LANGUAGE IDENTIFICATION OF TEXT

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

in

Computer Science and Engineering/Information Technology

By

Isha Pathak (141354)
Arpan Sharma (141450)

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 Identification of Text"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/Information Technology** 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 August 2017 to May 2018 under the supervision of **Ms. Ruhi Mahajan (**Assistant Professor (Grade-II), Computer Science & Engineering and Information Technology).
The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Isha Pathak (141354) ……………………….

Arpan Sharma (141450) ………………………

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 (Grade-II)
Computer Science & Engineering and Information Technology
Dated:

Acknowledgement

First of all, we would like to thank **Prof. Dr. S P Ghrera, Head of Department of Computer Science Engineering & Information Technology** for his kind support, constant encouragements and valuable discussions which helped in completion of the project.

We would also like to express our sincere gratitude towards our supervisor **Ms. Ruhi Mahajan,** Assistant Professor (Grade-II)**,** for her support, encouragement, and kind co-operation as well as for providing necessary information regarding the project, which has been instrumental for the success of this project. It was an invaluable experience for us to be one of her students. Because of her, we have gained a careful research attitude.

Lastly, we would also like to thank our parents for their love and affection and especially their courage, which inspired us and made us believe in ourselves.

Table of Content

LIST OF ABBREVIATIONS

1. NLP – Natural Language Processing.
2. LID – Language Identification of Text.
3. ING – Improved N gram.
4. ONG – Original N gram.
5. MNG – Modified N gram.
6. NBC – Naïve Bayesian Classification.
7. CFA – Cumulative Frequency Addition.

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Language identification is one of the most important step that need to be taken for the tasks that deals in the domain of NLP. Automatic detection of native language of documents can be used as a pre-filtering step to improve the quality of input data for improved analysis. Tasks such as summarization, machine translation, part of speech tagging, stemming, text mining etc. need to know the language of a given text in order to process it for further analysis. It can also be used as a filtering technique to support users of the task of information retrieval. The existing methods for identifying long string of texts does not work well with short strings of text which still remains a challenge in the field of NLP. In any real world use, a NLP system will encounter words that are not in its lexicon. If any system does not recognize these unknown words, its performance will degrade. Moreover, presence of misspelled words in the query can cause confusion for the system and thus lead to declaring incorrect results.

Language identification is the task of determining the natural language of a text based on the contents of the document. This report describes a method for language identification that is based on Cumulative Frequency Addition. An advantage of this method is that it uses a very simple and straightforward algorithm and does not require a large data set or high amount of training time. Our method is currently trained to recognize six different languages. In addition, we also tackle misspelled words by using Damerau-Levenshtein distance algorithm as well as try to predict the correct spelling and include it in our analysis which helps in improving the accuracy of our results.

Chapter 1
INTRODUCTION

1.1. INTRODUCTION

Language Identification falls in the category of natural language processing which is the field of describing how computers can decipher meaning and value from human languages. Automatic detection of native language of documents can be used as a pre-filtering step to improve the quality of input data, which has become increasingly important in the present scenario when the amount of data is increasing at an enormous rate. It has become an essential pre-processing step for other techniques such as machine translation, part of speech tagging, stemming, text mining and information retrieval, and has become the need in order to facilitate further textual analysis.

Language Identification can be used using two techniques, namely Computational Techniques and Non-Computational Techniques. Computational techniques are based on statistical methods and requires large set of training data for each of the language. Non-Computational techniques require that researcher must have extensive knowledge about the language that has to be identified.

In this paper we propose a method that identifies the natural language of the document as well as handles the misspelled words. In this method, we have taken the assumption that the document will be monolingual in nature and have addressed the problem of language identification in documents that contain text from only one of the languages from the candidate set taken into account. Our system has been designed to detect the texts in six different languages namely English, French, Spanish, Russian, German and Latin. In addition to detecting the correct language of the text it can also tell how closely the text is related to other languages as well. It is also reliable in determining if the sample text contains some spelling errors and can also suggest correct spellings for the misspelled words.

1.2. PROBLEM STATEMENT

Many accurate methods for language identification of long texts are present. However, these methods rarely work well with short texts and are typically evaluated in a manner that does not reveal all the problems in applications. Therefore its identification, still presents a major challenge in the field of NLP. In addition, in any real world use, a NLP system will encounter words that are not in its lexicon (say, misspelled words). Unknown words are problematic because a NLP system will perform well only if it recognizes the words that is meant to analyze. The more words system does not recognize the more the system's performance will degrade.

1.3. OBJECTIVES

Our goal is to develop a system that deals with short strings of text and can accurately identify the native language of the text. The system shall also automatically detect if any misspelled words are encountered in the users input and should then find the nearest match of that word in the database of our training set to determine the native language after combining it with result of other words in the query string.

1.4. METHODOLOGY

1.4.1 Building the database

The first step is to build a database of the words in order to develop the Lexicon of the languages considered in our research. This is done by web scrapping the news feeds, RSS feeds, wiki pages, web pages and articles written in these languages that were present on the internet. With the help of web scrapping these pages, we are able to get around 10,000 lines of text from each of the language. These documents are then processed through different smoothening and pruning techniques so that they do not corrupt our database. Junk characters, indentation spaces and words with other discrepancies are handled carefully and removed from these documents. Then these documents are processed line by line and tokenized into words. Frequencies are calculated for all the words present in the documents of each of the language and are then stored in the database for the corresponding language.

1.4.2 Determining the natural language initially

The string text or the document of which the language is to be identified, is firstly tokenized into words and is then further smoothened by removing the extra space characters, indentation and punctuation marks that do no play any role in identification of the language of the text. Then all of these words are run against the database for all of the languages and total frequency count for each of the language is calculated by summing up the frequencies of all the words that occurred in each of the language. The language with maximum frequency count is declared as the natural language of the document or the text under consideration.

1.4.3 Handling the unknown words

Since we are working with a very small database which hasn’t been trained over a large set of text from the languages under consideration, there may be a possibility that we encounter certain words that are not present in the database for any of the language. At first, we label those words as unknown words. Now these words are run against the dictionaries for all of the languages in the candidate set and the words which find a match are added to the database of that particular language so that we are able to provide more accurate results for future queries. Also a constant frequency of one is added to the calculated frequency count against these words. With this approach, we are improving our system with every new query being entered and thus making our system learn new words that it had missed during the formation of lexicon for each of the language. Those words which are not found in the dictionary of any of the language have to be misspelled words and they are separated out for further processing.

1.4.4 Handling the misspelled words

Misspelled words are tackled in a different manner. A distance factor (minimal number of one or more operations needed to transform the first string to the second) is calculated from the words of roughly similar length using the Damerau-Levenshtein algorithm by comparing them to the words in the existing database. Damerau-Levenshtein distance calculates how far the two strings are in terms of 4 basic operations – deletion of a character, insertion of a character, substitution of a character and transposition of two characters.

A list of possible suggestions for a particular word is generated for each of the language and then by using the Pythons' enchant library, we analyze the confidence factor for all of the suggestions and then the one with the highest value, is declared as the word that has the most correct form of that particular misspelled word. Similar task is done for all of these words and are then included as part of the query with their correct spellings.

1.4.5 Determining the natural language after inclusion of misspelled words.

After all of the misspelled words are corrected, those words are once again run against the database for all of the languages and if they are found, then their corresponding frequencies are included in the total frequency count for each of the language that had been calculated previously. In case they are not found in the database of our lexicon for any of the language in the candidate set, they are added to the database of that particular language so that we are able to provide more accurate results for future queries. The new results are analysed for any changes in the initial prediction of the identified language and the language with the updated maximum frequency count is declared as the natural language of the entered text.

1.5. ORGANIZATION

This project report is organized as follows. In Chapter 1 we outline the field of our project that is Natural Language Processing. Also, we throw light upon the challenges faced in identifying the native language of the text and our objective to overcome those. The survey done for this project has been briefed in Chapter 2. The steps followed to develop a system for language identification is discussed in Chapter 3, which also includes our methodology along with some flow charts. Chapter 4 shows the analysis of the system developed and the observed outcomes for various cases of input when run against our system. Concluding comments and future scope can be found in Chapter 5.

Chapter 2
LITERATURE SURVEY

2.1 Word Length Algorithm for language identification of under resourced languages (Nicholas Akosu, Ali Selamat) [1]

For solving the problem of language identification there are many available techniques that require enormous amount of training data but they are not available for under-resourced languages which form the bulk of the languages in the world. Languages that do not have a large repository of digital resources are termed as under resource languages. This research is concentrated on languages with lesser or null digital resources, and so they are named rightly as ‘under-resourced languages’. These are majorly those languages that are spoken by a very few people, but they cannot be neglected as these are acquiring significance due to rapidly increasing use of the Internet. The possibility of such languages to be used for communication over the Internet is also one of the reasons that accounts for its importance.

A lexicon based algorithm that will be able to perform language identification using minimum training data is the first basic objective of this study. Major emphasis has to be put on identifying the language in the shortest possible time and therefore it is important to explore methods that will help in achieving the objective. The effect of the Lexicon based algorithm on the run time performance has become the secondary objective of this research.

Based on the available dataset, the outcome of experiments for language identification at the sentence level as well as for the document level were accurate. On comparing the algorithm with the spelling checker, run time performance improved significantly. The language of the target text is identified by the simple lexicon based technique which compares the words in the document to the vocabulary for any language. That language is concluded as the native language of target document that shares the maximum number of words between its lexicon and the target text.

2.2 A Comparative Study on Language Identification Methods (Lena Grothe, Ernesto William De Luca and Andreas Nurnberger) [2]

This paper focuses on the objective to analyze and compare different language identification methods that have already been implemented with respect to their utility in information retrieval settings and similar domains. The language identification process can be branched majorly into two main steps: In the first step, a model of the document as well as a model for the language is generated for both; in the second step, the language of the document is determined based on the model of the language and it is then added to the document. This work mainly focuses on the significance of a dynamic value for the out-of-place measure.

Short Word-Based Approach: It consists of words up to a fixed length to construct the model of the language. It is entirely different and not at all dependent from the particular word frequency.

Frequent Word-Based Approach: This is a word-based approach. It considers a fixed amount of the words which are most frequent and then generates the language model. The set of words that have the maximum frequency among all the words present in the text are described by these words.

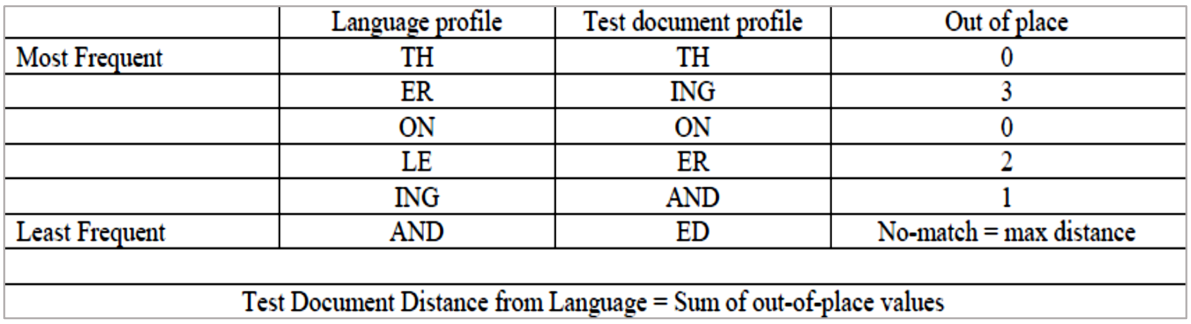
N-Gram-Based Approach: This approach deals with the third type of language model. This model is generated by the n-gram-based approach which takes into account the n-grams of different [11] or fixed [12] lengths of words obtained from tokenizing the words. In comparison [13], generates n-grams considering sequences of bytes instead of the previous discussed approach. A sequence of n characters makes up an n-gram. Firstly, the beginning and end of a word are marked using a special character and then the n-grams are created for that word.
After analysing the results for all the methods mentioned, the conclusion was made that the best results with 99% score was achieved using the first approach which was the frequent word approach.

2.3 Language Identification from Text using N-gram based Cumulative Frequency Addition (Bashir Ahmed, Sung-Hyuk Cha, and Charles Tappert) [3]

This paper focuses on an efficient language classifier that uses an ad-hoc approach of Cumulative Frequency Addition of N-grams. The classification technique discussed here is simpler than the traditional Naïve Bayesian classification method. Its positive side is that it performs with a similar speed but has an improved accuracy on strings which are short in length. When compared to the N-gram based rank-order statistical classifier, this is 5-10 times faster, therefore more efficient. This approach of Language classification based on rank-order statistics based on N-gram comes out to be highly accurate and on addition to it is also insensitive to typographical errors. As a result, this method has been extensively researched in the field of language processing.

