Probabilistic Model to Access the Possible Information using Query Representation

M.Sandhya
M.Tech.(SWE),Dept.of CSE
Kakatiya Institute of Technology & Science
Warangal-15,Telangana,India.

B.Hanmanthu
Assistant Professor, Dept.of CSE, Kakatiya Institute of Technology & Science
Warangal-15, Telangana, India.

B.Raghuram
Assistant Professor, Dept.of CSE, Kakatiya Institute of Technology & Science
Warangal-15, Telangana, India

ABSTRACT
The searching of data Process end users search their data needs using query representation, by using this way of retrieving data may not meet their expectations. To achieve end users goal, developers implement several techniques. Previously end users follow greedy algorithm with IQp [1]. But in this paper we will work forward with n-gram Language model. In this approach, end user selects the searchable keyword with the length of minimum n+1 data units. With n data units users failed to retrieve their expectations. This approach includes Probabilistic algorithms used for large vocabulary word correction system with language model. This paper explains data search using n gram data model [5], web server and allows users to interact via browser front end. We outline the challenges and discuss the implementation of our system including results of extensive experimental evaluation.

Keywords:
N-Gram, Spell Checker, Search Engine, Query construction.

1. INTRODUCTION
Data base implementers follow many techniques to achieve their requirements while retrieving their informational needs. Previous techniques were used ranked list or schema knowledge or decision tree search, graphical search [2][3] results some lacking of end users goal. They use greedy algorithm to retrieve data using keyword search in browser. Here in our paper, implementing new technique with keyword data contains at least n+1 data units, i.e. we are using n data gram language model [4]. Language models are mostly probabilistic language models. The main goal behind this model is assign probability to sentence or word [5]. Purpose is to convert the user specified keyword into machine understandable code i.e. machine translation, spell correction for user specified keyword and finally for speech recognition [6]. In this paper we can use spell checker for correct the misspelled words by user. Spell checker is used as enhancement of n gram model. This spell checker is used for displaying query construction with correct words when user gives in correct word as input. The goal of this project is mainly to get appropriate data retrieval even user specifies wrong keyword as input. This can be achieved by keyword retrieval using n gram language model. The project describes the retrieving of data with correct words according to prioritization. The process includes

1. Keyword entry with correct word or in correct word.
2. Searching.
3. Display accurate results.

We will discuss search process and all the process in deep in next session. Process must contain previous word histories and n gram is consecutive sequence of tokens.

2. PREVIOUS SCENARIO
Most of query construction process is used for retrieval of user specified data by giving input keyword. Sometimes it gives appropriate data and sometimes may not give. It results in search performance and time consuming. In previous paper[1] decision trees with greedy algorithm is used for giving appropriate data. By this algorithm we failed to retrieve all possible data, it works only for some words mentioned in dictionary. It is not possible for retrieving all the words mentioned in dictionary. In Paper[1] we implement IQP model for retrieving information from data base, it won't check for spellings, and user must need to know schema knowledge. To retrieve user accurate data we proposed a system with spell checker for n gram language model. The model can be described in proposed scenario. Figure 1 represents how the system will be working. It shows user gives input to search engine to retrieve appropriate data. Search engine look up in search reporter with dictionaries, indexes. Finally retrieved data updated within the data base.

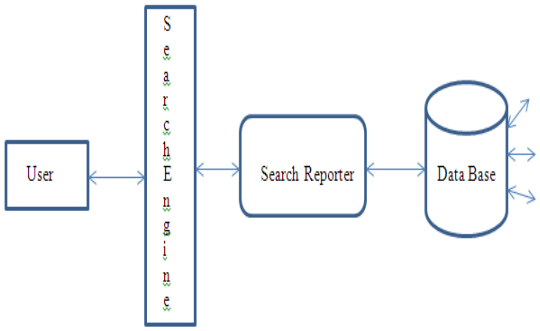


Fig1: Present system

3. PROPOSED SCENARIO
To achieve all words in the dictionary are to be retrievable and retrieves correct data after correction we can use N- gram language model. N-gram language model works based on probabilistic model [5]. Word is to be divided as tokens like unigram (word should be separated with single characters), Bigram (word should be separated with 2 letters) and finally n –gram. N-gram combines the word letter one with another letter in the same data set. The complete word is to be considered as tokens based on n gram. Word should be classified as possible tokens and each word set is to be compared with words mentioned in dictionary. When user gives wrong input data, it should be divided according to tokens and lookup for word sets in dictionary and selects appropriate word according to prioritization based on probability. Then it will show results based on probability. Most of unmatched word sets probabilities are represented as 0 and exact word which is user specified word is represented

with probability 1.By selecting the appropriate word based on probability, user have to go for further selection.

