Syntatic Feature based Classification Algorithm to Detect Validity of Text

Manika Gupta
(M.Tech. Scholar)
Computer Science Department
Rajasthan College of Engineering for Women
Jaipur, Rajasthan

Vineet Khanna
(Assistant Professor)
Computer Science Department
Rajasthan College of Engineering for Women
Jaipur, Rajasthan

ABSTRACT

The complexity of a natural language itself is very challenging as the natural language is not free from ambiguity problem. It is almost impossible to identify that the given text is having sense or not. In today's scenario it becomes even much important to detect that input is given by human or a machine. A valid input with sense is needed everywhere from Social media platforms to Business Intelligence. This Classification algorithm aims to detect whether the given input text is valid, or randomly typed in a keyboard. It returns a percentage value where a lower one means valid text, and a higher value means random text. The approach is based on identifying that the amount of unique chars, amount of vowels of letters, the word/char ratio (in %) are in a usual range. Then it further calculates "deviation score" to compute the accuracy of given input.

Keywords

Data mining; text mining; text classification; sentence validation ; pattern learning

1. INTRODUCTION

Data mining is the process of extracting information from a data set and transform it into an understandable form for further use.

The data mining task is the automatic or semi-automatic analysis of large quantities of data to extract previously unknown interesting patterns such as groups of data records(cluster analysis), unusual records (anomaly detection) and dependencies(association rule mining)[1]. It is an analysis task which derives useful patterns and trends from the data repository. Data mining is an important task in the whole knowledge discovery process [2].

Text mining is the analysis of data contained in natural language text. Classification is a basic functionality in mining. Classification has wide applications in the area of data mining, machine learning, database, and information retrieval etc. The various diverse applications include target marketing, medical diagnosis, news group filtering, and document organization.

As the most natural form of storing information is *text*, text mining is believed to have a commercial potential higher than that of data mining. In fact, a recent study indicated that 80% of a company's information is contained in text documents. Text mining, however, is also a much more complex task (than data mining) as it involves dealing with text data that are inherently unstructured and fuzzy [3].Text classification approach is gaining more importance because of the accessibility of large number of electronic documents from a variety of resource [4, 5]. Automatic Text Classification involves assigning a text document to a set of pre-defined classes automatically, using a machine learning technique. The classification is usually done on the basis of significant words or features extracted from the text document [6]. Since the classes are pre-defined it is a supervised machine learning task [7]. The goal of text mining is to enable users to extract information from textual resource and deals with operation such as retrieval, classification, clustering, data mining, natural language preprocessing and machine learning techniques together to classify different pattern [5, 8].

Text mining also referred to as text data mining is the process of deriving high quality information from text. This high quality information is derived through finding out patterns and trends through methods like statistical pattern learning [2]. Text mining can be visualized as consisting of two phases: *Text refining* that transforms free-form text documents into a chosen intermediate form, and knowledge distillation that deduces patterns or knowledge from the intermediate form [3].

Nowadays, enormous amount of digitally stored information is available on internet. In order to prevent sinking in it, filtering and extraction of information are necessary. A significant and opportune tool that assists and interprets huge quantities of text presented in documents is gibberish classifier. This paper deals with Sentence Validation - a sub-field of Natural Language Processing. Sentence Validation refers to verification of "Natural Language" sentence on basis of its syntax and semantics. Sentence Validation finds its applications in almost all fields of NLP - Information Retrieval, Information Extraction, Question-Answering, Visualization, Data Mining, Text Summarization, Text Categorization, Machine and Language Translation, Dialogue and Speech based Systems and many other one can think of [9, 10].It proposes tackling noise by syntactic filtering over a sentence level. Syntactic filtering ascertains the valid sentence structure as system utterances [9]. It draws a distinction between valid and invalid text; that is whether the given input text is having some sense, or randomly typed by keyboard.

For performance evaluation, we have calculated deviation score and classify the text as valid or nonsense based on percentage we obtained from different functions we used in this algorithm. The higher this score, the more the percentage deviates from usual range.