However, when rank-order statistics is used for classification, it is found that it is slower than other methods. The reason behind this is the essential requirement of frequency counting and then sorting of N-grams in the document profile under consideration. The most crucial for a classier are its Accuracy and speed of classification which enables it to be useful in an environment of high volume categorization. Thus, it is necessary to look into the performance of the classification methods based on N-grams. Classification speed could be increased substantially if it is possible to eliminate the sorting as well as counting operations in the rank-order statistics methods. The classifier discussed in this paper achieves that goal by using a new method based on the Cumulative Frequency Addition approach.

Methodology: Canvar and Trenkle [11] when rank-order statistics on N-gram profiles, they found the classification rate to be 99.8% on Usenet newsgroup articles written in different languages. They calculated the distances between the N-gram profiles of the test documents and the N-gram profiles for the training language and then choosing the language corresponding to the minimum distance. They had to sort the N-grams for both the training as well as the test profiles before performing the distance measurement. Table below depicts, how the calculation was performed.



	Language profile	Test document profile	Out of place
Most Frequent	TH	TH	0
	ER	ING	3
	ON	ON	0
	LE	ER	2
	ING	AND	1
Least Frequent	AND	ED	No-match = max distance
Test Document Distance from Language = Sum of out-of-place values			

Table 2.3.1. Distance calculation using rank-order statistics.

They encountered two main problems with their classification scheme. Firstly, was the requirements of counting the frequency of each N-gram in the test document and secondly was to sort the N-grams so that distance measurement could be performed. Here a new classifier is introduced using a similar N-gram profile. Its advantage is that it does not require a sort operation on N-gram profiles of both the training and testing data.

Collection of Text Samples and then creating N-gram Profiles from them:

The training sample sizes ranged from sixty-five thousand to a hundred and five thousand. The authors collected 2, 3, 4, 5, 6 and 7 grams from these language samples. Then the occurrence of their counts was calculated and then stored in the database table. After the collection the N-grams was done, some smoothing was done by deleting those words that occurred only once, which greatly reduced the number of N-grams. No pre-processing was performed on these training data.

N-gram Frequency Calculation

After the N-grams which were not necessary were eliminated, Firstly, the processing was done for calculating the total N-gram counts for each language and then the same procedure was applied for the entire training set as well. Internal frequency and the overall frequency for each N-gram were calculated as follows:

$FI (i, j) = C (i, j) / \Sigma\, i\, C (i, j)$

$FO (i, j) = C (i, j) / \Sigma\, i,j\, C (i, j)$

FI (i, j) = Internal frequency of a N-gram i in language j

FO (i, j) = Overall frequency of a N-gram i in language j

C (i, j) = Count of the ith N-gram in the jth language

Σ i C (i, j) = Sum of the counts of all the N-grams in language j

Σ i,j C (i, j) = Sum of the counts of all the N-grams in all the languages

Language	N-gram	N-gram Count	Total N-grams in this Language	Total N-grams in all Languages	Internal Frequency	Overall Frequency
English	xyz	10	100	1000	10/100	10/1000
Danish	xyz	15	150	1000	15/150	15/1000
French	xyz	10	200	1000	10/200	10/1000
Italian	xyz	15	300	1000	15/300	15/1000
Tagalog	xyz	3	40	1000	3/40	3/1000
German	xyz	13	60	1000	13/60	13/1000
Spanish	xyz	28	150	1000	28/150	28/1000

Table 2.3.2 Sample calculation using a hypothetical N-gram "xyz"

Length of String Tested	Percent Correct Rank Order Statistics (All languages)	Percent Correct Cumulative Frequency Sum (All languages)	Percent Correct Naïve Bayesian Classifier (All languages)
50	96.56	97.59	88.66
100	98.97	98.97	96.90
150	100.00	100.00	99.70

Table 2.3.3 The number of files tested in each language.

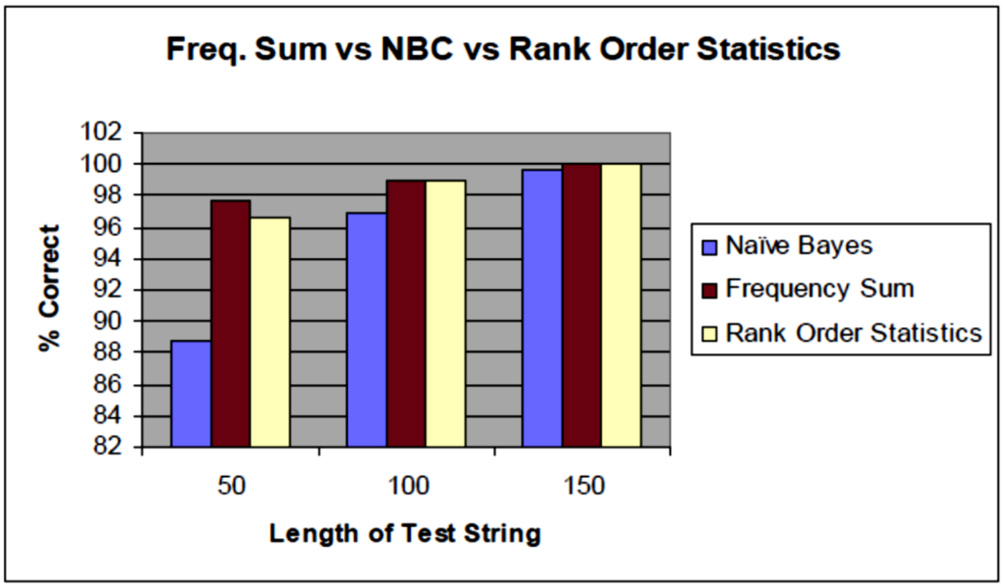


Figure 2.3.1 Percent accuracy of classification of NBC, CFA, and rank-order statistics.

Summary of Results: Table 2.3.3 depicts the no. of files tested in each language alongside with their % accuracy obtained in all of the three methods. Figure 2.3.1 shows the results in from fifty, hundred, and hundred and fifty byte strings using the three classification methods.

Conclusion: The key highlights of both the cumulative frequency addition and the Naïve Bayesian classification methods are the speed with which the classification is done. In contrast, the strength of the rank-order statistics method is its precision and accuracy. When talking about short strings, the CFA method and the rank-order statistics method are comparable in terms of accuracy.

When processing long text strings for identifying the major language, the speed of the new approach plays a very important role. It is also useful when dealing with strings with shorter length for calculating the minor part. In the Naïve Bayesian classification, an assumption is taken that the probabilities of unique N-grams are not dependent on the other N-grams. This is not a good assumption, as it has high chances of introducing errors in the system.

2.4 Automatic Language Identification for Romance Languages using Stop Words and Diacritics (Ciprian-Octavian Truica, Julien Velcin, Alexandru Boicea) [4]

This paper presented a statistical method for identification of language of written text by utilizing the dictionaries containing stop words and diacritics. The authors propose varied approaches that included combining the two dictionaries to accurately determine the language of text under consideration. This method was selected because stop words and diacritics belongs to a particular language, although some languages do share some similarities in words and special characters, these words are not at all common. The languages taken into consideration were romance languages because they show a high similarity factor and usually it is a difficult task to distinguish between them from the view of a computational process. They have tested their method using corpus obtained from Twitter as well as from the articles from the news. Both corpora consist of text encoded in UTF-8 format, so the diacritics could be easily processed. When there is such a case that the text contains no diacritics in it, the language of the text is determined by using only the stop words. The results obtained after successfully completing the experiments suggest that the proposed method has achieved an accuracy of over 90% in the case of small texts and over 99.8% when considering large texts.

Stop words and diacritics prove to be very useful when dealing with the process of automatic identification of language. Some words are particular only to a specific language and the processing can be skipped completely if those word are directly found. The length of the text of which the language has to be determined also has a considerable impact on the overall accuracy as it provides more information to the process of identification. In conclusion, when considering both the cases of short and long texts, stop words provide a good basis for identification of the language but the accuracy of the system can be improved to a good state by utilizing the diacritics in the language. Based on the experimental result, they come to the conclusion that the dictionary for the stop words must be built strongly in order to remove the errors and miss-classification of the words.

In future work, the parameter p will be in the limelight so that it is fine tuned in a manner which can help in achieving an increased accuracy. More testing need to done and this approach will be applied on other Romance languages (e.g. Occitan, Catalan, Venetian, Aromanian, Galician etc.) as well as on some other languages belonging to the Indo- Europena families such as Germanic and Slavic. In order to improve the architecture of the system, work need to be done on parallelizing the algorithm to a much greater extent to reach real-time performance.

2.5 VarClass: An Open Source Language Identification Tool for Language Varieties (Marcos Zampieri, Binyam Gebrekidan Gebre) [5]

This paper presents VarClass which is a tool for identifying the language which is also open source. It is easily available, both to be downloaded as well as exploring it through a graphical user-friendly interface. The key point that differentiates this approach to the other existing and currently on-going processes is its emphasis on dealing with the variety of languages [14]. The traditional language identification tools do not consider language varieties and therefore the main aim of this paper is to fill that gap. VarClass presently contains language models for over twenty-seven languages which also includes 10 languages that are varieties. When differentiated the 27 classes considered here, trigrams for the character was made the basis and a performance of 90.5% F-Measure was achieved. For evaluation, VarClass considered test set containing fifty-four hundred documents that belonged to a wide variety of languages consisting of about two hundred documents belonging to each class. The algorithm used in VarClass takes a lot of hints from the algorithm described in Zampieri and Gebre [14] and adopts it in a useful manner which was later tested in different scenarios. The algorithm consisted of a simple likelihood function which was calculated over language models which were smoothened beforehand. The models for these languages can be obtained by using words, characters or even POS categories The calculation for the likelihood function is done as described in Figure 2.5.1

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

Figure 2.5.1 Equation for likelihood function.

Here, N is the count of n-grams present in the test text, ni is the ith n-gram and L holds the value for the language models. The probability for each of the language models is calculated. The language model with highest probability determines the identified language of the text

In conclusion, this paper presented a new resource, the VarClass language identification tool. This work is a first step towards the evaluation of language varieties into broader language identification settings. It is believed that this tool fills an important gap among other language identification tools that do not consider language varieties. The tool can be used by linguists and computational linguists interested in language identification, contrastive linguistics as well as by other users not related to the NLP and linguistics research community.

2.6 Language Identification of Short Text Segments with N-gram Models (Tommi Vatanen, Jaakko J. V¨ayrynen, Sami Virpioja) [6]

The task of identifying natural language of any document which contains samples of long texts can be accomplished using various number of accurate techniques. But the same task is challenging in case of short text samples. The research of this paper revolves around this problem and the sample text of 5-21 characters are takin into account.

The two different methods: A Naïve Bayesian classifier which is based on character N-grams and the other one, ranking method proposed by Canvar and Trenkle [11], are compared to achieve the task. Many smoothening techniques are tested along with the state-of-the-art method namely Kneser-Ney interpolation, for the n-gram models. The results were also compared, using subset of 50 languages, with the identifier provided by the Google AJAX Language API.

The fact that identification of language is being carried out for at least several words becomes its uniqueness. Also, word boundaries were not taken into notable consideration. The Universal Declaration of Human Rights was taken as a corpus and the experiments were performed considering 281 languages. Furthermore, n-gram smoothening and model pruning techniques that are important for improvement are not included.

- **Methods**

The task of identifying language is sub-divided into two steps, mainly language modelling and its classification. In language modelling, the models of the languages are constructed independently without considering the next classification process. In the step of classification, the classifier has a crucial role. Various machine learning techniques, such as, support vector machines Kruengkrai et al [15], normalized dot product Damashek [16], k-nearest neighbour and relative entropy Sibun and Reynar [17], are applied in language detection. Some other applied techniques also include neural networks, multiple linear regression Botha and Barnard [18] and decision trees.

The following are the methods applied in this experiment (i.e. ranking method and the classification based on n-gram model):

- **Ranking Method for Language Identification**

Canvar and Trenkle [11] proposed the ranking method which is useful in text categorization and language identification. In this method, the profiles specific to a language and contains the most frequent character N-grams from the training corpus and these are sorted according to frequencies of the N-grams. Figure 2.6.1 demonstrates the ranking method. The same kind of text profile is created from the classified text and they are also sorted on the basis of frequencies of N-grams. An "out-of-place" measure between the text profile and the language profile of each language is cumulatively computed which determines how an n-gram in one profile is far out-of-place from its place in the other profile. The maximum distance is obtained when a n-gram of text profile is not found in the language profile and this is equal to the number of n-grams present in the language profile. For the case where input contains short text, low number of n-grams are contained in the text profile as compared to the language profile. Thus, to avoid the case where language profile n-grams are not present in the text profile, the experiments were performed with a revised ranking method.

