity measure is the point where these techniques differ from one another.

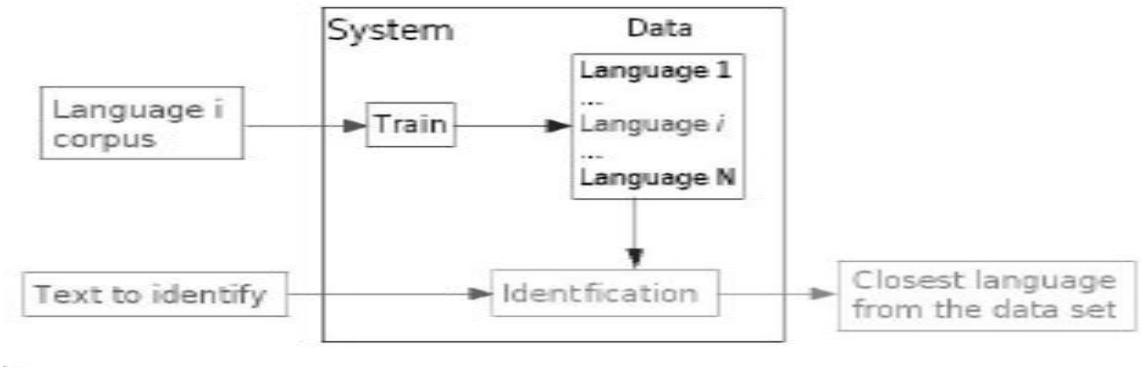


Figure 2: General architecture of the three systems used

For identification of spoken languages, Hidden Markov Models (HMM) are generally employed (Zissman and Singer, 1994; Lamel and Gauvain, 1994) and can also be used in identification of written text (Xafopouloset al., 2004; Ueda and Nakagawa, 1990). Regardless of that this task can also be accomplished using first of the three techniques i.e, visible Markov Model (MM).

In second method employed i.e., the Trigram Frequency Vector Technique (as stated by Damashek, 1995) vectors of known languages are used for classification of text after comparing the vectors of trigram frequencies, and the closest one is selected as the language of the text.

While training, for each trigram their relative frequencies are computed for a particular language, which are further employed to build the vector for that language.

For the piece of text whose language is to be identified a vector ($\vec{w}$) is computed in same manner as in training and then compared with vector of every language, by finding normalised dot products, as shown below:

$$\vec{w} \cdot \vec{l^j} = \frac{\sum_{i=1}^{N} w_i l_i^j}{|\vec{w}||\vec{l^j}|}$$

If the product is closer to 1, then more is the similarity of vectors. Hence, the language with maximum dot product is the language of the text.

If a text is to be identified then, N-gram frequency profile is built for this text and compared with every pre-computed language profile. This comparison is based on the distance measure done by counting the difference in the position of N-gram in unidentified text w.r.t the pre-computed language profile and adding up all the differences.

They studied the influence of training set size, amount of text to be classified and the options of languages available to system, to determine their effect on the performance of the system and their application in multilingual NLP applications.

First of all they displayed the precision in different cases depending on the dataset size used to train the system.

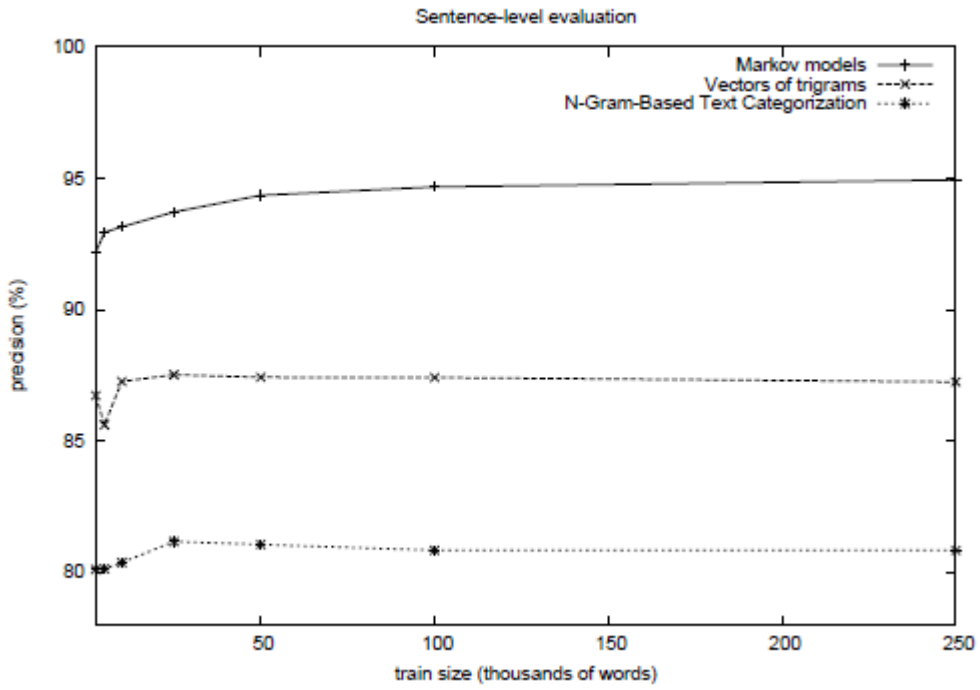


Figure 3: Precision of the systems distinguishing 6 languages

It was observed that when the size of text was bigger than 50 keywords then influence of size of training text was not important. Rest of the factors were found to be of much more importance. Longer texts were recommended but not necessary for higher accuracy. Markov Model was turned out to be best of other methods in case of language detection for smaller piece of texts.

These observations proved that sequencing information is as much important as the appearing frequency of n-grams. Moreover, due to faster response by this system it was selected as the best one for the role of statistical language recogniser.

Moreover, for the detection of languages present in multilingual text, the number of languages to be identified plays a major role. As the number of languages to be identified increases accuracy decreases. Also if the languages to be identified are similar in nature, this also leads to declination in accuracy and vice-versa.

[3] VarClass: An Open Source Language Identification Tool for Language Varieties, by Marcos Zampieri, Binyam Gebrekidan Gebre

This paper was about VarClass which is an open-source Language Identification Tool, which can either be downloaded or can be used through its graphical interface. This tool was more focused on the varieties of languages it can detect in comparison to the other tools present at that time. They are currently dealing with over 27 language models. Moreover, the average performance they reported for a challenging dataset was having 90.5% accuracy.

The algorithm calculated over the language models a simple likelihood function. One such likelihood function is shown in equation 1.

$$P(L|text) = \arg\max_{L} \sum_{i=1}^{N} \log P(n_i|L) + \log P(L) \quad (1)$$

N refers to the number of n-grams of testing data
n_i refers to the ith n-gram
L refers to the language models employed.

On the basis of given testing data, probability is calculated w.r.t. every language model, and highest probable model is determined as the language of the text.

Given below is a complete list of the available languages along with their respective ISO-3166 code:

Language	Country	ISO
Albanian	Albania	ALB
Bosnian	Bosnia	BIH
Bulgarian	Bulgaria	BGR
Croatian	Croatia	HRV
Czech	Czech Republic	CZE
Dutch	Netherlands	NLD
English	United States	USA
English	United Kingdom	GBR
French	Canada	CAN
French	France	FRA
German	Germany	DEU
Greek	Greece	GRC
Indonesian	Indonesia	IDN
Italian	Italy	ITA
Macedonian	Macedonia	MKD
Malay	Malaysia	MYS
Portuguese	Brazil	BRA
Portuguese	Portugal	PRT
Romanian	Romania	ROU
Serbian	Serbia	SRB
Slovakian	Slovakia	SVK
Spanish	Argentina	ARG
Spanish	Mexico	MEX
Spanish	Peru	PER
Spanish	Spain	ESP
Swedish	Sweden	SWE
Turkish	Turkey	TUR

Table 1: Language Models

For the calculation of language model corpora were collected from different sources like Wikipedia data), the SETimes Corpus (Tyers and Alperen, 2010), DSL Corpus Collection (Tan et al., 2014) and OPUS (Tiedemann, 2012).

VarClass provides its user with the choice to distinguish between the languages based on characters, words or both. But they avoided the usage of POS tags in spite of positive testing of its functionality to prevent the downfall of performance of the system.