Computing Probabilities

P(w1,w2,….wT) = P(w1)p(w2/w1)p(w3/w2w1)

p(wT/w1w2..WT-1)

This process can't keep track of all historical possible keywords. For this purpose we can go for approximating probabilities

Basic Idea Limit history to fixed number of words N (Markov Assumption)

P (wk/w1…..wk-1)~p(wk/wk-n+1,….wk-1)

N=1 Unigram model

P (wk/w1…..wk-1)~ p(wk)

->p (w1, w2….wn) ~p (w1) p (w2)….p (wn)

Building N-Gram Language Models

Use existing sentences to compute n-gram probability estimates (training)

Terminology:

N = total number of words in training data (tokens)

V = vocabulary size or number of unique words (types)

C (w1... wk) = frequency of n-gram w1... wkin training data

P (w1... wk) = probability estimate for n-gram w1 ... wk

P (wk|w1... wk-1) = conditional probability of producing wkgiven the

historyw1, wk-1

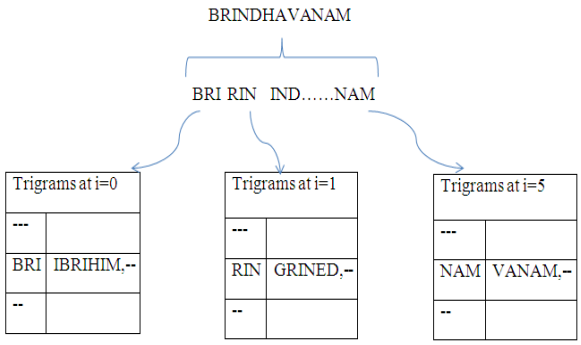


Fig.2: 3-Ggram language model example

Fig 2 represents how the data will be divided based on 3 gram language model. Given input should be classified with the length of 3 words. We can get number of possible words according to probability.

3.1 N-Gram Language Model

Architecture of n-gram for spell checker of word having 3 steps process as

1. Initialization
2. Indexing
3. Spell Correction

3.1.1 Initialization

Complete data is loaded into main memory. This data includes dictionary and data analysis according to priority, it should display results. Analysis lookups for each word in dictionary when user gives misspell word.

3.1.2 Indexing

After data loading into memory, data in the dictionary and analysis are assigned with some indexes for easy identification of data. Indexes are represented in Integer, it represents the word position in the dictionary as well analysis. These indexes used for lookups only small portion of location not the entire data set.

3.1.3 Spell Correction

Interface consists of Text area like word or anything else. When user gives improper word with insertion of extra word or substitution or deletion of some letters, our language model can look ups for tokens based on indexes and displays the result according to probability wise. The prioritization should be done with analysis phase. For more understanding of n gram language model, refer to [4][5].Detailed description will be present here. The errors which are with input keyword are defined below.

Types of Errors

When user enters error prone word to retrieve expected data, there should some types of error pruned data.

1. Non Word errors
2. Real Word Errors

3.1.4 Non Word Errors

Non word errors are not presented in dictionary. To get exact results, we have to generate words similar to error pruned words. From these similar words, user selects shortest weighted edit distance error. With this type user face highest noisy…… channel problem.

3.1.5 Real Word Errors

Solution for non-word errors is, for every word in dictionary, we have to generate word set. Word set includes similar pronunciation of error data; similar spelling of error pruned word and finally word should be present in word set.

Error Theory

Error data should be in these formats.

1. Substitution format
2. Deletion format
3. Insertion format
4. Transposition format

Substitution Format

User specified word is to be in the form of; any letter in the word w should be substituted with any other letter.