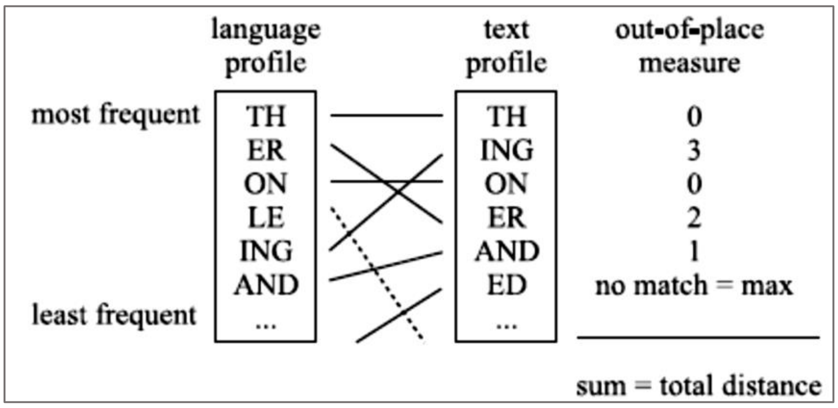
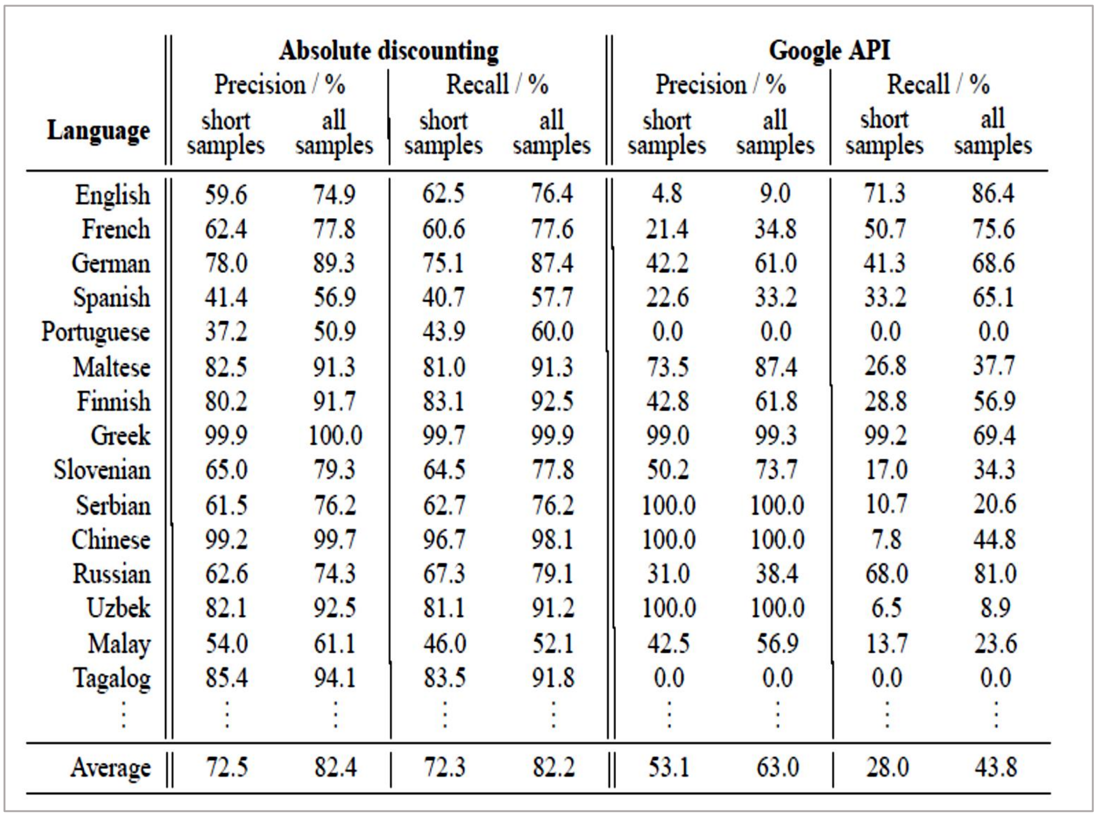


Figure 2.6.1. A demonstration of the rank order method.

- **Language Identification with N-gram Models**

An n-gram model elucidates a probability distribution method in which the probability of an observation, mostly a word or a character, is presumed to depend on the last (n-1) observations only. The probabilities with paramount similarity tend to overlearn the training data. The improvement in these estimates can be done by smoothening methods listed below. With these techniques, the probability mass from the rarely occurring n-grams can be moved to those that do not occur at all.

→ Additive Smoothing
→ Katz Smoothing
→ Absolute Discounting
→ Kneser-Ney Smoothing
→ Model Pruning



Language	Absolute discounting				Google API			
	Precision / %		Recall / %		Precision / %		Recall / %	
	short samples	all samples	short samples	all samples	short samples	all samples	short samples	all samples
English	59.6	74.9	62.5	76.4	4.8	9.0	71.3	86.4
French	62.4	77.8	60.6	77.6	21.4	34.8	50.7	75.6
German	78.0	89.3	75.1	87.4	42.2	61.0	41.3	68.6
Spanish	41.4	56.9	40.7	57.7	22.6	33.2	33.2	65.1
Portuguese	37.2	50.9	43.9	60.0	0.0	0.0	0.0	0.0
Maltese	82.5	91.3	81.0	91.3	73.5	87.4	26.8	37.7
Finnish	80.2	91.7	83.1	92.5	42.8	61.8	28.8	56.9
Greek	99.9	100.0	99.7	99.9	99.0	99.3	99.2	69.4
Slovenian	65.0	79.3	64.5	77.8	50.2	73.7	17.0	34.3
Serbian	61.5	76.2	62.7	76.2	100.0	100.0	10.7	20.6
Chinese	99.2	99.7	96.7	98.1	100.0	100.0	7.8	44.8
Russian	62.6	74.3	67.3	79.1	31.0	38.4	68.0	81.0
Uzbek	82.1	92.5	81.1	91.2	100.0	100.0	6.5	8.9
Malay	54.0	61.1	46.0	52.1	42.5	56.9	13.7	23.6
Tagalog	85.4	94.1	83.5	91.8	0.0	0.0	0.0	0.0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
Average	72.5	82.4	72.3	82.2	53.1	63.0	28.0	43.8

Table. 2.6.1 Values of precision and recall in case of absolute discounting with n = 5 and Google AJAX Language API for chosen languages for all lengths of test sample.

Comparison with the Google API: The smaller language sets were used in experiments with the Google AJAX language API. The results showed overall weak accuracy in identification process, in between 20.8% and 59.9%, when different lengths of sample texts were considered. In contrast, accuracy in range 64.1% and 90.7% was acquired when the best n-gram models were used in experimentation.

Conclusion – As mentioned earlier that having a good method for identification of language in case of short texts presented a challenge as compared to long text samples, this research came out with some conclusions. The absolute discounting method, which is one of the advanced smoothening techniques, is more accurate when compared to the ranking method in the process of identifying language in case of short text samples. The accuracy obtained at the cost of slower classification speed and larger model is higher.

2.7 Improved N-grams Approach for Web Page Language Identification (Ali Selamat) [7]

For Web Page language identification, an improved n-gram approach which is based on the combination of original n-gram method and a modified n-gram method, is proposed by the authors of this paper. Also, the experiments showed that the improved n-gram approach [25] is able to identify the language of web pages with more improvement, mainly for the scripts written in Roman and Arabic languages.

The original n-gram approach is different from the modified n-gram approach in a way that original n-grams approach uses distance measurement while the modified n-gram approach is based on Boolean matching rate. The improved n-gram approach based on the combination of these two becomes the uniqueness for this research paper because it uses both n-grams frequency and n-gram position to find the languages of web page scripts written in Roman and Arabic.

Language identification of web pages is a challenging task due to the everyday rising number of web documents and users of the web services [9][10].
Conclusion - The improved n-gram approach after being compared to the original n-gram and modified n-gram approach has been able to determine language of the web pages with accuracy.

Future Scope – Due to the limitations of computational facilities, the parameters like cost of computation and scalability have been left to address in the future, so that the reliability of proposed approach can be increased.

2.8 Categorizing Unknown Words: Using Decision Tress to Identify Names and Misspellings (Janine Toole) [8]

This paper addresses the problem of encountering unknown words in any natural language processing system. The performance of system degrades when larger number of unknown words are found in language identification. The system provided in this paper is based on multiple component architecture in which every component identifies a particular class of unknown words.

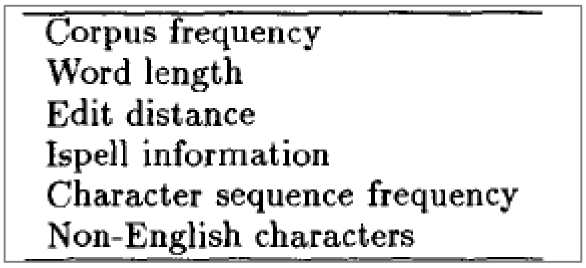
The paper emphasises the research on only two components: the first component identifies names and the second component identifies the spelling errors. A common approach using decision tree architecture is being used by both the components. The results from both the identifiers are combined using a weighted voting process. The following are the two components: The Name Identifier and The Misspelling Identifier.

- The Name Identifier Component

A Name in this context can be defined as any word identifying a place, person or a concept which when written in English language requires capitalization. This identifier distinguishes those unknown words that are proper names, and those that are not.

- The Misspelling Identifier Component

This component works in a way that it segregates the spelling errors from the correct words. In this, a word is considered to be a misspelling when it differs from the correctly written word by any one operation - addition, subtraction, deletion, or reversal of letters, or if any punctuation is excluded. The list of features used to categorize misspellings from the training set are depicted in Table 2.8.1.

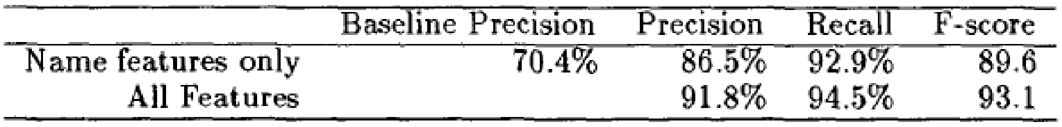


Corpus frequency
Word length
Edit distance
Ispell information
Character sequence frequency
Non-English characters

Table 2.8.1. Features used in misspelling decision tree

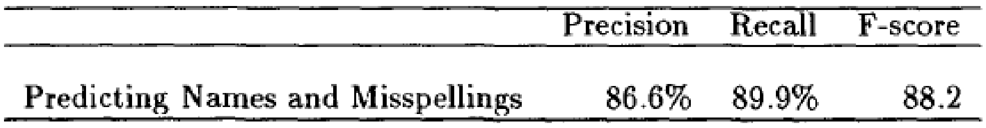
Decision making component: Both the identifiers, i.e. misspelling identifier and the name identifier returns a prediction factor for unknown words. The prediction is said to be compatible if misspelling identifier says it's not a misspelling and the name identifier predicts the unknown word to be a name. In this case the unknown word can be clearly decided to be a name error. Therefore, in the case where one identifier makes a positive prediction while the other makes a negative prediction, decision making is straightforward. Similarly, if both trees make negative predictions, then it can be presumed that the unknown word is neither a misspelling nor a name.
Although, it is also possible that both the tress makes positive predictions. This means the name identifier says it's a name and the misspelling identifier says it is a spelling error. In such cases, a decisive measure should be chosen to help in finding which decision tree has to be accepted. Since the decision trees return a confidence measure for each leaf of the tree, this measure is computed from a specific leaf in the training set. The proportion of right predictions over the total number of predictions at that leaf node is considered.

The following figures shows the experimental results for precision and recall for both the components.



	Baseline Precision	Precision	Recall	F-score
Name features only	70.4%	86.5%	92.9%	89.6
All Features		91.8%	94.5%	93.1

Figure 2.8.1. Precision and recall for name identification



	Precision	Recall	F-score
Predicting Names and Misspellings	86.6%	89.9%	88.2

Figure 2.8.2. Precision and recall for decision-making component

Conclusion: The unknown word categorizer consisting of name and misspelling identifier are implemented using a component viz. decision tree architecture. The encouraging results were shown by the system when the evaluation was done against the transcripts obtained from live closed captions.