Basically proposed algorithm takes input text either from machine or from human and split it into chunks, and then it find unique characters per chunk percentage. Further it calculates vowels percentage and word to character ratio. Finally it calculates the deviation Score and classify the text given in invalid or valid.

2. RELATED WORK

Review of Related Work-In this section we will discuss related work we found in literature to propose classification algorithm based on Text Analysis and Pattern Learning. A number of papers have been published based on classification algorithms. Lakshay Arya in his research titled "Sentence validation by Statistical Language Modeling and Semantic Relations" compares different models used for sentence validation. Sentence Validation is approached in two ways - by Statistical approach and Semantic approach. In both approaches database is trained with the help of sample sentences of Brown corpus of NLTK. The statistical approach uses trigram technique based on N-gram Markov Model and modified Kneser-Ney Smoothing to handle zero probabilities. As another testing on statistical basis, tagging and chunking of the sentences having named entities is carried out using pre-defined grammar rules and semantic tree parsing, and chunked off sentences are fed into another database, upon which testing is carried out. Finally, semantic analysis is carried out by extracting entity relation pairs which are then tested [10].

D.Y. Sakhare in the research titled "Syntactic and Sentence Feature Based Hybrid Approach for Text Summarization" presented an approach of text summarization that combines feature and syntactic structure of the sentences where, two neural networks are trained based on the feature score and the syntactic structure of sentences. Finally, the two neural networks are combined with weighted average to find the sentence score of the sentences. The results showed that the proposed approach achieved F-measure of 80% for the compression ratio 50 % [11]. Ian Tenney in the research titled "A general-purpose sentence-level nonsense detector" has classified well-formed English sentences from the large volume of fragments, headlines, incoherent drivel, and meaningless snippets present in internet text. He take a lightweight approach, using a mix of heuristics, lightweight token-based features, part-of-speech features, and language model scores as features to avoid computationally-intensive parsing or neural networks. He structure his system as a binary classification task on a heterogeneous feature space, consisting of to produce a final answer of "sentence" or "nonsense" for a given line of text. His model is capable of reaching 96% precision while maintaining 80% recall [12]. Kush Jain in the research titled "Chunked N-Grams for Sentence Validation" integrate the two approaches used for sentence validation i.e. Statistical Means (n-grams) or by Semantic Means (by constructing some kind of Knowledge graph).It provides a threshold value for probability which will help us to distinguish between correct and incorrect test sentences. In this paper, he only applied Named Entity Pre-processing. However there might be even better semantic preprocessors available which could make this model even better [9]. To outfit the shortcoming of Ian Tenny's model which tries to discriminates valid sentence from nonsense on the basis of grammar. Therefore in this paper I have attempt to create a approach that will be able to detect nonsense from large text documents, portals, tweets, web pages, Wikipedia, social web sites, or text file in any format. This approach tries to detect randomly typed sentences from valid sentence regardless of grammatically.

3. PROBLEM STATEMENT

With the increase in the volume of content available in digital format gives rise to a problem to manage this online textual data. As a result, it has become necessary to classify/categorize large texts (documents) in order to check their validity. To classify millions of text document manually and to check their validity is an expensive and time consuming task. Therefore an approach for automatic text classification is needed to classify the text documents whose accuracy and time efficiency is need to be better than manual text classification.

4. PROPOSED ALGORITHM

Initially, this algorithm takes input text either from machine or from human and split it into chunks, and then it find unique characters per chunk percentage. Further it calculates vowels percentage and word to character ratio. Finally it calculates the deviation Score and classify the text given in input. In this algorithm, various functions are applied on input text, detailed approach for proposed algorithm is:

Checking the unique characters:

1. To check the % of unique characters, first split the string in chunks of 35 characters (considering that 35 is an average character length in a sentence).
2. When doing this, it can happen that the last chunk does not have 35 characters -- in this case, if the chunk size is less than 10 characters, add these chars to the chunk before the last and delete the last. (This is only possible if there are 2 chunks or more)
3. . After splitting the string into chunks, create an empty list. Then loop over all chunks. Calculate the amount of unique characters in the current chunk. Then divide it by the total amount of characters in the chunk and add it to the list.
4. After doing the above, calculate the average of the list, multiply it by 100, and return it. That calculates the percentage of unique characters in chunks; checking whether it is in a usual range is done later.