The algorithm achieved a F-measure score of 90.5% by using character trigrams. Moreover, in the studies done by Zampieri and Gebre, 2012 discussion on the performance of the algorithm had already been done.

On the basis of documents of up to 300 characters, with 5400 such documents grouped into separate classes of 200 documents VarClass was evaluated. Also, during the training stage this test set was excluded.

The variation of performance was dependent on the languages and was found ranging from 0.68 for British English to 1.0 for Albanian. Average score of Recall (R), Precision (P) and F-measure (F) obtained in the 10 classes of four pluricentric languages are given below:

Language	Classes	P	R	F
English	2	0.784	0.725	0.753
Spanish	4	0.864	0.825	0.844
Portuguese	2	0.971	0.970	0.970
French	2	0.980	0.977	0.979

Table 2: Discriminating Pluricentric Languages

This paper explained about a Language Identification tool, VarClass which promoted the ideology of working towards the varieties of languages instead of focusing on a narrow segment of languages. This tool was considered to be of great use to the researchers related to linguistics research and NLP and also the people who aren't related to research in linguistics and NLP.

This tool left a scope for future research in the field of very short texts' language detection, even though this tool had shown satisfactory performance in identifying texts with word count greater than 3.

[4] Language Identification: The Long and the Short of the Matter,
by Timothy Baldwin, Marco Lui

This paper is based on the experiments using three datasets with distinct characteristics relevant to research context of the topic studied in this paper. The three corpora being employed were:

- EUROGOV
- TCL
- WIKIPEDIA

Corpus	Documents	Languages	Encodings	Document Length (bytes)
EUROGOV	1500	10	1	17460.5±39353.4
TCL	3174	60	12	2623.2±3751.9
WIKIPEDIA	4963	67	1	1480.8±4063.9

Table 3: Summarised Details of the Three Corpora Incorporated

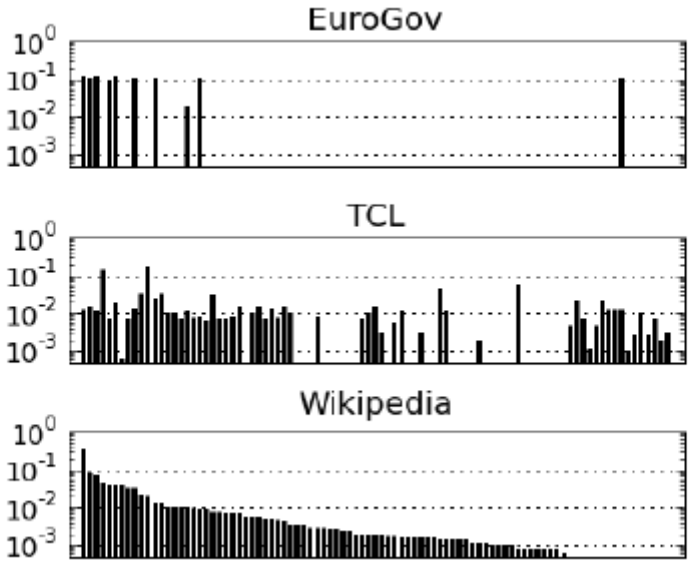


Figure 4: Language Distribution in the three datasets
(Vector of languages vs. the proportion of documents in that language)

They needed document representation which was language-neutral so any possibility of assuming the document's source language can be reduced. Thereby fulfilling their interest in detection of arbitrary web-documents. Moreover, they wanted to test if raw byte stream was sufficient for the extraction of character sequence or the character encoding of the document was required.

They experimented with two document representations, to understand the above stated query. They were:

(1) Codepoint n-grams
(2) Byte n-grams

In both of the above representations, document was represented as a feature vector of token counts. Knowledge of character encoding of document was essential in Codepoint n-gram to perform tokenisation. Byte n-grams, on the contrary can be extracted directly.

Moreover, the documents used uni-231 as a common encoding for all document to prevent:

(a) spurious matches between encodings
(b) over-fragmenting the feature space

EUROGOV and Wikipedia were in single encoding but the main issue of character encoding detection was with TCL only. TCL document was successfully transcoded to Unicode when character encoding was provided with document, else Mozilla browsers' encoding detection library was employed for encoding detection followed by the transcoding to Unicode.

The experiments were carried out over different combinations of the following options:

- tokenisation (×2): codepoint and byte n-grams
- model (×7): nearest-neighbour (COS_{1NN}, $SKEW_{1NN}$, OOP_{1NN}), nearest-prototype (COS_{AM}, $SKEW_{AM}$), SVM, NB
- n-gram (×3): n-grams ranging from 1 to 3

for a total of 42 distinct classifiers.

Every classifier is compared with the three datasets (Wikipedia, TCL and EUROGOV) using cross validation method of 10-fold stratified. The models are evaluated using micro-averaged F-score (F_μ), recall (R_μ) and precision (P_μ), also including macro-averaged F-score (F_M), recall (R_M) and precision (P_M). The average performance per document is marked by the micro-averaged scores. Also, in this scenario $F_\mu = R_\mu = P_\mu$.

On the contrary, average performance per language is indicated by macro-averaged scores. In this scenario, in every case 10 folds of cross-validation is calculated by calculating the average of F_M, R_M and P_M. Else, ZeroR classifier is employed as a baseline (majority class) assigner which assigns the highest priority training data language to each of the testing text.

Model	Token	$\mathcal{P}_M$	$\mathcal{R}_M$	$\mathcal{F}_M$	$\mathcal{P}_\mu/\mathcal{R}_\mu/\mathcal{F}_\mu$
ZEROR	—	.020	.084	.032	.100
COS_{1NN}	byte	.975	.978	.976	.975
COS_{1NN}	codepoint	.968	.973	.970	.971
COS_{AM}	byte	.922	.938	.926	.937
COS_{AM}	codepoint	.908	.930	.913	.931
SKEW_{1NN}	byte	.979	.979	.979	.977
SKEW_{1NN}	codepoint	.978	.978	.978	.976
SKEW_{AM}	byte	.974	.972	.972	.969
SKEW_{AM}	codepoint	.974	.972	.973	.970
OOP_{1NN}	byte	.953	.952	.953	.949
OOP_{1NN}	codepoint	.961	.960	.960	.957
NB	byte	.975	.973	.974	.971
NB	codepoint	.975	.973	.974	.971
SVM	byte	**.989**	**.985**	**.987**	**.987**
SVM	codepoint	.988	.985	.986	.987

Table 4: Results for byte vs. codepoint (bigram) tokenisation over EUROGOV

They were more focused towards the task of language identification of monolingual document and so they re-examined this task of language identification. Through their experimental activities they demonstrated the increase in complexity of the task for larger number of languages with greater class skew and shorter documents. They denied the necessity of explicit detection of character encoding and stated SVM with linear kernel or

a simple 1-NN model with cosine similarity, using a byte trigram or bigram document representation as the most consistent model for the task at hand. They also concluded that although it is easier to classify longer documents, but yet multilingual documents are a problem for language detection using a standard model.

[5] A high performance centroid-based classification approach for Language Identification, by Hidayet Takçı, Tunga Güngör

In a set of data, the central value is termed as centroid. This central value or centroid is used to represent the data belonging to a class. Hence, following the assumption of best representative of data being the central value of the class, single centroid represents every class. Training phase involves the calculation of centroid value for every class. Testing phase involves comparison of sample with each centroid to find the most similar class to new data sample. This method is based on vector space representation model and is one of the efficient method of language detection. Every vector ($\vec{d}$) represents a document and its every element points to term within the collection of documents. Generally, for assignment of weights to the terms, tf-idf metric is used.

Centroid vectors can be obtained from training data using two methods (Guan et al., 2009).

The first one is termed as arithmetic average centroid (AAC), which is calculated by finding mean values of corresponding term weights of the vectors of documents in the class. Class c_i's centroid vector $\vec{c_i}$ can be calculated as given below:

$$\vec{c}_i = \frac{1}{|D_{c_i}|} \sum_{d \in D_{c_i}} \vec{d}$$

The second method is termed as cumuli geometric centroid (CGC) , which is calculate by simply summing up the weights of each term:

$$\vec{c}_i = \sum_{d \in D_{c_i}} \vec{d}$$

To improve the training and classification performances and the discriminative power of centroid a new method named, Inverse Class Frequency (ICF) was proposed. This approach was unique to their previous work in three ways:

Firstly, focus was on language identification instead of document classification achieved using the class frequencies in centroid-based learning.