Chapter 3
SYSTEM DEVELOPMENT

Our proposed system for Language Identification of text includes the following steps:

i. Collect raw data to build up the training set for out database from web scrapping news articles, news feed, RSS feed etc., corresponding to different languages.
For our work, we collected the data from The Leipzig Corpora Collection [23] [24] that provides tools and data for download in different languages using the same format and comparable sources. The languages included in our system for now are English, German[21], Russian [22], Latin, French [19], and Spanish [20].

ii. Extract and tokenize the words from the unprocessed data files of each language.

iii. Calculate the overall frequency count for each of the words in all the languages which are part of our candidate set.

iv. Store the data into respective databases of languages to develop our Lexicon.

v. Remove any fallacy or redundant entries from the database, such as Junk characters, indentation spaces and words with other discrepancies.

The database is created for each language. Now in order to detect the language:

vi. Obtain the query string of which the language is to be determined.

vii. Tokenize the query string into words.

viii. These words are run against the database for all of the languages and total frequency count for each of the language is calculated by summing up the frequencies of all the words that occurred in each of the language.

ix. Compare the results and output the identified language.

x. Certain words that are not present in the database for any of the language are firstly labelled as unknown words.

xi. Unknown words are run against the dictionaries for all of the languages in the candidate set and the words which find a match are added to the database of that particular language. If those words are not found in the dictionary of any of the language, they are categorized as misspelled words and are separated out for further processing.

xii. Misspelled words are tackled using Damearu-Lavenshtein algorithm and pythons enchant library to find the correct suggestions.

xiii. Corrected words are once again run against the database for all of the languages and if they are found, then their corresponding frequencies are included in the total frequency count.

xiv. In case they are not found in the database of our lexicon for any of the language in the candidate set, they are added to the database of that particular language so that we are able to provide more accurate results for future queries.

xv. The language with the updated maximum frequency count is declared as the natural language of the entered text.

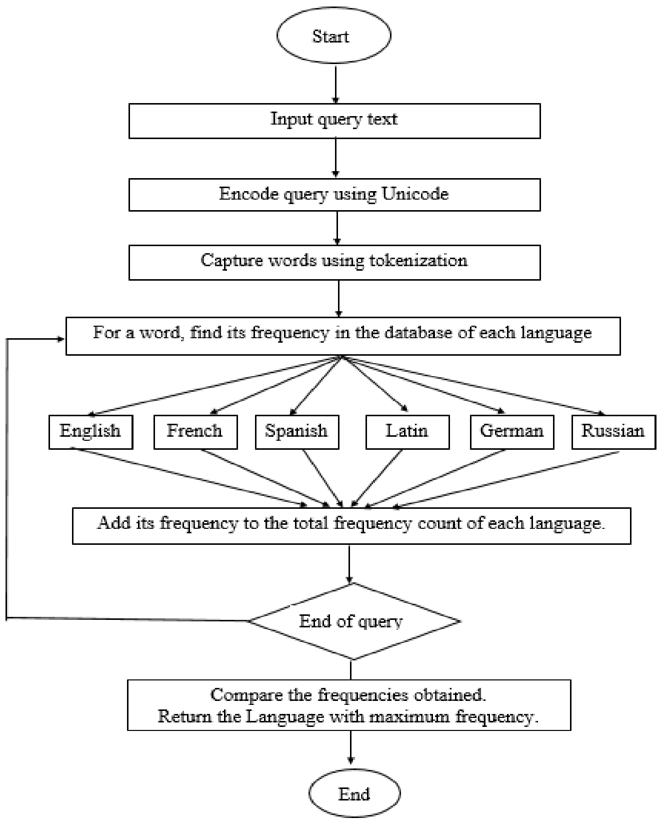

Start
Input query text
Encode query using Unicode
Capture words using tokenization
For a word, find its frequency in the database of each language
English
French
Spanish
Latin
German
Russian
Add its frequency to the total frequency count of each language.
End of query
Compare the frequencies obtained.
Return the Language with maximum frequency.
End

Fig 3.1 A Flow Chart representation of initial process of language identification.

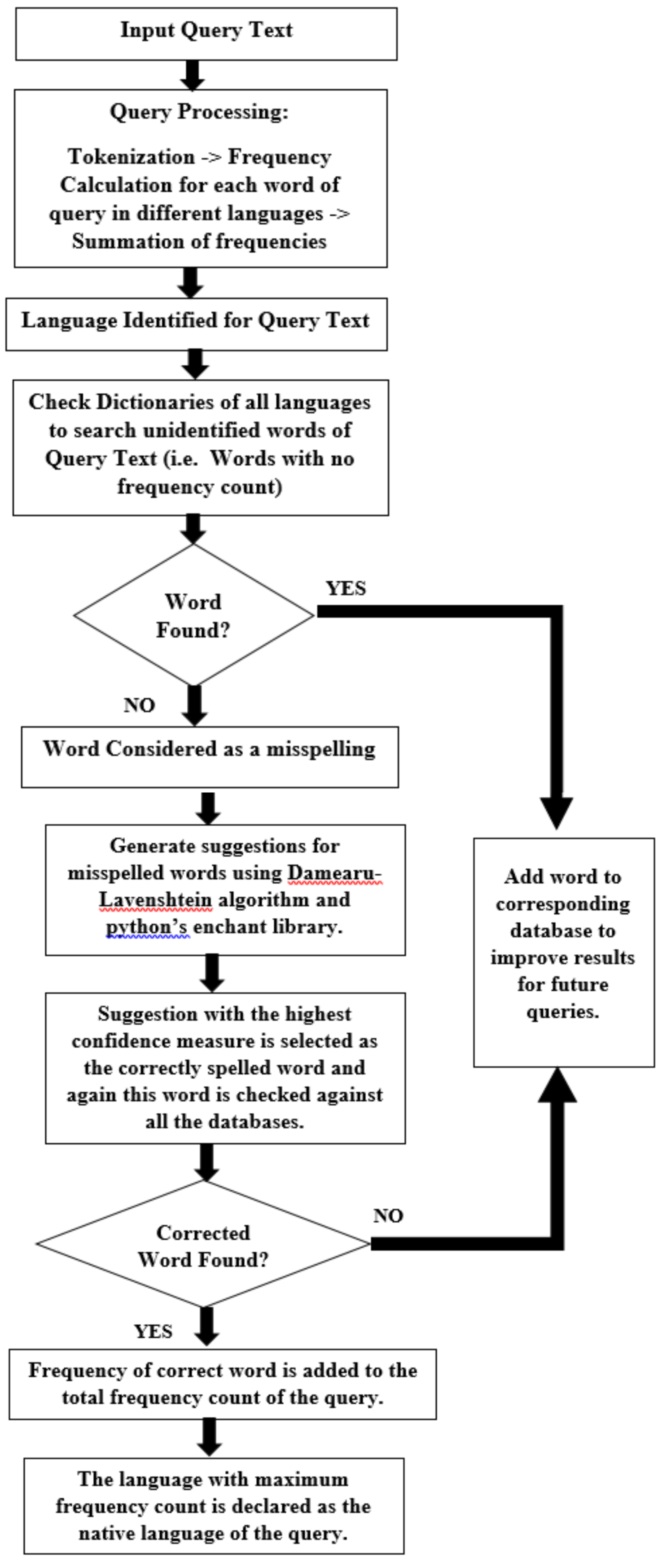
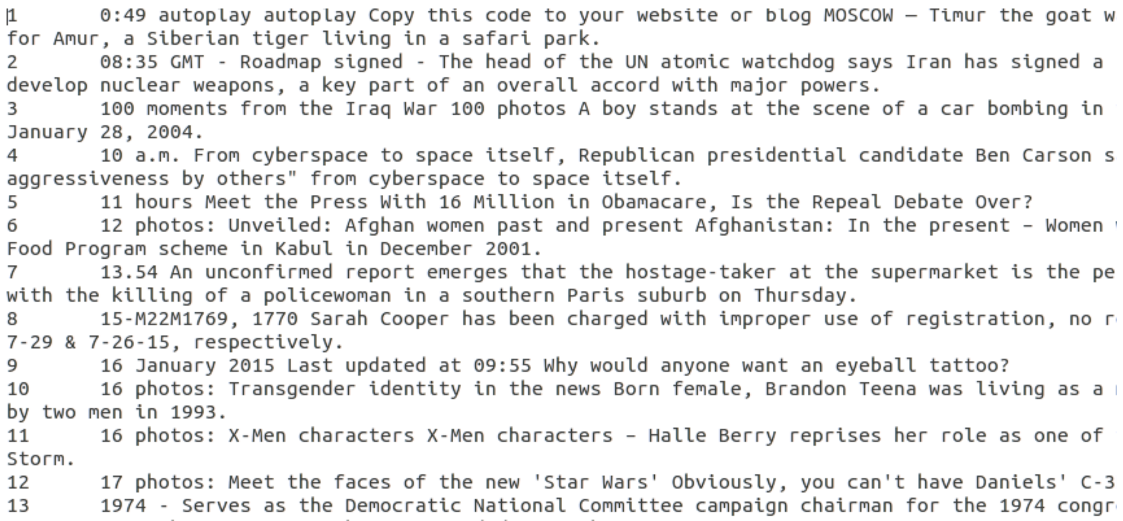

Input Query Text
Query Processing:
Tokenization -> Frequency Calculation for each word of query in different languages -> Summation of frequencies
Language Identified for Query Text
Check Dictionaries of all languages to search unidentified words of Query Text (i.e. Words with no frequency count)
Word Found?
YES
NO
Word Considered as a misspelling
Generate suggestions for misspelled words using Damearu-Lavenshtein algorithm and python's enchant library.
Suggestion with the highest confidence measure is selected as the correctly spelled word and again this word is checked against all the databases.
Corrected Word Found?
NO
YES
Add word to corresponding database to improve results for future queries.
Frequency of correct word is added to the total frequency count of the query.
The language with maximum frequency count is declared as the native language of the query.

Fig 3.2 The summarized process of Identification of Language with inclusion of misspelled words.

CHAPTER 4
PERFORMANCE ANALYSIS

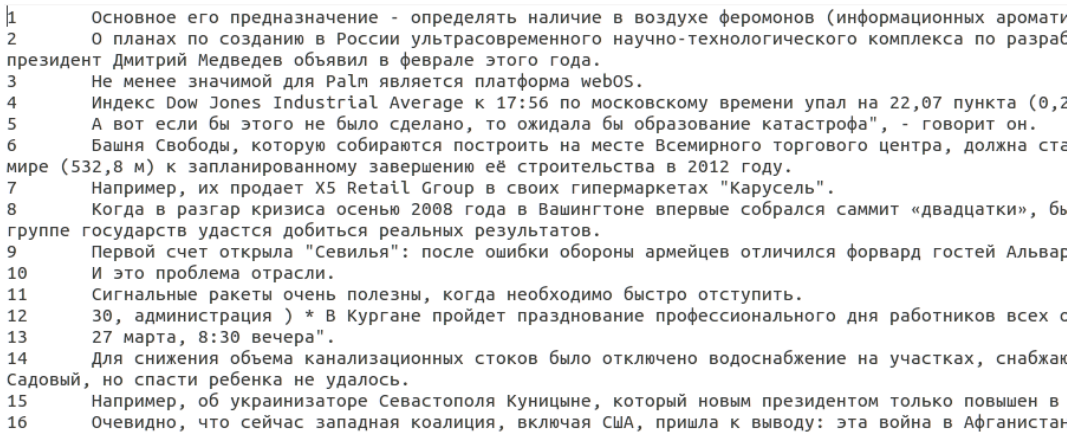
For developing a basic system at first, the languages we have chosen are English, Russian, French, Spanish, Latin and German. The first step is to build a database of training set in order to develop the Lexicon of our languages that have been included in our project at present.