Checking the vowels:

1. Firstly two integer variables 'vowels' and 'total' are initialized. Then run through each and every character within the given string. If the character is not an alphabet letter, proceed further without doing anything for that letter.
2. . If it is an alphabet letter, increase variable 'total' by 1 and check whether it is a vowel or not and if it is, increase variable vowels by 1. After running through all characters, divide vowels by total, multiply it by hundred and return it.

Checking the word/character ratio:

1. For calculating the word/character ratio, we need to split the string, and for that regular expression (regex [W_] is used (splitting by all non-word characters and an underscore)
2. Further remove all whitespace-only/empty items from the resulting array. Thereupon, divide the count of words by the count of chars, multiply it by 100 and return it.

Calculating "deviation score":

1. The above functions all return a percentage, but this cannot be directly used to calculate the final score – for that there is need to calculate how much the percentage deviates from the usual range, and then give it a score. The higher this score, the more the percentage deviates.
2. The function to calculate this score has to accept three arguments: the given percentage, the lower bound of the usual range and the upper bound.
3. Using all functions, we can calculate the final score! First we calculate the percentages using the first

three functions. Then, we call the deviation score for each of them.

Further it'll use some log functions with some base value to return the exact final percentage value of a sentence being valid.

The proposed approach can be applied on any text document and this is explained with the help of flow chart below.

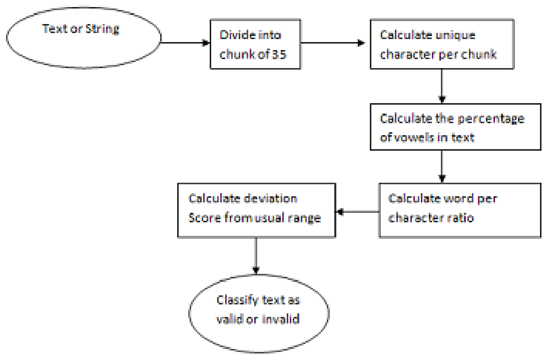
Text or String
Divide into chunk of 35
Calculate unique character per chunk
Calculate the percentage of vowels in text
Calculate word per character ratio
Calculate deviation Score from usual range
Classify text as valid or invalid

Figure 1: Flow Chart of proposed algorithm

These can be the possible applications of proposed algorithm:

1. Spam Detection -This is also one of the key steps to check the validity of contents in spam detection. The most widely recognized form of spam is e-mail spam. Text classification systems can classify incoming e-mail as negative (non-spam) or positive (i.e. spam) and reject those that they finds to be spam.

2. Content Filtering -Needed Everywhere in Social Media Platforms, Search Engines etc. It is the activity of classifying a stream of incoming documents dispatched in an asynchronous way by an information producer to an information consumer.

3. Text Categorization -In terms of validity it is helpful to categorize text. Text Categorization is need to handle and organize large quantities of documents, i.e. large enough that their manual organization into classes is either too expensive or not feasible within the time constraints imposed by the application.

4. Hierarchical categorization of Web pages- Due to the tremendous increase of the amount of Web pages or sites, it is more and more difficult to find the information users are interested in. Classifying Web pages or sites under hierarchical catalogues can make a Web search easier by restricting the search to a particular category of interest. While manual categorization of Web pages is infeasible and costly to maintain.

5. Analysis for Text Clustering- Removal of random text is an important step in process of text clustering. Text Classification is analysis to textual documents. It has applications in automatic document organization, topic extraction and fast information retrieval or filtering.

6. Word Sense Disambiguation- Word sense disambiguation (WSD) is the activity of finding, given the occurrence in a text of an ambiguous word, the sense of any particular word occurrence. WSD is very important for many applications, including natural language processing, and indexing documents by word senses rather than by words for Information Retrieval purposes.