Secondly, a new updating factor which is class frequencies was used in the centroid instead of putting in the ratio of number of classes.

Finally, the idea of feature selection was dropped and simply a character-based feature set was used.

The relation of a term to a document is indicated by term weight in a particular document. Term weighting is one of the crucial factor that affects the text classifiers performance (Leopold and Kindermann, 2002). This can be approached either by using supervised term weighting or unsupervised term weighting. Here, ICF (inverse class frequency) is used as new method for term weighting. The average frequency of a term in a class and other classes in the corpus is a deciding factor for the relevance of a term for the class. This approach is a bit differentiated from idf method, which is based on the document or class count in which the term occurs.

This part deals with the comparisons of proposed methods with other methods employed in previous studies. Also, an explanation about the evaluation of performance measure and dataset used for experiment is given.

Module Train
Input
Training set
Output
Centroid vector $\vec{c}_i$ for each class c_i
Local variables
$freq_{i,k}$: total frequency of letter l_k in class c_i
$freq_k$: total frequency of letter l_k in the corpus
begin
for each class c_i
for each document d_j in class c_i
for each letter l_k in document d_j
$freq_{i,k} = freq_{i,k} + 1$ // Increment class frequency of letter l_k
$freq_k = freq_k + 1$ // Increment corpus frequency of letter l_k
end for
end for
end for
for each class c_i
for each letter l_k
$c_{ik} = \log(((freq_{i,k} + 0.001) / freq_k) * 10)$ // Calculate centroid value c_{ik} (sf=10)
end for
end for
end

Module Test
Input
Test document d
Centroid vector $\vec{c}_i$ for each class c_i
Output
Class index of document d
Local variables
$\vec{f}$: frequency vector of letters in the corpus
begin
for each letter l_k in document d
$f_k = f_k + 1$ // Increment document frequency of letter l_k
end for
for each class c_i
compute $sim(\vec{f}, \vec{c}_i)$
end for
return $\arg\max_i sim(\vec{f}, \vec{c}_i)$
end

Figure 5: ICF Algorithm for Training and Testing (sf=10 and z=0.001)

They employed a corpus named ECI/MC1 (European Corpus Initiative Multilingual Corpus I) which is comprised of scientific papers, legal texts, newspaper text, transcribed

speech, dictionary entries and novels and is a frequently employed corpora in language identification studies.

In this corpus, nine languages were being experimented upon. Every language had training to testing data ratio of 90:10 and 10-fold cross validation was employed in all the experiments done. Appropriate size of data was picked randomly from test data to fulfil the need for each length parameter.

Commonly used measures like f-measure metrics, recall and precision were employed to judge the success rates of the method. Let $correct_l$ and $predicted_l$ denote the set of test documents, respectively belonging to language (class) 1 and by the method are classified as belonging to language 1. Then calculation for the success rates can be done as shown below:

$$precision_l = \frac{|correct_l \cap predicted_l|}{|predicted_l|}$$

$$recall_l = \frac{|correct_l \cap predicted_l|}{|correct_l|}$$

$$f - measure_l = 2\frac{precision_l * recall_l}{precision_l + recall_l}$$

The results obtained through experiments conducted in previous section came out to be in favour of inverse class frequencies proving its superiority in generating successful results in comparison to other classifiers. A plus point for this method was its comparatively low space and time complexity for training and testing. Time complexity was of the order O(km), where k were the number of languages and m were the features. These properties paved the way for efficient processing of data, formation of centroids and calculation of similarities.

This paper did the empirical measure of training and testing time complexities and performed a comparison with other methods.

The time requirements of n-gram and short-term methods were more due to the usage of large feature vectors. Moreover, ICF had lower time complexity than SVM in spite of using the same feature set. Hence, they stated the superiority of ICF approach in terms of time complexity and success rate.

[6] Automatic Language Identification for Romance Languages using Stopwords and Diacritics, by Ciprian-Octavian Truică, Julien Velcin, Alexandru Boicea

This paper proposed a statistical technique of automatically detecting language with the help of two dictionaries one being that of diacritics and another containing stopwords. Combination of most common words for the studied language written with and without diacritics helped in the construction of stopwords dictionary. The diacritics of each language helped in the creation of diacritics dictionary. The method either judged the term weight based on occurrence of term in each language, or used the dictionaries as it is.

News article corpora along with the Twitter corpora was employed for experimental tests. Source of the data was used to determine the texts' language, which helped in determination of the accuracy of their approach. This method employed pre-processed text free of punctuations and non-alphabetical characters to improve the language detection accuracy.

Creation of stopwords and diacritics dictionary was done prior to the application of algorithm. Since the text in Twitter corpus were written without diacritics, hence they could be misclassified. So a stopword dictionary was created with manual addition of stopwords without diacritics.

Language	Diacritics
French	àâæçèéêëîïôœùûü
Italian	àáèéìíòóùú
Portuguese	áâãàçéêíóôõú
Romanian	ăâîșşțţ
Spanish	áéíóúñü

Table 5: Diacritics of some languages

This method judged the text based on the term frequency of diacritics and stopwords. Possibility of term being a diacritic or a stopword is equi-probable. Equation (1) assigned each text a score judging from the two dictionaries, and the unknown text was judged to be the language with highest score. In the absence of diacritics only stopwords dictionary was employed for score calculation, p=1. Weighted sum of frequency of diacritics and stopwords led to the calculation of final score. The concept of weight was thought to be of greater importance as few terms which are specific to a language can heavily affect the

score in comparison to the one's which are common between two or more languages, which was similar to IDF (inverse document frequency) factor.

$$score(text, lang) = p.\sum_{w} f(w, text).weight(w, lang) + (1-p).\sum_{d} f(d, text).weight(d, lang) \quad (1)$$

$$f(term, text) = \begin{cases} f_{term,text} \\ \log(1 + f_{term,text}) \end{cases} \quad (2)$$

$$weight(term, lang) = \begin{cases} 1 \\ \frac{N}{n} \\ \log\left(1 + \frac{N}{n}\right) \end{cases} \quad (3)$$

In Equation (1), the following notations are used:

- *lang* represented the language that is being tested
- *w* represented a stopword in a given language's stopwords dictionary
- *d* represented a diacritic in a given language's diacritics dictionary
- *N* refers to the number of languages tested
- $n = |\{term: term \in lang\}|$. Here, n > 0 as it appears in at least one language.

For correlation of stopwords with diacritics and for the increment of the impact of terms, parameter $p \in [0, 1]$ was used. The term frequency was computed using function *f(term,text)* in Equation (2). Moreover, the term weighing function *weight(term, lang)*, is computed using several approaches as portrayed in Equation (3).

On the conclusive note, Stopwords proved to be highly effective for automatic language detection or identification. Possibility of similar stopwords with different semantical meanings in related languages were found which often created a possibility for misclassification of text. Also several diacritics were found, which were very specific to

the language and their occurrence can avoid the need of scoring due to immediate results. Hence, if a text is correct is every sense i.e., on the basis of its diacritics and stopwords, then a simple observation can reveal the text of the language avoiding the complex methods of scoring and easing the detection.

CHAPTER-3
SYSTEM DEVELOPMENT

3.1 Tools Used

3.1.1 PHP

PHP is a recursive acronym for PHP: Hypertext Pre-processor. It is a server-side scripting language which is open source and general purpose, designed with a primary motive of web development but can be used as general – purpose programming language. It can be either embedded in HTML code or cam be used in association with web frameworks, web content managements and web templates. It is processed as a CGI executable or as a module in the web server by PHP interpreter.

3.1.2 MySQL

MySQL is an open-source RDBMS (relational database management system) software with support for query based processing. It employs a structured query language for processing of its databases and tables. Its source code is made public under GNU General Public License and under a variety of proprietary agreements. MySQL was previously owned by MySQL AB, a Swedish company but was later bought by renowned Oracle Corporation. It also has a paid edition with added features for proprietary uses.