Raw files are collected from articles, news feed, RSS feeds and web pages in these languages that are present on the internet.



```
1       0:49 autoplay autoplay Copy this code to your website or blog MOSCOW — Timur the goat w
for Amur, a Siberian tiger living in a safari park.
2       08:35 GMT - Roadmap signed - The head of the UN atomic watchdog says Iran has signed a
develop nuclear weapons, a key part of an overall accord with major powers.
3       100 moments from the Iraq War 100 photos A boy stands at the scene of a car bombing in
January 28, 2004.
4       10 a.m. From cyberspace to space itself, Republican presidential candidate Ben Carson s
aggressiveness by others" from cyberspace to space itself.
5       11 hours Meet the Press With 16 Million in Obamacare, Is the Repeal Debate Over?
6       12 photos: Unveiled: Afghan women past and present Afghanistan: In the present – Women
Food Program scheme in Kabul in December 2001.
7       13.54 An unconfirmed report emerges that the hostage-taker at the supermarket is the pe
with the killing of a policewoman in a southern Paris suburb on Thursday.
8       15-M22M1769, 1770 Sarah Cooper has been charged with improper use of registration, no r
7-29 & 7-26-15, respectively.
9       16 January 2015 Last updated at 09:55 Why would anyone want an eyeball tattoo?
10      16 photos: Transgender identity in the news Born female, Brandon Teena was living as a
by two men in 1993.
11      16 photos: X-Men characters X-Men characters – Halle Berry reprises her role as one of
Storm.
12      17 photos: Meet the faces of the new 'Star Wars' Obviously, you can't have Daniels' C-3
13      1974 - Serves as the Democratic National Committee campaign chairman for the 1974 congr
```

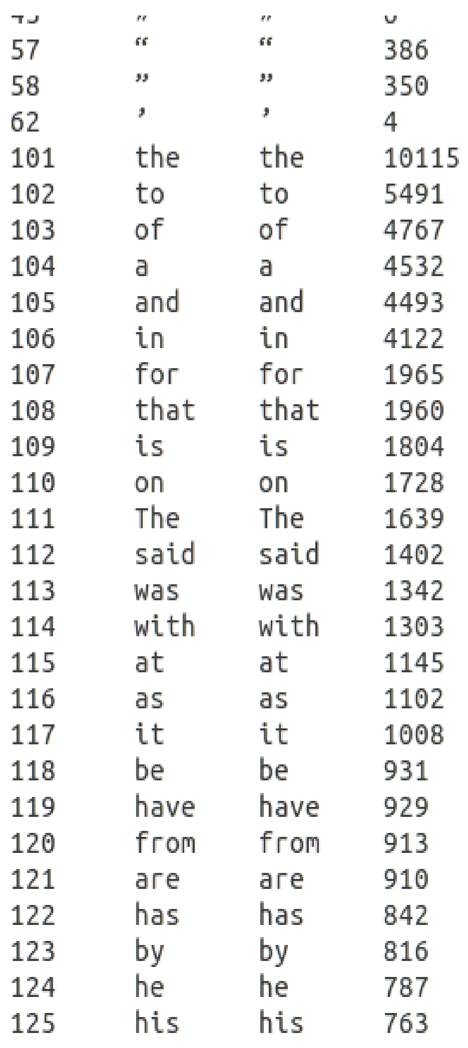
Fig 4.1 Raw Data from the web in English.



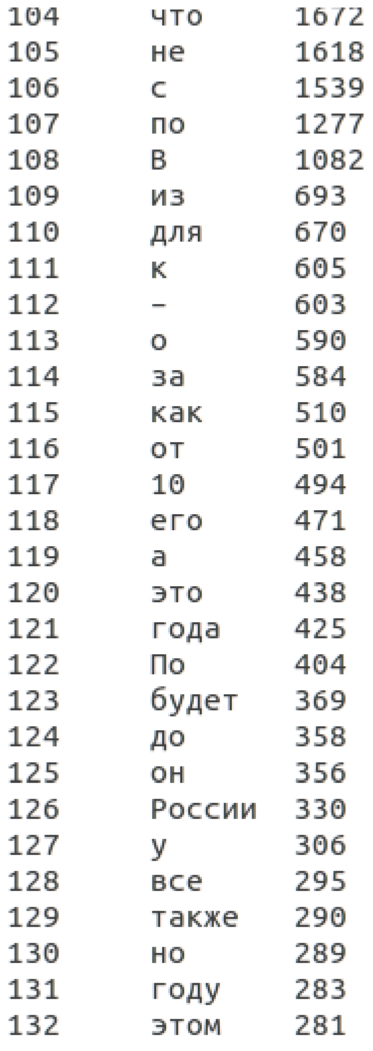
```
1       Основное его предназначение - определять наличие в воздухе феромонов (информационных аромати
2       О планах по созданию в России ультрасовременного научно-технологического комплекса по разраб
президент Дмитрий Медведев объявил в феврале этого года.
3       Не менее значимой для Palm является платформа webOS.
4       Индекс Dow Jones Industrial Average к 17:56 по московскому времени упал на 22,07 пункта (0,2
5       А вот если бы этого не было сделано, то ожидала бы образование катастрофа", - говорит он.
6       Башня Свободы, которую собираются построить на месте Всемирного торгового центра, должна ста
мире (532,8 м) к запланированному завершению её строительства в 2012 году.
7       Например, их продает X5 Retail Group в своих гипермаркетах "Карусель".
8       Когда в разгар кризиса осенью 2008 года в Вашингтоне впервые собрался саммит «двадцатки», бь
группе государств удастся добиться реальных результатов.
9       Первой счет открыла "Севилья": после ошибки обороны армейцев отличился форвард гостей Альвар
10      И это проблема отрасли.
11      Сигнальные ракеты очень полезны, когда необходимо быстро отступить.
12      30, администрация ) * В Кургане пройдет празднование профессионального дня работников всех с
13      27 марта, 8:30 вечера".
14      Для снижения объема канализационных стоков было отключено водоснабжение на участках, снабжак
Садовый, но спасти ребенка не удалось.
15      Например, об украинизаторе Севастополя Куницыне, который новым президентом только повышен в
16      Очевидно, что сейчас западная коалиция, включая США, пришла к выводу: эта война в Афганистан
```

Fig 4.2 Raw Data from the web in Russian.

Then all these Raw files obtained from the web are tokenized and are stored in the respective databases of our training set after calculating their frequency of occurrence in that particular language.



[illegible]	″	″	[illegible]
57	“	“	386
58	”	”	350
62	’	’	4
101	the	the	10115
102	to	to	5491
103	of	of	4767
104	a	a	4532
105	and	and	4493
106	in	in	4122
107	for	for	1965
108	that	that	1960
109	is	is	1804
110	on	on	1728
111	The	The	1639
112	said	said	1402
113	was	was	1342
114	with	with	1303
115	at	at	1145
116	as	as	1102
117	it	it	1008
118	be	be	931
119	have	have	929
120	from	from	913
121	are	are	910
122	has	has	842
123	by	by	816
124	he	he	787
125	his	his	763

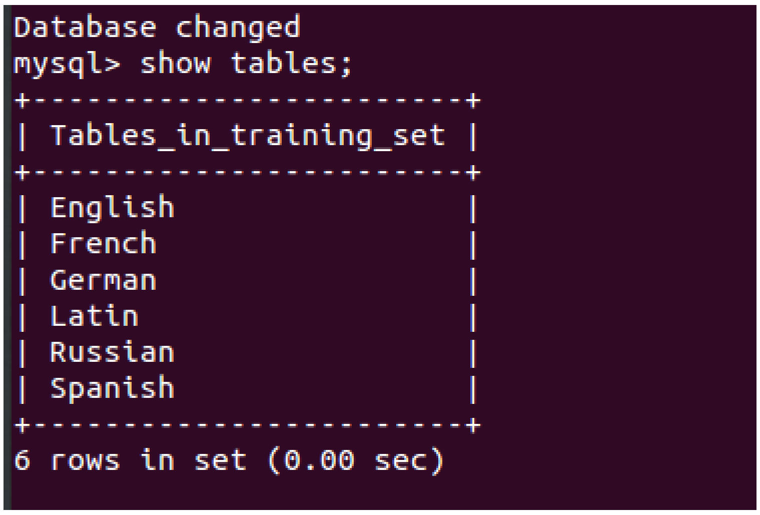


104	что	1672
105	не	1618
106	с	1539
107	по	1277
108	в	1082
109	из	693
110	для	670
111	к	605
112	–	603
113	о	590
114	за	584
115	как	510
116	от	501
117	10	494
118	его	471
119	а	458
120	это	438
121	года	425
122	По	404
123	будет	369
124	до	358
125	он	356
126	России	330
127	у	306
128	все	295
129	также	290
130	но	289
131	году	283
132	этом	281

Fig 4.3 Tokenized words with frequency from English.

Fig 4.4 Tokenized words with frequency from Russian.

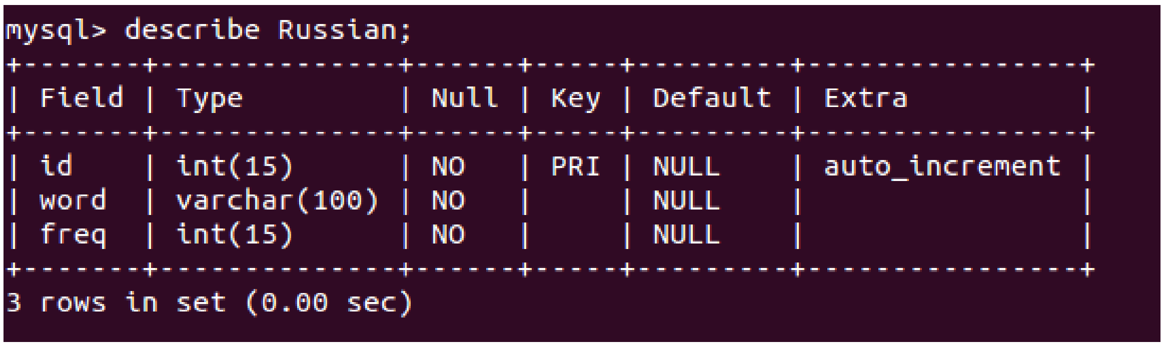
The same method is applied to Raw unprocessed files of all the languages and in the end we get our training set.



```
Database changed
mysql> show tables;
+-----------------------+
| Tables_in_training_set |
+-----------------------+
| English                |
| French                 |
| German                 |
| Latin                  |
| Russian                |
| Spanish                |
+-----------------------+
6 rows in set (0.00 sec)
```

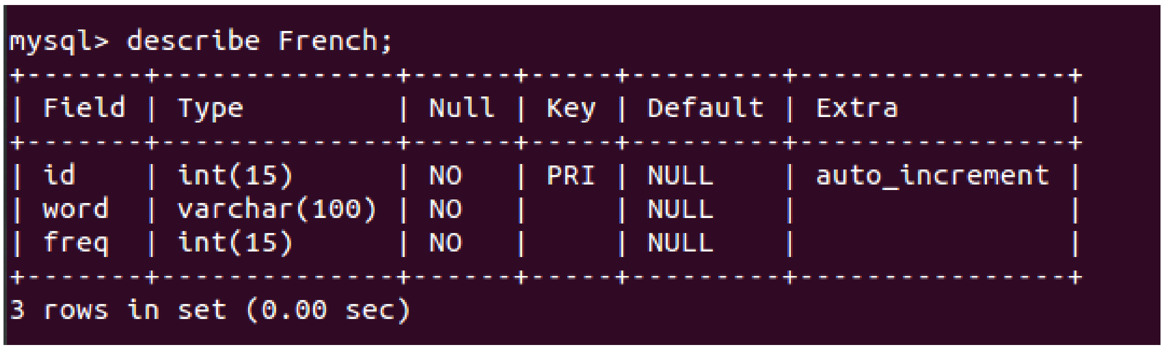
Fig. 4.5 List of Tables in our training set.