5. RESULTS AND DISCUSSION

This section describes the detailed the experimental results. The proposed Classification algorithm is implemented using Microsoft Visual studio.net(C#) ultimate 2013 on windows platform. C#(C Sharp) programming language is platform independent, so it is better to implement any program in such type of languages. Based on functions results and some parameters used, percentage of accuracy is obtained by executing this classification algorithm and I have presented it here in the form of table shown in Table 1.

Dataset which we have used is real data collected from Wikipedia and other web pages ranging from news portal to educational sites. Results are obtained after applying this algorithm on dozens of distinct datasets. For each input dataset, it calculates accuracy i.e. manually calculated percentage of random text to the percentage of random text calculated by proposed scheme. In Table 1 we have displayed results for only 3 datasets and so in our various graphs. Moreover, accuracy obtained by this algorithm is generally more than 90%. Therefore it is time efficient than doing manually. Results also show the percentage of accurate/valid text in the input file.

Table 1: Percentage of accuracy obtained using different dataset

Datasets (No. of Lines)	Manual Random text	Manual Accurate text	Calculated random text	Calculated accurate text	Accuracy
100	0	100	1	99	99
1320	50	50	52	48	96
1585	0	100	1	99	99

Table 1 distinguishes manually random text with calculated random text and manually accurate text to calculated accurate text of 3 different datasets and accuracy obtained by this algorithm.

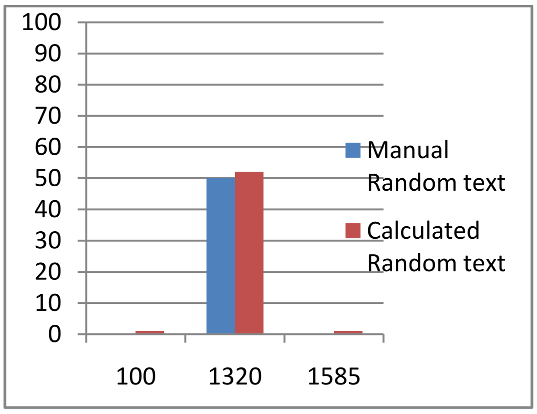

Fig 2: Comparison of manual random text and calculated random text

Fig 2 shows the percentage of random text calculated by this algorithm. Random text here means text that is randomly typed by keyboard or text not having any sense. It is a plot between random text calculated manually and random text calculated by proposed algorithm.

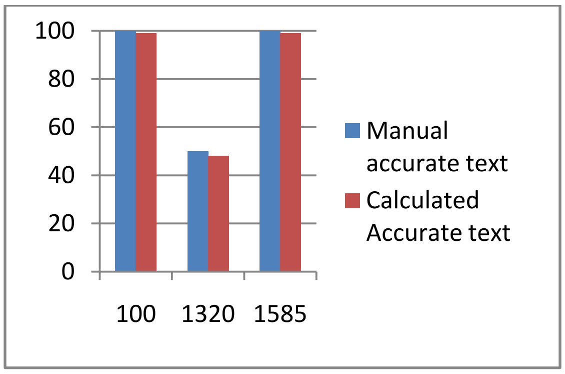

Fig 3: Comparison of manual accurate text to calculated random text

Fig 3 shows the percentage of accurate text calculated by this algorithm. Accurate text here means text that is valid or having any sense. It is plot between meaningful sentences calculated manually and meaningful sentence calculated by this algorithm. Graphs are plotted between manually accurate text and calculated accurate text of 3 different text datasets.