3.1.3 XAMPP

XAMPP is a complete package comprising of PHP and Perl interpreters, Apache HTTP server and Maria DB database. It is open source, cross platform and available for free on internet. XAMPP is an abbreviation in which X stands for Cross-Platform, A stands for Apache, M stands for MariaDB, P stands for PHP and another P stands for Perl. It provides user with facility for easy to create local web servers for testing and deployment

of webpages. It is complete package packed with everything required for setting up of web servers like scripting language, server application and database. And since it is cross platform hence, works equally well on all the systems of Windows, Linux and Mac. Also, it gives near to real experience of the deployments made by live server, so transitioning is easier to do as well.

3.1.4 Sublime Text

Sublime Text is a source code editor which is available in free as well as paid version. It is cross platform and proprietary having Python Application Programming Interface. It supports various plugins allowing users to extend and utilize its various features. Even though it has a native support for markup and programming languages.

3.1.5 NLTK

NLTK or Natural Language Toolkit is a large suite comprising of programs, libraries and corpora for symbolic and statistical technique to study natural language processing. It is available for several languages and has been written in Python programming language. Steven Bird and Edward Loper were the mind behind the development of NLTK. It started as computational linguistics course in University of Pennsylvania in 2001. NLTK is a great resource to work with human lingual and written language data. It provides with about 50 corpora and lexical resources for language processing.

NLTK supports semantic reasoning functionalities, tagging, parsing, classification, stemming, tokenization and wrappers for industrial strength NLP libraries.

3.1.6 Python

Python is a 3rd generation language with abilities to combine some 4th generation features in this general purpose programming language. It completely supports object-oriented programming and structured programming. Python allows for more concise code in comparison to C++ or Java, and uses indentations to segregate instead of parenthesis. Thus, it allows to write clean program.