Here is a description of the layout of tables in our dataset. All the tables in our training set follow the same configuration. The Describe command of the SQL language provides information about the columns of the table.



```
mysql> describe Russian;
+-------+--------------+------+-----+---------+----------------+
| Field | Type         | Null | Key | Default | Extra          |
+-------+--------------+------+-----+---------+----------------+
| id    | int(15)      | NO   | PRI | NULL    | auto_increment |
| word  | varchar(100) | NO   |     | NULL    |                |
| freq  | int(15)      | NO   |     | NULL    |                |
+-------+--------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

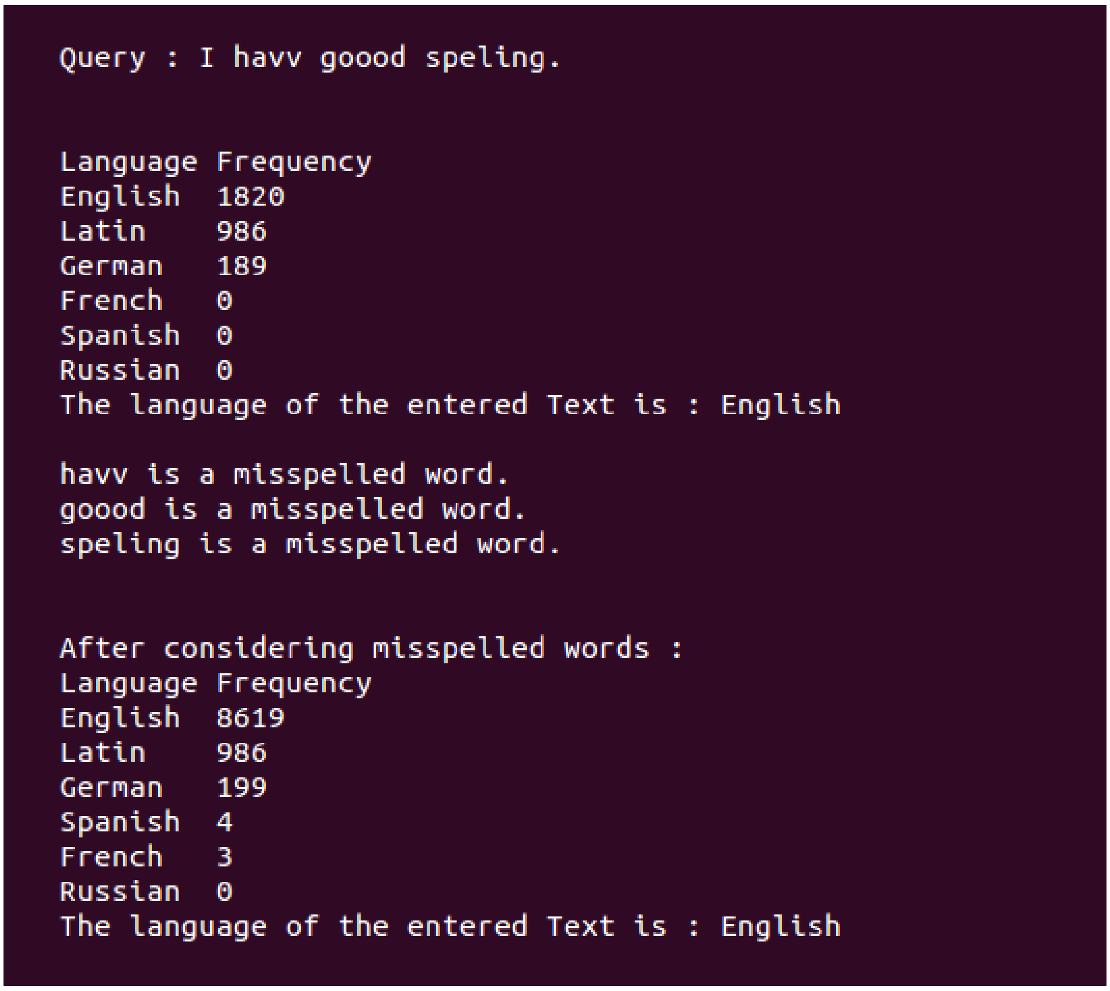
Fig 4.6 Description of the Russian table.



```
mysql> describe French;
+-------+--------------+------+-----+---------+----------------+
| Field | Type         | Null | Key | Default | Extra          |
+-------+--------------+------+-----+---------+----------------+
| id    | int(15)      | NO   | PRI | NULL    | auto_increment |
| word  | varchar(100) | NO   |     | NULL    |                |
| freq  | int(15)      | NO   |     | NULL    |                |
+-------+--------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

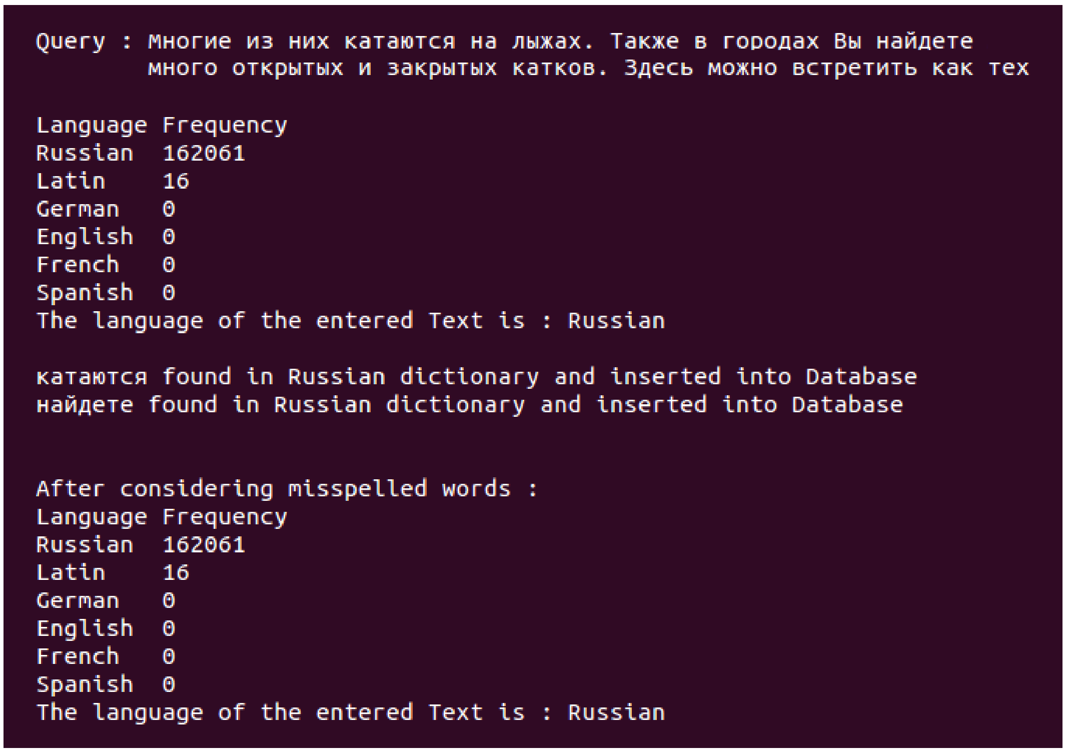
Fig 4.7 Description of the French table.

Now we process the query string, of which the language needs to be identified. It is tokenized into words and then all the words are run against the database of all languages. Probable language is determined by combining the frequencies of all the words of the query text and their occurrence all together and native language that is identified for the query text is displayed. Words that were not present in the database of any of the language are first looked up in the dictionary and if found, are added to the database. The remaining words will be misspelled words. A list of possible suggestions for a particular word is generated for each of the language and then by using the Pythons' enchant library, we analyze the confidence factor for all of the suggestions and then the one with the highest value, is declared as the word that has the most correct form of that particular misspelled word and then its frequency is added to the previously calculated results.



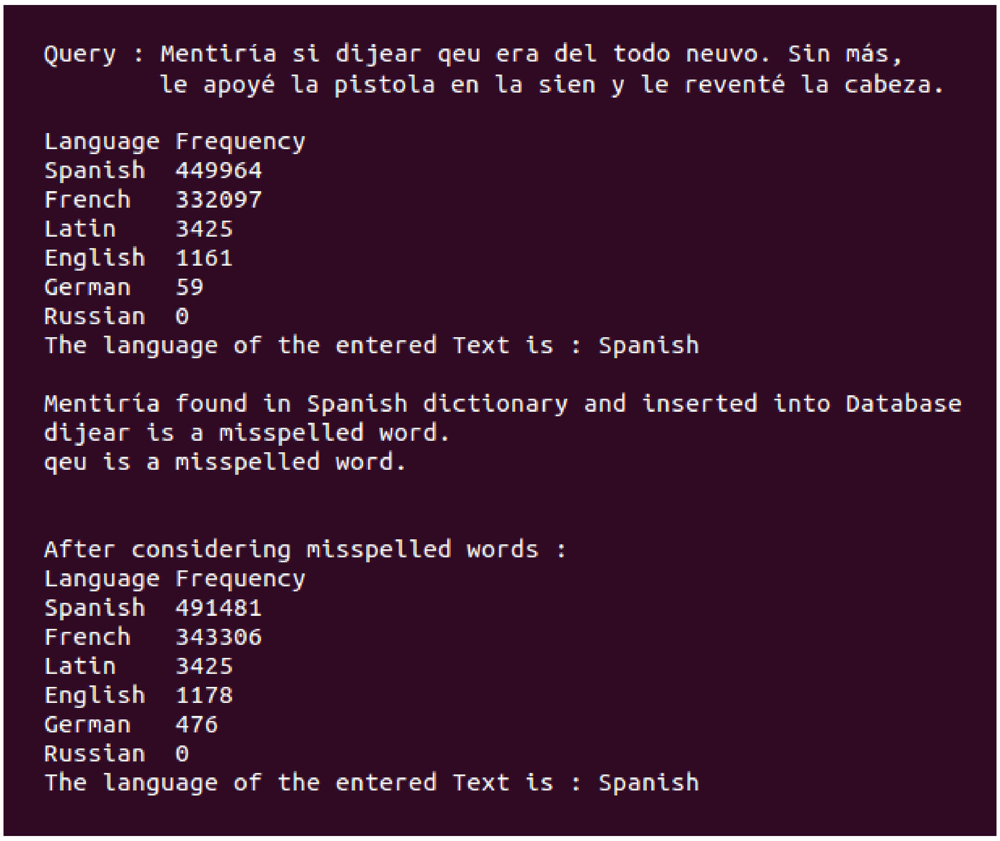
```
Query : I havv goood speling.


Language Frequency
English  1820
Latin    986
German   189
French   0
Spanish  0
Russian  0
The language of the entered Text is : English

havv is a misspelled word.
goood is a misspelled word.
speling is a misspelled word.


After considering misspelled words :
Language Frequency
English  8619
Latin    986
German   199
Spanish  4
French   3
Russian  0
The language of the entered Text is : English
```

Fig. 4.8 Result for sample text entered in English.



```
Query : Многие из них катаются на лыжах. Также в городах Вы найдете
        много открытых и закрытых катков. Здесь можно встретить как тех

Language Frequency
Russian  162061
Latin    16
German   0
English  0
French   0
Spanish  0
The language of the entered Text is : Russian

катаются found in Russian dictionary and inserted into Database
найдете found in Russian dictionary and inserted into Database


After considering misspelled words :
Language Frequency
Russian  162061
Latin    16
German   0
English  0
French   0
Spanish  0
The language of the entered Text is : Russian
```

Fig. 4.9 Result for sample text entered in Russian.



```
Query : Mentiría si dijear qeu era del todo neuvo. Sin más,
        le apoyé la pistola en la sien y le reventé la cabeza.

Language Frequency
Spanish  449964
French   332097
Latin    3425
English  1161
German   59
Russian  0
The language of the entered Text is : Spanish

Mentiría found in Spanish dictionary and inserted into Database
dijear is a misspelled word.
qeu is a misspelled word.


After considering misspelled words :
Language Frequency
Spanish  491481
French   343306
Latin    3425
English  1178
German   476
Russian  0
The language of the entered Text is : Spanish
```

Fig. 4.10 Result for sample text entered in Spanish.

Fig. 4.8 shows the results when the query was entered and the system identified the language to be English. Initially only a single word was correctly spelled and therefore we obtained a low frequency count for the English language. The remaining three words “havv”, “goood" and “speling” were identified as misspelled and then their correct spelling is judged by the system. Frequencies of these words are then added on to the previous results thus increasing the total frequency count by a larger factor.

Fig. 4.9 shows the results when the query was entered and the system identified the language to be Russian. Due of its unlikeliness with other languages, the query string was perspicuously identified rightly. The resemblance with any other language is negligible which is supported by the high cumulative frequency count. There were two unknown words encountered which were first searched against the database and when not found were looked into the respective dictionary. Both the words were updated to the Russian database of our system, with a new constant frequency of one, hence improving the system performance for future queries.

Fig. 4.10 illustrates the case when the query was entered and the system identified the language to be Spanish. Also, it found three unknown words, out of which one was successfully found in Spanish dictionary while the other two were categorized as misspelled words. These words were then tackled using Damerau-Levenshtein algorithm and out of all the suggestions obtained, the correct word was decided by using Pythons’ Enchant library that had the highest confidence measure. The frequency for these words were looked up in the database and added on to the previous results. As we can see that there has been an increase in the frequency count for Spanish which increases the gap between its resemblance to French and thus providing more accurate results after considering misspelled words.

The precision, recall and F-measures have been used to determine the efficiency of the proposed Cumulative Frequency Addition method, which are the standards of measurement in the field of Natural Language Processing. Precision is the proportion of correctly classified test samples in all samples classified to the given language, whereas recall is the proportion of correctly classified test samples in all samples of the given language. F-measure is the weighted harmonic mean of Precision and Recall that has been calculated by giving equal importance to both of the measures.