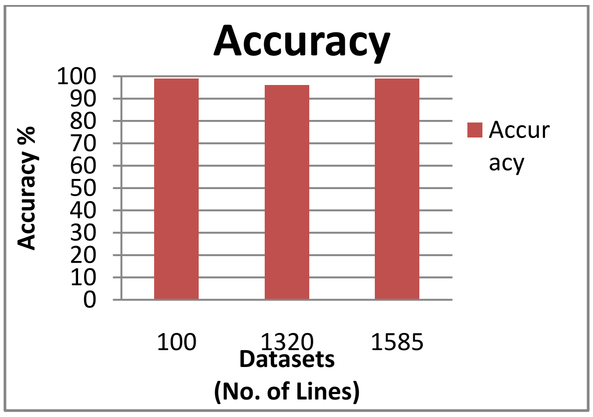

Fig 4: Percentage of accuracy obtained using proposed algorithm

Fig 4 shows accuracy calculated by this algorithm. Accuracy here means manually calculated percentage of random text to the percentage of random text calculated by proposed algorithm.

6. CONCLUSION AND FUTURE WORK

The algorithm present here successfully able to discriminate valid sentences from nonsense. This algorithm takes input text either from machine or from human and is capable to detect whether the given input text is valid, or randomly typed in a keyboard. Proposed approach for automatic text classification is to classify the text documents whose accuracy and time efficiency is far better than manual text classification. It can be useful if you have, for example, forum software where you want to check the posts before publishing them.

Proposed work is used for long sentences or long text. If there is requirement to check this algorithm for single words, add a dictionary file instead. The algorithm is not accurate for single words; you have to use it on sentences and longer the sentence, more accurate the result. An easy addition to the algorithm that'd 'fix' that case of single words would be to match each word to a dictionary word. You'd expect 'normal' text to have more than 50% of its words in a standard dictionary. You could even find out the frequencies of just the most common words a, an, and, at, it, he, she, the, etc. and just look for a substantial deviation from that. As a part of future work the algorithm can be enhanced to include features for identifying emotions or behavioral patterns from a given set of text.

7. REFERENCES

[1] Ms. Ranju Marwaha, Data Mining Techniques and Applications in Telecommunication Industry,International Journal of advanced research in computer science and software engineering, Volume 4, Issue 9,September 2014

[2] Jijy George ,Sandhya .N., Suja George," Classification Problem In Text Mining" International Journal of Innovative Research in Advanced Engineering (IJIRAE) ISSN: 2349-2163 Volume 1 Issue 8 (September 2014)

[3] *Ah-Hwee Tan,* "Text Mining:The state of the art and the challenges"

[4] Monica Bali, Deipali Gore," A Survey on Text Classification with Different Types of Classification Methods, International Journal of Innovative Research in Computer and Communication Engineering Vol. 3, Issue 5, May 2015.

[5] Bhumika, Prof Sukhjit Singh Sehra, Prof Anand Nayyar, A Review Paper On Algorithms Used For Text Classification, Internatioal Journal of Application or Innovation in Engineering & Management (IJAIEM), Volume 2, Issue 3, March 2013

[6] Pratik Agrawal, Prof. A.J.Agrawal , Implementation of Semantic Analysis Using Domain Ontology, IOSR Journal of Computer Engineering (IOSR-JCE), 8727Volume 11, Issue 3 (May. - Jun. 2013)

[7] Mita K. Dalal, Mukesh A. Zaveri," Automatic Text Classification: A Technical Review", International Journal of Computer Applications (0975 – 8887)Volume 28– No.2, August 2011

[8] Vandana Korde, C Namrata Mahender," Text Classification And Classifiers:A Survey, International Journal of Artificial Intelligence & Applications (IJAIA), Vol.3, No.2, March 2012

[9] Kush Jain, Priya Khatri and Garima Indolia," Chunked N-Grams for Sentence Validation" 2015 International Conference on Computational Science

[10] Lakshay Arya," Sentence Validation by Statistical Language Modeling and Semantic Relations, International Journal of Computer Applications Technology and Research,Volume 3– Issue 12, 812 - 814, 2014

[11] D.Y. Sakhare, Dr. Raj Kumar," Syntactic and Sentence Feature Based Hybrid Approach for Text Summarization, *I.J. Information Technology and Computer Science,* 2014, 03, 38-46 Published Online February 2014 in MECS

[12] Ian Tenney." A general-purpose sentence-level nonsense etector", December 2014