3.2 Model Development

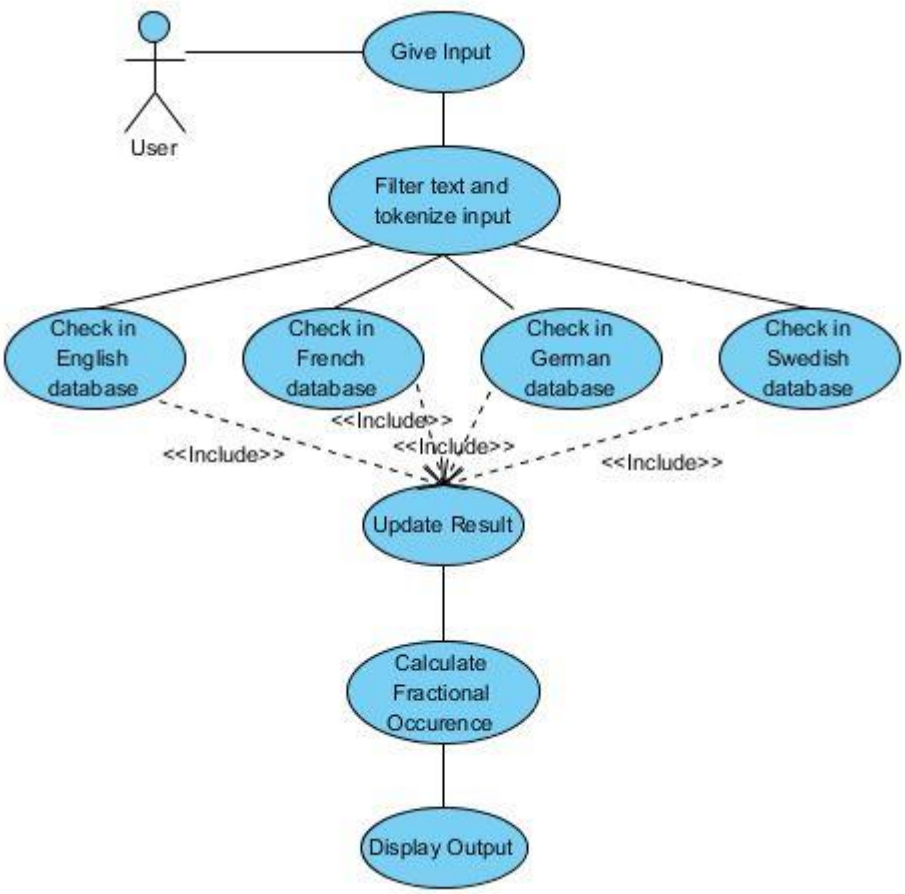


Figure 6: Use Case Diagram for Phase1

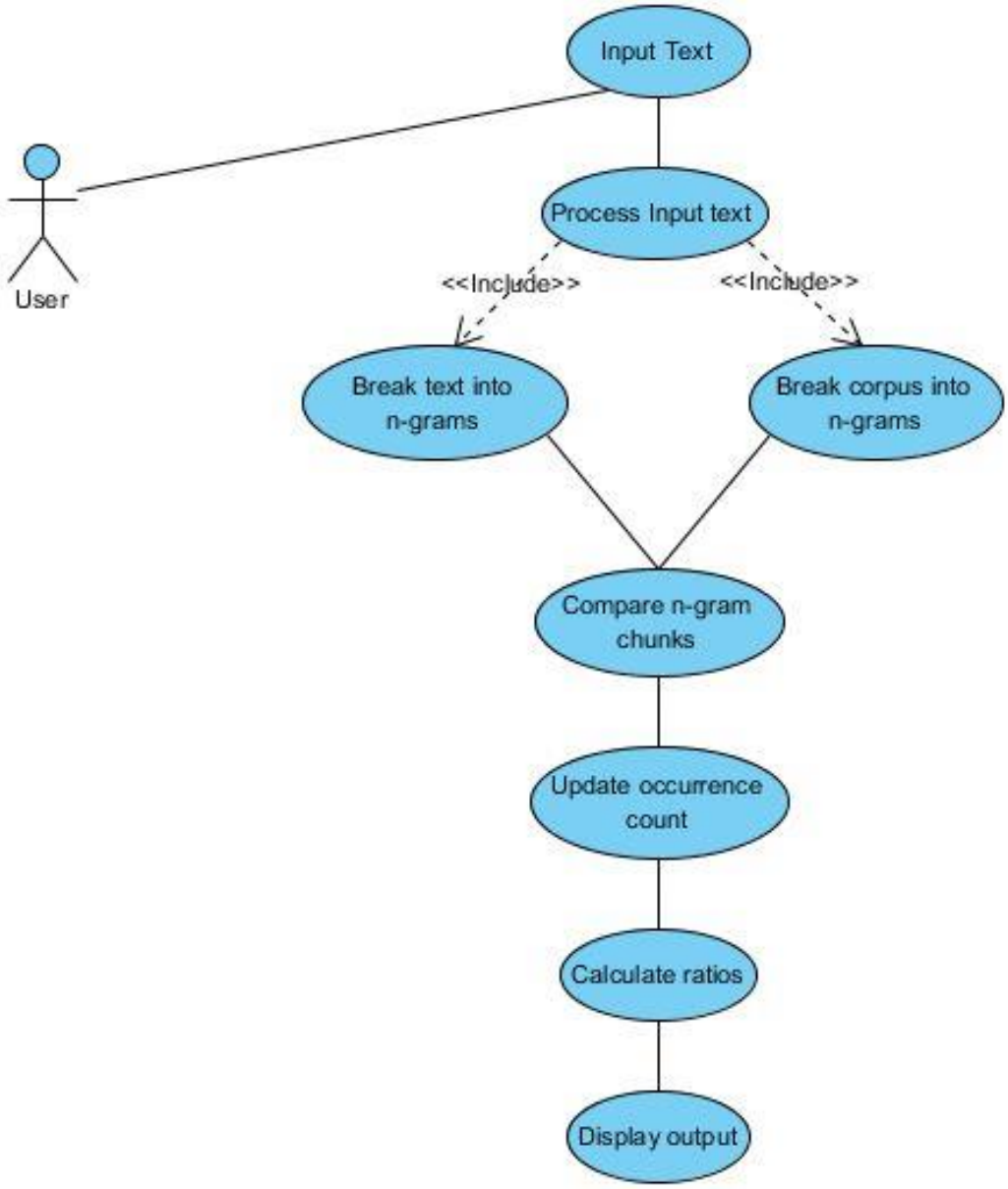


Figure 7: Use case diagram for Phase 2

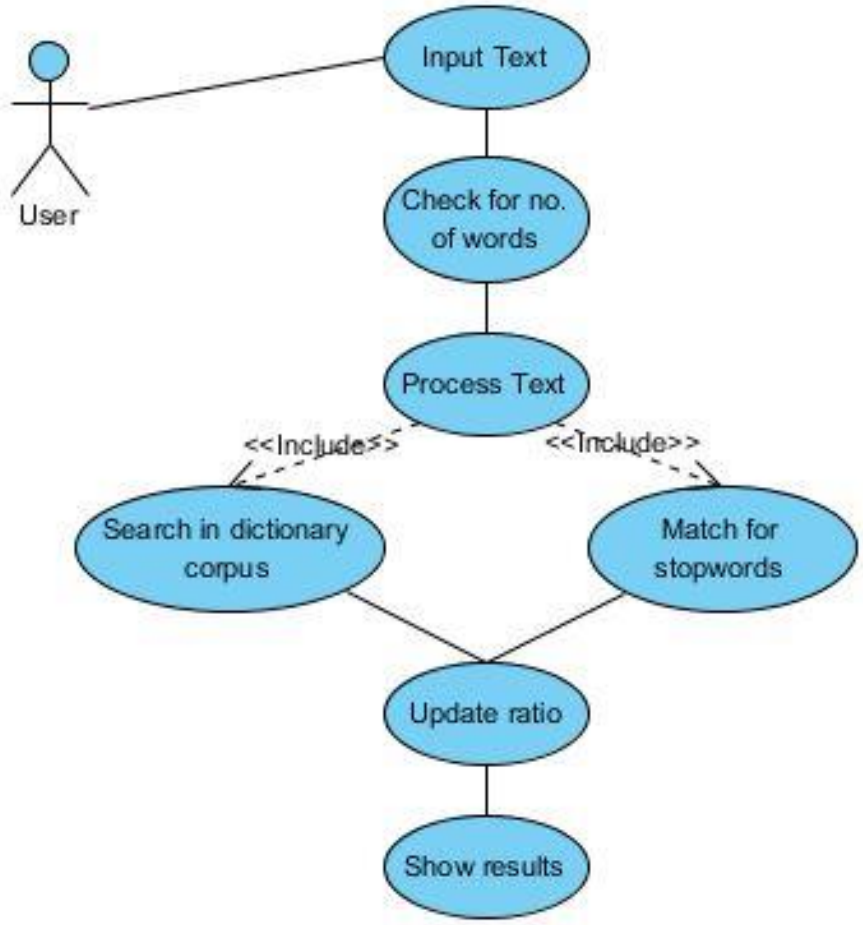


Figure 8: Use Case Diagram for Phase 3

3.2.1 For Phase 1 using n-gram model(word-wise)

Flow of Events:

1. Take input
2. Filter text and Tokenize the input
3. For all the tokens:
    a. Check the index of previous word's language
    b. Check in that language
    c. Update result
    d. If word found then *continue*

e. Check in rest 3 languages in order English>French>German>Swedish which ever applicable.
f. Update the result

4. Calculate fractional occurrence
5. Display output

3.2.2 For Phase 2 using n-gram model(character-wise)

Flow of events:

1. Take input
2. Process text
    a. Tokenize the text into sets of n-grams
    b. Fetch corpus
    c. Tokenize the corpus sentence wise
    d. Further tokenize each sentence into sets of n-grams
    e. Update count
3. Calculate ratios and find max ratio
4. Display result

3.2.3 For Phase 3 using combined search and stop word approach

Flow of events:

1. Take input
2. Count number of words in input
3. If number of words < 3
    a. Tokenize word-wise
    b. Search in the dictionary corpus
    c. If word found update count
    d. If common occurrence language priority is English>French>German>Swedish
    e. Find the language with highest count
4. If number of words >= 3

a. Create a set of input text
b. Create a separate set of stop words of each language
c. Find common elements between set of input text and each set of stop words.
d. Count the common elements between each comparisons
e. Find the set with most number of counts

5. Display result

3.3 Mathematical calculation

- $Total\ Words = \sum_{i=1}^{4} Words\ encountered\ in\ Language\ \ i$

- $FPLi = \frac{\sum Words\ encountered\ in\ Language\ i}{Total\ Words}$

Where, FPLi is Fractional Presence of Language i

- $Final\ result = \max(Fractional\ Presence\ of\ Languages)$

- $Total\ ngrams = \sum_{i=1}^{4} ngrams\ encountered\ in\ corpus\ of\ Language\ \ i$

- $FPLi = \frac{\sum ngrams\ encountered\ in\ corpus\ of\ Language\ i}{Total\ ngrams}$

Where, FPLi is Fractional Presence of Language i

CHAPTER-4

PERFORMANCE ANALYSIS

We tested out application for multiple numbers of inputs and for different cases like

- Single language English
- Single language French
- Single language German
- Single language Swedish

And we did it for different numbers of inputs like

- 1 word
- 10 words
- 100 words
- 500 words

I have completed this project in three phases to fulfil my objective with improvement in time and accuracy in every phase.

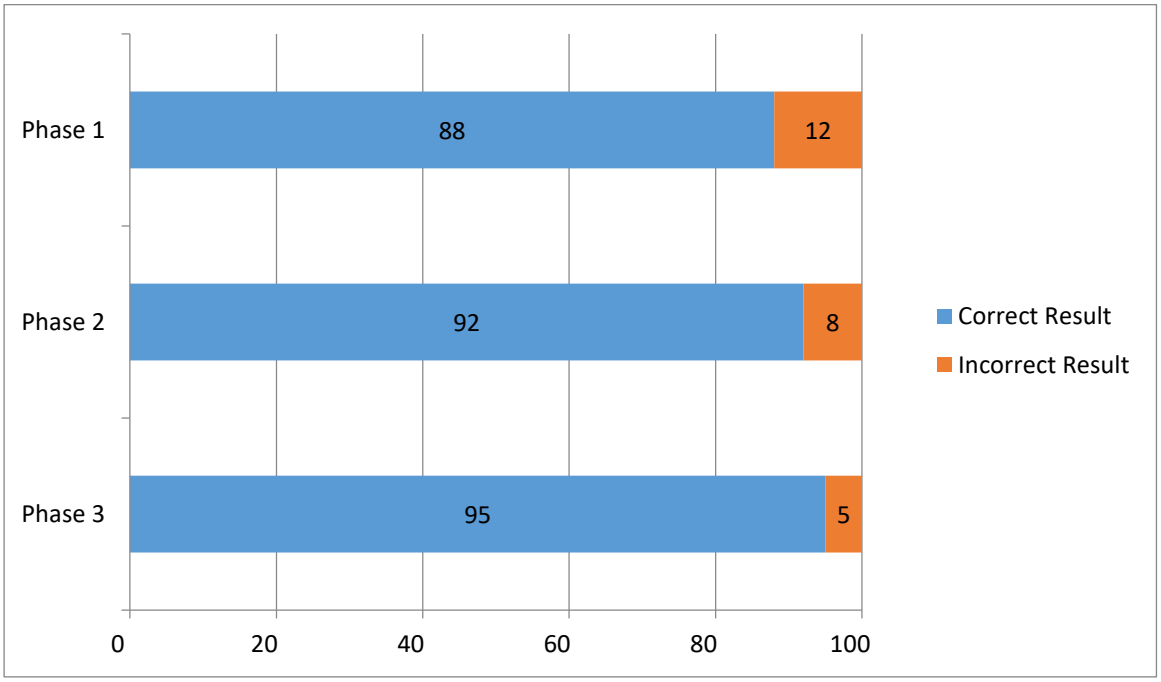
- Phase 1 being ordinary dictionary search with slight modification i.e., using 1 previous word to narrow the possibility of language of word
- Phase 2 being n-gram character-wise tokenization approach
- Phase 3 being a combination of search and stop word based approach depending upon number of words given as an input

Phase 1 involved basic approach involving accurate but more time taking methods.
In Phase 2 time taken was reduced but accuracy was dependent on number of words present.
In Phase 3 time taken was greatly reduced in general and accuracy was increased but for accuracy in specific cases time taken was increased

Following are the analysis of correctness for my application within all three phases:

i. This is an analysis on English language text. Result for all the phases are depicted underneath.