Language	**Precision (%)**	**Recall (%)**	**F-measure (%)**
Russian	98.91	98.64	98.75
English	97.76	96.44	97.09
German	97.53	95.78	96.64
Spanish	89.52	88.21	88.86
Latin	93.85	74.88	87.65
French	87.60	82.10	84.76

Table 4.1 Precision, Recall and F-measure values.

The performance of our system was examined using six thousand query test cases for each of the languages. The precision and recall values calculated on the basis of these test cases are given in the Table 1 along with their F- measures.

From the results, high precision values can be seen to have been achieved for all of the languages in the candidate set which has been the main objective right from the beginning. We see that we have achieved quite high values of precision (98.91%, 97.76%, 97.53%) as well as recall (98.64%, 96.44%, 95.78%) for Russian, English and German respectively. This clearly justifies the fact that these languages are very unique in their nature so their chances of being interpreted as some other language are extremely low. The highest values are achieved in the case of Russian language due to the uniqueness of their characters and grammar which is also seconded by the high frequency values.

In case of Spanish and French, we have a slightly lower precision (89.52% and 87.60%) when compared to German since both of these language shares its roots with Latin and hence share a large set of words with each other. In few test cases one language was incorrectly identified as the other due to a slight difference in their frequency count. Recall value for Spanish (88.21%) is high when compared to French (82.10%) and this is due to the reason that French was sometimes incorrectly identified as Spanish but the vice versa had a very few cases.

Considering the case of Latin, it has precision 93.85% and lower recall 74.88%. Latin being the parent of Spanish, French, Italian, Romanian, and other Romance languages, shares a lot of lexicon with these languages. Therefore, in cases in which our system predicted the language to be Latin has a high precision but a lower recall since it identified a few cases belonging to Spanish or French as well.

To summarize, our system not only identifies the language of the text that are short in length but also takes into consideration, the misspelled words present in the query, handles them gracefully and adds their significance in the entire process.
Overall, our system has a high accuracy with an encouraging performance seconded by the F- score of 91.60% for all of the languages in the candidate set.

Chapter 5
CONCLUSION

5.1 CONCLUSION

Although there are numerous language identification methods that have implemented various strategies for large documents or long texts, correctly identifying the short text has always been a challenge in this domain. Our method tackles the short texts quite gracefully and in an efficient manner providing accurate results most of the times. Our method also handles misspelled words and is able to provide accurate spellings for those words most of the times. This reduces the chances of incorrect results that might have been produced due to presence of misspelled words in the query. Our method learns with every new query that has been executed by adding words to the lexicon that are missing in the database. The system provides encouraging results when evaluated against a particularly challenging domain: short text with misspelled words which is supported by the overall F-measure score of 91.60%.

5.2 FUTURE SCOPE

More work need to be done on improving the precision values of Spanish and French so that the cases in which the language has been identified incorrectly are minimized. Latin being the parent of Spanish and French shows high resemblance with them and therefore work needs to be done on improving its recall values so that its result is not confused with the other two languages. The factor of computational costs haves not been included in our research and can be explored at a later stage to improve the efficiency of our algorithm.

REFERENCES

[1] Nicholas Akosu , Ali Selamat. Word Length Algorithm for language identification of under resourced languages. Journal of King Saud University - Computer and Information Sciences Volume 28, Issue 4, October 2016, Pages 457-469.

[2] Lena Grothe, Ernesto William De Luca and Andreas Nurnberger. A Comparative Study on Language Identification Methods. Proceedings of the International Conference on Language Resources and Evaluation, LREC 2008, 26 May - 1 June 2008, Marrakech, Morocco.

[3] Bashir Ahmed, Sung-Hyuk Cha, and Charles Tappert. Language Identification from Text using N-gram based Cumulative Frequency Addition. Proceedings of Student/Faculty Research Day, CSIS, Pace University, May 7th, 2004.

[4] Ciprian-Octavian Truic_a, Julien Velcin, Alexandru Boicea. Automatic Language Identification for Romance Languages using Stop Words and Diacritics. International Symposium on Sym- bolic and Numeric Algorithms for Scienti_c Computing (SYNASC 2015) , Sep 2015, Timioara, Romania. pp.243-246.

[5] Zampieri, M., & Gebre, B. G. VarClass: An open-source language identification tool for language varieties. In N. Calzolari, K. Choukri, T. Declerck, H. Loftsson, B. Maegaard, J. Mariani, et al. (Eds.), Proceedings of LREC 2014: 9th International Conference on Language Resources and Evaluation, 2014(pp. 3305-3308).

[6] Tommi Vatanen and Jaakko J. V, Language Identification of Short Text Segments with N-gram Models. In the roceedings of Vatanen2010LanguageIO,2010, LREC.

[7] Selamat A. Improved N-grams Approach for Web Page Language Identification. In: Nguyen N.T. (eds) Transactions on Computational Collective Intelligence V. Lecture Notes in Computer Science, vol 6910. Springer, Berlin, Heidelberg, 2011.

[8] Toole, Janine. Categorizing Unknown Words: Using Decision Trees to Identify Names and Misspellings. ANLP, Toole 2000 categorizing, 2000.

[9] ˇReh˚uˇrek, R., Kolkus, M.: Language identification on the web: Extending the dictionary method. In: Gelbukh, A. (ed.) CICLing 2009. LNCS (LNAI), vol. 5449, Springer, Heidelberg (2009) pp. 357–368.

[10] Radim Rˇ ehu°rˇek and Milan Kolkus. 2009. Language identification on the web: Extending the dictionary method. In Proceedings of CICLing 2009, pages 357–368.

[11] W.B. Cavnar and J.M. Trenkle. N-gram based text categorization. In Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval, 1994. pp 161–169.

[12] G. Grefenstette. Comparing two language identification schemes. In 3rd International conference On Statistical Analysis of Textual Data. 1995.

[13] T. Dunning. Statistical identification of language. Computing Research Laboratory, New Mexico State University. 1994. Technical report mccs pp 94-273

[14] Marcos Zampieri and Binyam Gebrekidan Gebre. Automatic identification of language varieties: The case of Portuguese. In Proceedings of KONVENS2012, Vienna, Austria. 2012, pp233–237.

[15] Canasai Kruengkrai, Prapass Srichaivattana, Virach Sornlertlamvanich, and Hitoshi Isahara. 2005. Language identification based on string kernels. In Proceedings of ISCIT-2005, pp 896–899.

[16] Marc Damashek.. Gauging similarity with n-grams: Language-independent categorization of text. Science, 1995, 267(5199): pp843–849.

[17] Penelope Sibun and Jeffrey C. Reynar. Language identification: Examining the issues. In Proceedings of SDAIR'96,1996, pp 125–135.

[18] Gerrit Reinier Botha and Etienne Barnard. Factors that affect the accuracy of text-based language identification. In Proceedings of PRASA 2007, 2007, pp 7–10.

[19] http://people.ds.cam.ac.uk/cjp16/queens/frreading1.htm, 15/10/17.

[20] http://people.ds.cam.ac.uk/cjp16/queens/spreading1.htm. 15/10/17.

[21] https://owlcation.com/humanities/FREE-German-Essays-Part-1-Family, 07/11/17.

[22] http://www.study-languages-online.com/reading_text_01.html, 09/ 11/ 17.

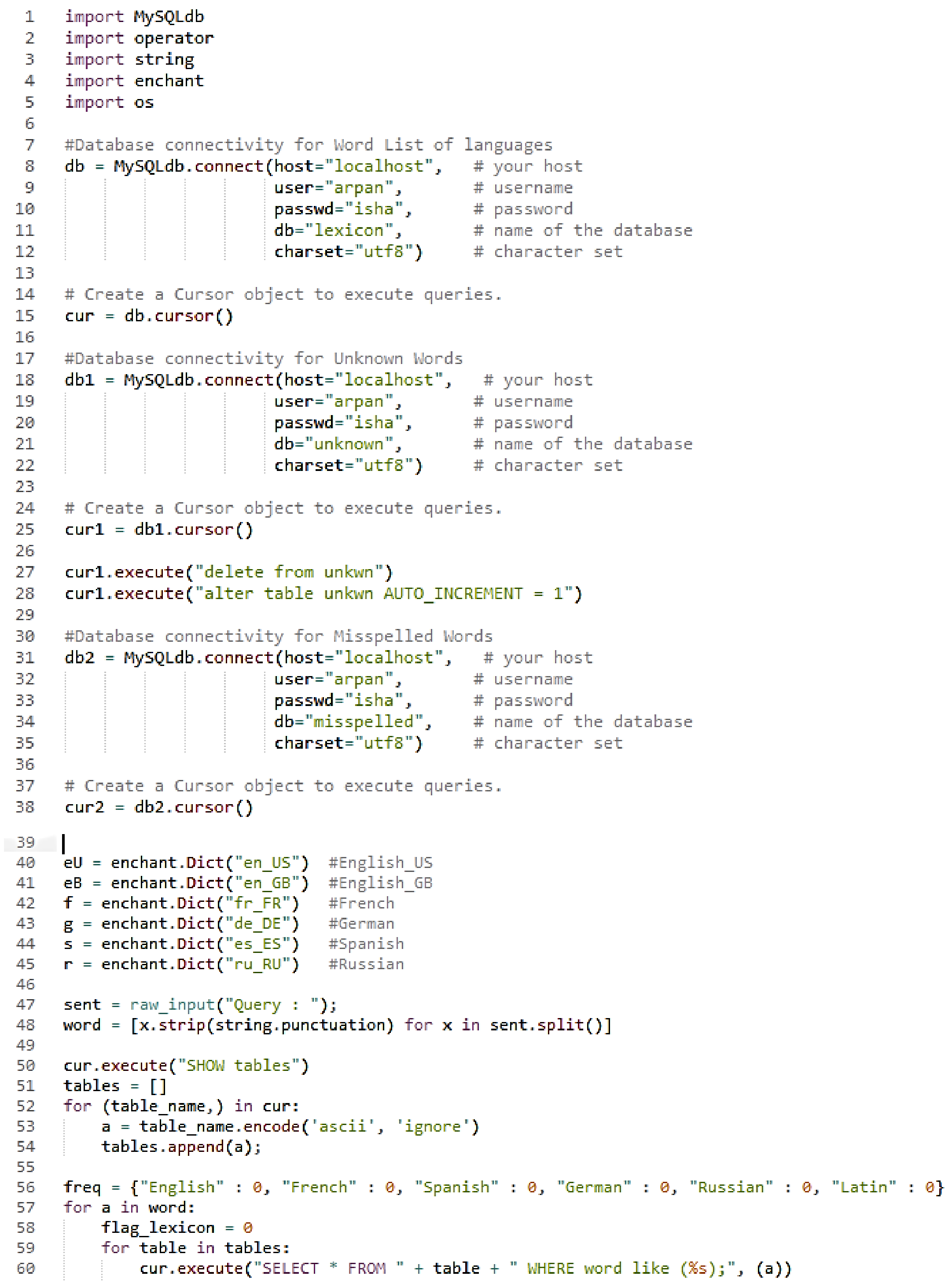
[23] https://www.lipsum.com/, 11/11/17.

[24] http://wortschatz.uni-leipzig.de/en, 25/11/17.

[25] http://practicalcryptography.com/miscellaneous/machine-learning/tutorial-automatic-language-identification-ngram-b/#reducing-the-amount-of-training-material, 25/11/17.