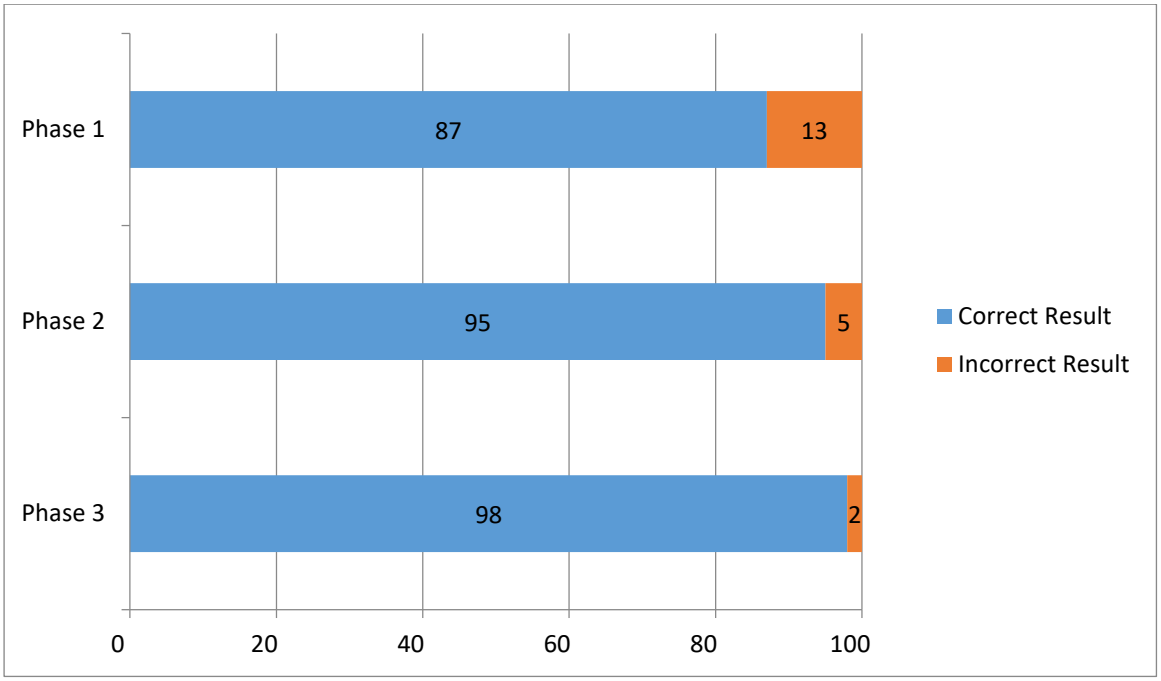
   This error came due to presence of slangs and abbreviation. Moreover several other factors were responsible like the type of corpus used.



Graph 1: Analysis on English Text (figures in %)

ii. This is an analysis on French language text. Result for all the phases are depicted underneath.

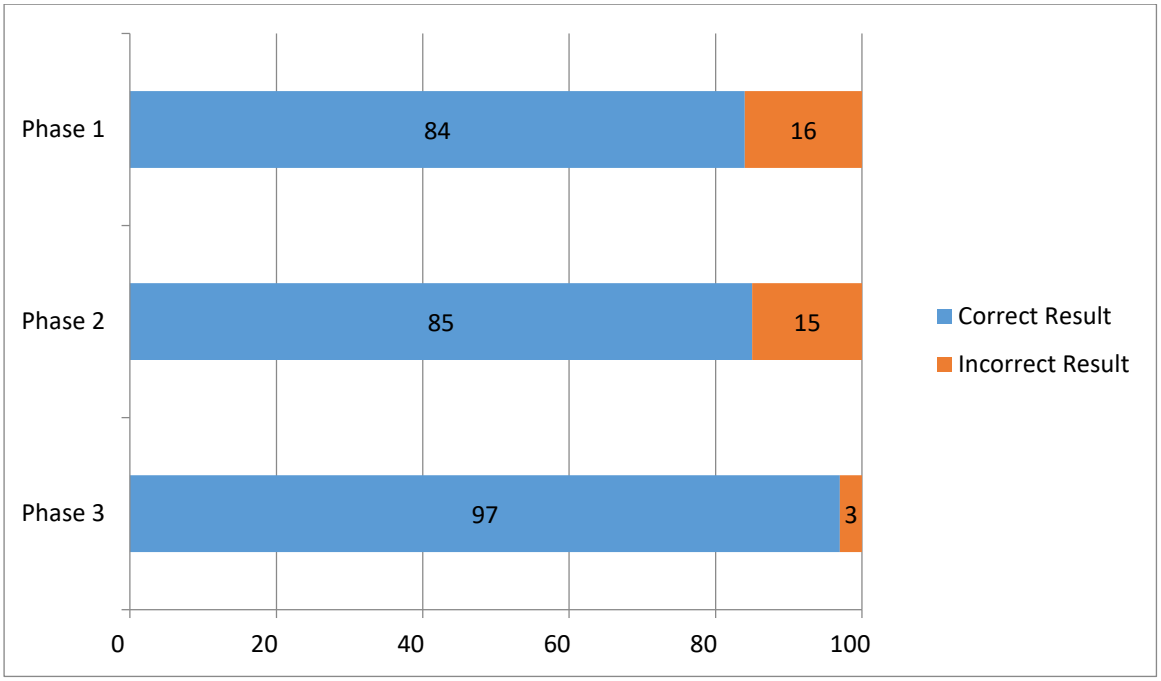
This error came due to presence of slangs and abbreviation. Moreover several other factors were responsible like the type of corpus used.



Graph 2: Analysis on French Text (figures in %)

iii. This is an analysis on German language text. Result for all the phases are depicted underneath.

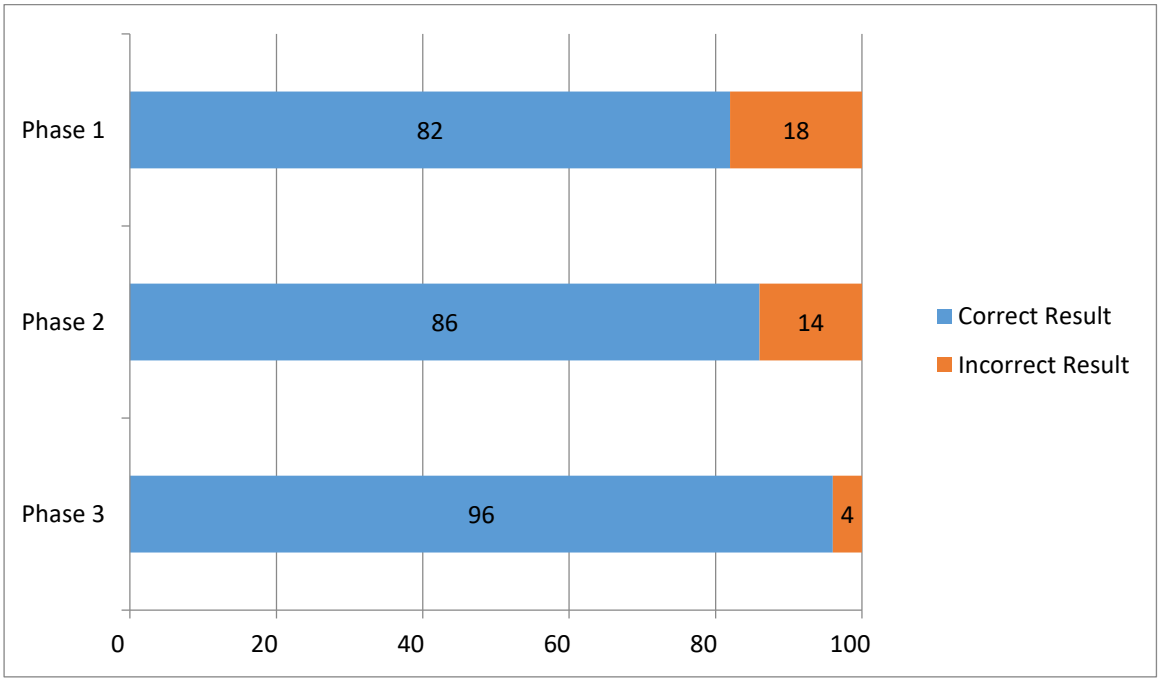
This error came due to presence of slangs and abbreviation. Moreover several other factors were responsible like the type of corpus used.



Graph 3: Analysis on German Text (figures in %)

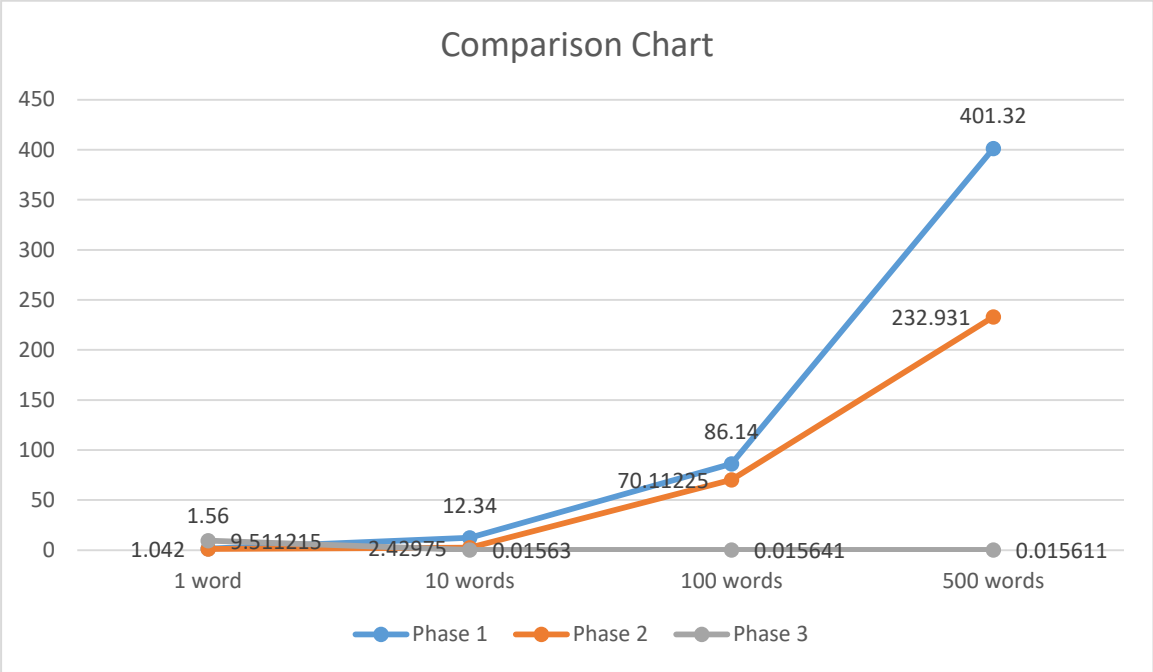
iv. This is an analysis on Swedish language text. Result for all the phases are depicted underneath.

This error came due to presence of slangs and abbreviation. Moreover several other factors were responsible like the type of corpus used and diacritics used which was not used in training of text.



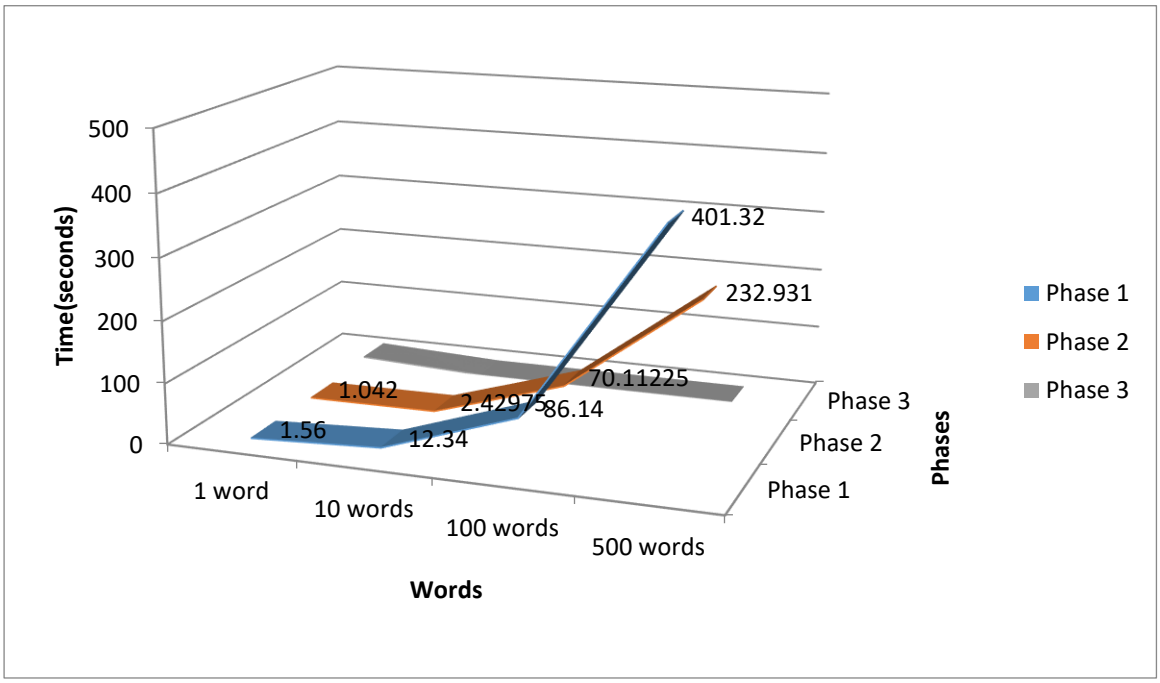
Graph 4: Analysis on Swedish Text (figures in %)

v. This graph shows time required for detecting language in all the three phases and shows the drastic changes observed in the time taken in each approach



Graph 5: Time Comparison (in seconds) for Detecting Language in all Three Phases

vi. This graph shows time required for detecting language in all the three phases and the number of words used to depict the change in time taken in each phase



Graph 6: Time required (in seconds) for detecting language using the three approaches belonging to each phase

CHAPTER-5
CONCLUSION

5.1 Conclusion

Various novel approaches proposed by different researchers and applications of language identification have been reviewed. Language identification is done using n-gram techniques, Centroid based techniques, different classifiers based techniques, supervised techniques, profile feature based techniques, hybrid techniques etc. Different approaches worked for different set of languages and different type of documents and gives high level accuracy for identifying the text. A number of limitations are outlined by different researchers which can be the base of future research.

Apart from this based on my personal experience of comparing the approaches used in the three phases, I came across various advantages and dis-advantages of each approach leading to better understanding of the various tools and techniques that are needed to be utilised to overcome these obstacles.

In Phase 1 I utilised n-gram approach (word-wise) to detect the language of the text. This approach on one side was accurate but was time taking as traversal through whole dictionary list was required in order to detect the language. In it I observed that within n-gram as the value of n increases corresponding correctness of result and efficiency increases, but at the same time it also leads to increase in complexity and hence execution time is affected. Thus, both of these factors act as trade off characteristics for each other. Hence, great care has to be taken in order to deal with it.

In Phase 2 I utilised n-gram approach (character-wise) to detect the language of the text. This approach was based on the concept that in all languages several words tend to occur collectively in most of the cases. This approach was also accurate and was less time taking but in this approach accuracy and number of words in text were directly proportional. So, for single or fewer number of words there were more chances than previous approach for occurrence of ambiguous answers. Moreover there was a high dependency on the corpus used as due to usage of “The Bible” as a corpus for the task usage of modern day words also led to the possibility of ambiguous answers. Same

scenario was also observed when there was a possibility of usage of words which occurred in more than one language.