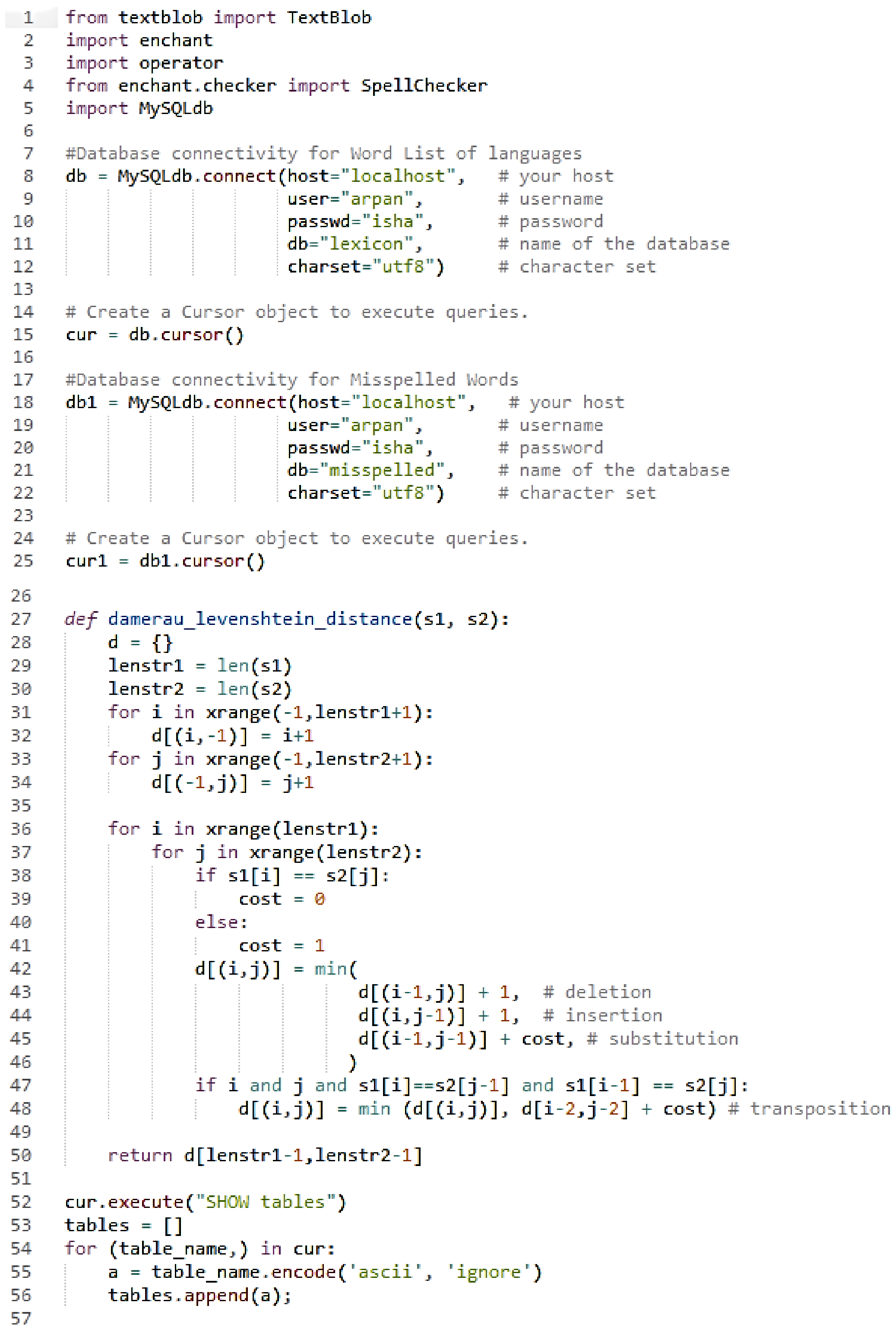
APPENDICES

Code for initial process of Language Identification.



```python
import MySQLdb
import operator
import string
import enchant
import os

#Database connectivity for Word List of languages
db = MySQLdb.connect(host="localhost",   # your host
                     user="arpan",       # username
                     passwd="isha",      # password
                     db="lexicon",       # name of the database
                     charset="utf8")     # character set

# Create a Cursor object to execute queries.
cur = db.cursor()

#Database connectivity for Unknown Words
db1 = MySQLdb.connect(host="localhost",   # your host
                     user="arpan",       # username
                     passwd="isha",      # password
                     db="unknown",       # name of the database
                     charset="utf8")     # character set

# Create a Cursor object to execute queries.
cur1 = db1.cursor()

cur1.execute("delete from unkwn")
cur1.execute("alter table unkwn AUTO_INCREMENT = 1")

#Database connectivity for Misspelled Words
db2 = MySQLdb.connect(host="localhost",   # your host
                     user="arpan",       # username
                     passwd="isha",      # password
                     db="misspelled",    # name of the database
                     charset="utf8")     # character set

# Create a Cursor object to execute queries.
cur2 = db2.cursor()

eU = enchant.Dict("en_US")  #English_US
eB = enchant.Dict("en_GB")  #English_GB
f = enchant.Dict("fr_FR")   #French
g = enchant.Dict("de_DE")   #German
s = enchant.Dict("es_ES")   #Spanish
r = enchant.Dict("ru_RU")   #Russian

sent = raw_input("Query : ");
word = [x.strip(string.punctuation) for x in sent.split()]

cur.execute("SHOW tables")
tables = []
for (table_name,) in cur:
    a = table_name.encode('ascii', 'ignore')
    tables.append(a);

freq = {"English" : 0, "French" : 0, "Spanish" : 0, "German" : 0, "Russian" : 0, "Latin" : 0}
for a in word:
    flag_lexicon = 0
    for table in tables:
        cur.execute("SELECT * FROM " + table + " WHERE word like (%s);", (a))

        if cur.rowcount != 0:
            flag_lexicon = 1
            for row in cur.fetchall() :
                freq[table] = freq[table] + row[2]
    if flag_lexicon == 0:
        #print "" + a + " inserted into unknown"
        cur1.execute("INSERT INTO unkwn (word, freq) VALUES (%s,%s);", (a,"0"))

print "\n"
sorted_freq = sorted(freq.items(), key=operator.itemgetter(1), reverse = True)
print "{:<8} {:<15} ".format('Language','Frequency')
for lang, fr in sorted_freq:
    print "{:<8} {:<15} ".format(lang, fr)

result = max(freq, key=freq.get)
if freq[result] == 0:
    print "\nLanguage is difficult to Identify !\n"
else:
    print "The language of the entered Text is : " + result + "\n"

with open('variable.txt', 'w') as file:
     file.write(str(freq))

cur1.execute("SELECT * FROM unkwn")
if cur1.rowcount != 0:
    for row in cur1.fetchall():
        word = row[1]
        flag_dict = 0
        if eU.check(word) or eB.check(word):
            flag_dict = 1;
            print "" + word + " found in English dictionary and inserted into Database"
            cur.execute("INSERT INTO English (word, freq) VALUES (%s,%s);", (word,"1"))
        if f.check(word):
            flag_dict = 1;
            print "" + word + " found in French dictionary and inserted into Database"
            cur.execute("INSERT INTO French (word, freq) VALUES (%s,%s);", (word,"1"))
        if g.check(word):
            flag_dict = 1;
            print "" + word + " found in German dictionary and inserted into Database"
            cur.execute("INSERT INTO German (word, freq) VALUES (%s,%s);", (word,"1"))
        if r.check(word):
            flag_dict = 1;
            print "" + word + " found in Russian dictionary and inserted into Database"
            cur.execute("INSERT INTO Russian (word, freq) VALUES (%s,%s);", (word,"1"))
        if s.check(word):
            flag_dict = 1;
            print "" + word + " found in Spanish dictionary and inserted into Database"
            cur.execute("INSERT INTO Spanish (word, freq) VALUES (%s,%s);", (word,"1"))

        if flag_dict == 0:
            print "" + word + " is a misspelled word."
            cur2.execute("INSERT INTO mispld (word, freq) VALUES (%s,%s);", (word,"0"))

db.commit()     cur.close()     db.close()

db1.commit()    cur1.close()    db1.close()

db2.commit()    cur2.close()    db2.close()

os.system("python levenshtein_misspelled.py")

```

Code for Identification of Language with inclusion of misspelled words


```python
from textblob import TextBlob
import enchant
import operator
from enchant.checker import SpellChecker
import MySQLdb

#Database connectivity for Word List of languages
db = MySQLdb.connect(host="localhost",   # your host
                     user="arpan",       # username
                     passwd="isha",      # password
                     db="lexicon",       # name of the database
                     charset="utf8")     # character set

# Create a Cursor object to execute queries.
cur = db.cursor()

#Database connectivity for Misspelled Words
db1 = MySQLdb.connect(host="localhost",   # your host
                     user="arpan",       # username
                     passwd="isha",      # password
                     db="misspelled",    # name of the database
                     charset="utf8")     # character set

# Create a Cursor object to execute queries.
cur1 = db1.cursor()

def damerau_levenshtein_distance(s1, s2):
    d = {}
    lenstr1 = len(s1)
    lenstr2 = len(s2)
    for i in xrange(-1,lenstr1+1):
        d[(i,-1)] = i+1
    for j in xrange(-1,lenstr2+1):
        d[(-1,j)] = j+1

    for i in xrange(lenstr1):
        for j in xrange(lenstr2):
            if s1[i] == s2[j]:
                cost = 0
            else:
                cost = 1
            d[(i,j)] = min(
                            d[(i-1,j)] + 1,  # deletion
                            d[(i,j-1)] + 1,  # insertion
                            d[(i-1,j-1)] + cost, # substitution
                           )
            if i and j and s1[i]==s2[j-1] and s1[i-1] == s2[j]:
                d[(i,j)] = min (d[(i,j)], d[i-2,j-2] + cost) # transposition

    return d[lenstr1-1,lenstr2-1]

cur.execute("SHOW tables")
tables = []
for (table_name,) in cur:
    a = table_name.encode('ascii', 'ignore')
    tables.append(a);

```

```python
with open('variable.txt', 'r') as file:
        s = file.read()
        freq = eval(s)

cur1.execute("SELECT * FROM mispld")
if cur1.rowcount != 0:
    for row_mispld in cur1.fetchall() :
        word = row_mispld[1]
        #print "**" + word + "**"

        table = "English"
        w = TextBlob(word)
        sugEn = w.correct()
        #print sugEn + "   English"
        cur.execute("SELECT * FROM " + table + " WHERE word like (%s);", (sugEn))
        if cur.rowcount != 0:
            for row in cur.fetchall() :
                freq[table] = freq[table] + row[2]
        else:
            cur.execute("INSERT INTO " + table + " (word, freq) VALUES (%s,%s);", (sugEn,"1"))

        table = "French"
        chkrFr = enchant.checker.SpellChecker("fr_FR")
        chkrFr.set_text(word)
        for err in chkrFr:
            if err.suggest():
                sugFr = err.suggest()[0]
                #print sugFr + "   French"
                cur.execute("SELECT * FROM " + table + " WHERE word like (%s);", (sugFr))
                if cur.rowcount != 0:
                    for row in cur.fetchall() :
                        freq[table] = freq[table] + row[2]
                else:
                    cur.execute("INSERT INTO " + table + " (word, freq) VALUES (%s,%s);", (sugFr,"1"))

        table = "Spanish"
        chkrEs = enchant.checker.SpellChecker("es_ES")
        chkrEs.set_text(word)
        for err in chkrEs:
            if err.suggest():
                sugEs = err.suggest()[0]
                #print sugEs + "   Spanish"
                cur.execute("SELECT * FROM " + table + " WHERE word like (%s);", (sugEs))
                if cur.rowcount != 0:
                    for row in cur.fetchall() :
                        freq[table] = freq[table] + row[2]
                else:
                    cur.execute("INSERT INTO " + table + " (word, freq) VALUES (%s,%s);", (sugEs,"1"))

        table = "German"
        chkrDe = enchant.checker.SpellChecker("de_DE")
        chkrDe.set_text(word)
        for err in chkrDe:
            if err.suggest():
                sugDe = err.suggest()[0]
                #print sugDe + "   German"
                cur.execute("SELECT * FROM " + table + " WHERE word like (%s);", (sugDe))
                if cur.rowcount != 0:
                    for row in cur.fetchall() :
                        freq[table] = freq[table] + row[2]
                else:
                    cur.execute("INSERT INTO " + table + " (word, freq) VALUES (%s,%s);", (sugDe,"1"))

        table = "Russian"
        chkrRu = enchant.checker.SpellChecker("ru_RU")
        chkrRu.set_text(word)
        for err in chkrRu:
            if err.suggest():
                sugRu = err.suggest()[0]
                #print sugRu + "    Russian"
                cur.execute("SELECT * FROM " + table + " WHERE word like (%s);", (sugRu))
                if cur.rowcount != 0:
                    for row in cur.fetchall() :
                        freq[table] = freq[table] + row[2]
                else:
                    cur.execute("INSERT INTO " + table + " (word, freq) VALUES (%s,%s);", (sugRu,"1"))

        #print "\n"
print "\n"
print "After considering misspelled words : "
sorted_freq = sorted(freq.items(), key=operator.itemgetter(1), reverse = True)
print "{:<8} {:<15} ".format('Language','Frequency')
for lang, fr in sorted_freq:
    print "{:<8} {:<15} ".format(lang, fr)

result = max(freq, key=freq.get)
if freq[result] == 0:
    print "\nLanguage is difficult to Identify !\n"
else:
    print "The language of the entered Text is : " + result + "\n"

cur1.execute("delete from mispld")
cur1.execute("alter table mispld AUTO_INCREMENT = 1")

db.commit()
cur.close()
db.close()

db1.commit()
cur1.close()
db1.close()
```