This ambiguity led to development in Phase 3 which combined the approach of conditional search and stop words approach. Condition being if there were less than 3 words, words were needed to be searched in self assembled corpus else the matching was done through stop words. Stop words are the set of words which are unique to a language and are needed to be removed for several NLP applications or in search engines to get better and accurate search results. Common words between stop words set and text entered were found and highest count text was termed the language of the text. This approach overcame the disadvantage of Phase 2 by detecting single words more accurately moreover usage of stop words for detection of language drastically reduced the time for cases with words more than 3 hence overcoming disadvantage of Phase 1. As a disadvantage in order to overcome the disadvantage of Phase 2, the code took slightly more time than the other cases hence accuracy was increased but on the cost of time taken.

5.2 Future Work

A number of novel approaches are proposed by different researchers for language identification. Most of the researchers worked for language identification of monolingual documents (web pages, search engine queries, microblog posts, tweets etc.). Multilingual documents were less considered. It was also observed through researcher papers that different classifiers used by them for classification of documents also affect the accuracy of language identification. It was also studied that language of short documents were quite hard to determine as compared to long documents. So, in future works, the issue of multilingual documents can be considered. Along with it 3-gram and 4-gram approaches can be experimented with in order to observe the effect of this trade-off. A hybrid model can also be proposed that can successfully identify the short and long documents both with high accuracy.

5.3 Applications

Language Identification is an interesting problem. In many applications, it works as a primary step of some larger process. It provides the facility of using background information about the language and using specialized approaches in many natural language processing tasks that deals with the collection of texts written in different language. With the increase of international communication and business, systems are required for correctly identifying the language of documents (emails, letters, web pages etc.). The task of language identification is working in various areas of natural language processing.

i. Resnik (1999) employed language identification technique to create a bilingual corpus using a system called Strand.

ii. Classification of text from a noisy source containing contextual errors can employ language identification techniques for studying system behaviour. Such text can be either a data of OCR system or can be a piece of unsupervised e-mail.

iii. Several operations like parsing, stemming or spell checking requires the prior knowledge of the language of text. Also pre-processing tasks like machine translation, question-answering, text categorization, summarization etc. requires the knowledge of the language of the text, they operate on.

iv. Language Identification can be used for detecting and informing about the sensitive languages comprised of few keywords that are used by terrorists while conversing with one another. This can be done by identification of spoken languages using complex systems.

v. Language identification can be used as a biometric authentication to pattern recognition to human-computer interaction when it is employed for classification and signal modelling.

vi. Machine translation employs language identification as a primary and very important step. It involves the identification of language by employing required technique. And then text is further processed for mapping of pronunciation and preservation of meaning.

vii. Language identification can employed in task of text categorization. When the text is segregated on the basis of written material, then the most basic and important step is the identification of language of the text for the successful classification of text.

REFERENCES

[1] Xi Yang and Wenxin Liang, An N-Gram-and-Wikipedia Joint Approach to Natural Language Identification, 978-1-4244-7820-0/10/$26.00 ©2010 IEEE , 2010, pp05-12

[2] Muntsa Padró and Llúıs Padró, Comparing methods for language identification, Language identification in web documents using discrete hmms.,2004 *PR*,pp 13-18 37(3):583–594.

[3] Marcos Zampieri1, Binyam Gebrekidan Gebre, VarClass: An Open Source Language Identification Tool for Language Varieties,9th International conference on Language Resources and Evaluation, 2014, pp 19-23

[4] Timothy Baldwin and Marco Lui, Language Identification: The Long and the Short of the Matter, International Journal of Computer Science and Applications, ©Technomathematics Research Foundation Vol. 7, No. 1, 2010, pp. 24 –28

[5] Hidayet Takçı, Tunga Güngör, A high performance centroid-based classification approach for language identification, Pattern Recognition Letters 33 ,2012,pp 29-34

[6] Ciprian-Octavian Truică, Julien Velcin, Alexandru Boicea, Automatic Language Identification for Romance Languages using Stop words and Diacritics, 17th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing.

APPENDICES

Code for n-gram approach (character-wise) using python

```
import nltk
from nltk import ngrams
from nltk.corpus import genesis
from nltk.tokenize import sent_tokenize,word_tokenize



train_text_en = genesis.raw("english-web.txt")
train_text_fr = genesis.raw("french.txt")
train_text_gm = genesis.raw("german.txt")
train_text_sw = genesis.raw("swedish.txt")

#sentence wise tokenization of corporas
tok_en = sent_tokenize(train_text_en)
tok_fr = sent_tokenize(train_text_fr)
tok_gm = sent_tokenize(train_text_gm)
tok_sw = sent_tokenize(train_text_sw)

#sample space
lentg = [0,0,0,0]

#favouring events
langprob = [0,0,0,0]

language = {0: 'English', 1: 'French', 2: 'German', 3: 'Swedish'}

#value of n-gram
```

```
#here it is 3-gram
n = 3

text = input("Enter text: ")
tgrams = list(ngrams(text.lower(),n))

#for english
twograms = []
for i in range(len(tok_en)):
    twograms = list(ngrams(tok_en[i].lower(),n))
    lentg[0] += len(twograms)
    for gram in tgrams:
        for grams in twograms:
            if ( gram == grams ):
                langprob[0] += 1;

#for french
twograms = []
for i in range(len(tok_fr)):
    twograms = list(ngrams(tok_fr[i].lower(),n))
    lentg[1] += len(twograms)
    for gram in tgrams:
        for grams in twograms:
            if ( gram == grams ):
                langprob[1] += 1;

#for german
twograms = []
for i in range(len(tok_gm)):
```

```
    twograms = list(ngrams(tok_gm[i].lower(),n))
    lentg[2] += len(twograms)
    for gram in tgrams:
        for grams in twograms:
            if ( gram == grams ):
                langprob[2] += 1;

#for swedish
twograms = []
for i in range(len(tok_sw)):
    twograms = list(ngrams(tok_sw[i].lower(),n))
    lentg[3] += len(twograms)
    for gram in tgrams:
        for grams in twograms:
            if ( gram == grams ):
                langprob[3] += 1;

#calculating probabilities
final = [0,0,0,0]
for i in range(4):
    final[i] = langprob[i]/lentg[i]

#final verdict
max_index = final.index(max(final))

print(language[max_index])
```

Code for combination of stop word and search approach using python:

```
import nltk
from nltk.corpus import dictionary, stopwords
from nltk.tokenize import sent_tokenize,word_tokenize

#search module for 1-2 words data
def search(text):
    langc = [0,0,0,0,0]
    text = list(word_tokenize(text))
    language = {0: 'english', 1: 'french', 2: 'german', 3: 'swedish', 4: 'null'}
    train_txt_en = dictionary.raw('english')
    train_txt_fr = dictionary.raw('french')
    train_txt_gm = dictionary.raw('ngerman')
    train_txt_sw = dictionary.raw('swedish')
    tok_en = word_tokenize(train_txt_en)
    tok_en = [word.lower() for word in tok_en]
    tok_fr = word_tokenize(train_txt_fr)
    tok_fr = [word.lower() for word in tok_fr]
    tok_gm = word_tokenize(train_txt_gm)
    tok_gm = [word.lower() for word in tok_gm]
    tok_sw = word_tokenize(train_txt_sw)
    tok_sw = [word.lower() for word in tok_sw]
    for word in text:
        if word in tok_en:
            langc[0] += 1;
        elif word in tok_fr:
            langc[1] += 1;
        elif word in tok_gm:
            langc[2] += 1;
```

```
        elif word in tok_sw:
            langc[3] += 1;
        else:
            langc[4] += 1;
    for i in range(5):
        if langc[i] > 0:
            return language[i]


#detecting max ratio and giving the language
def detect_lang(text):
    ratio = {}
    token = word_tokenize(text)
    words = [word.lower() for word in token]

    for lang in stopwords.fileids():
        #creating a set of stopwords of a language
        sw_set = set(stopwords.words(lang))
        #creating a set of input text
        word_set = set(words)
        #finding intersection of two sets
        common = word_set.intersection(sw_set)
        ratio[lang] = len (common)

    lang = max(ratio, key = ratio.get)
    return lang


#main part of the code
```

```
text = input('Enter text: ').lower()
if len(text.split()) <= 2:
    lang = search(text)
else:
    lang = detect_lang(text)

if lang in ['english','french','german','swedish']:
    print(lang)
else:
    print('Unable to detect language')
```