Ex: User gives input (wrong) as STATBS instead states, our searching process should construct query with another letters as STATES. Here B is to be replaced with E.

Deletion Format

User specified word is STATS instead of STATES. Here word lost a letter. Any way our searching process construct query with suitable word and it shows result as STATES.

Insertion Format
User specified word includes STATTES instead of STATES. Here word is having one extra letter, our searching process displays by deleting extra letter and gives correct word as STATES.

Transposition Format
User gives input by replacing one letter with another letter from the data word, and then it should shows results in correct word format. Use gives any of these formats of word; initially it should be search in the world dictionary according to the indexes. It should search for location where exactly the wrong input word similarly matching, then it should replace that wrong word with accurate word according to probability.

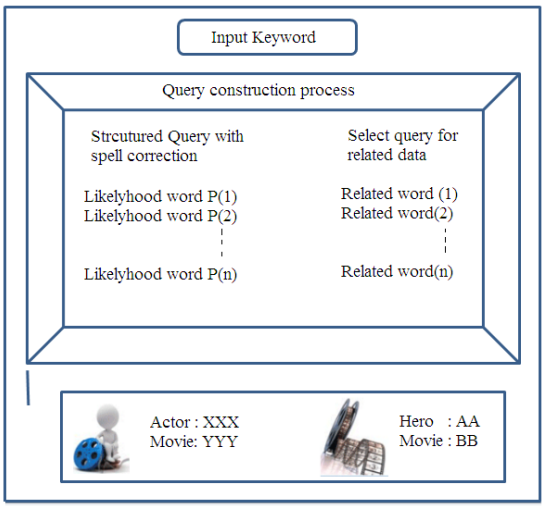


Fig3. Expected Result Window

Figure3 shows the current system outlook, how the process should be done within the system and displays result. This could be explained in next session.

3.2 Spell Correction

It takes a user input and provides correct word. This can be done by search engine request. Words that are the searchable are look up in index with possible words that are matched to that input keyword. This can be done within the search engine it also includes language model [5][8].

3.3 Search Engine

It is server side search engine program for web application. It provides fast way to get users needed information. Java search engine has JAVA API interfaces like servlets, JSPs. These pages save the results in XML/HTML format.

3.4 Query Construction

Query construction is used for getting appropriate data from data base when user gives wrong input as keyword. It has so many processes [2][6][7] explained briefly. According to given keyword, query should be constructed upto getting exact result.

4. EVALUATION

Java is a programming language[10], with features of inheritance, polymorphism, encapsulation, data hiding, object oriented programming language. Java has one important character as write once and run anywhere as it is platform independent language. Java applications are run on Java virtual machine regardless of computer architecture.

SQL YOG is GUI Tool[9] for easy handling and understanding of SQL commands. SQQL YOG window is divided into 3 sub windows as object browser, SQL window and result window. Object windows shows tables, rows, columns. We can also make our object window show/hidden by selecting these options. Result window will display all the details in table format

The input key words that the search engine is providing access to are added both to the search index and a language model. The language model stores seen words and maintains statistics about them. When a query is submitted, the Spellcheck class looks for possible character edits, namely substitutions, insertions, replacements, transpositions, and deletions, that make the query a better fit for the language model. So if you type 'asemah' as a query, the language model will be much more familliar with the mildly edited 'Mahesh' and also display some related words which are having its sub strings in main string(word).

Fig4 Result page in detailed

Fig 4 represents how the data will be given as input and how the system should response to user. It suggests some keywords to select according to user mentality.

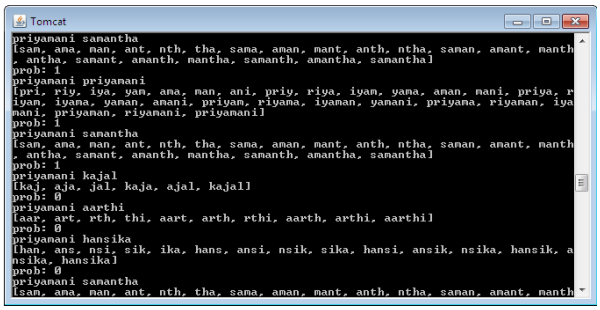
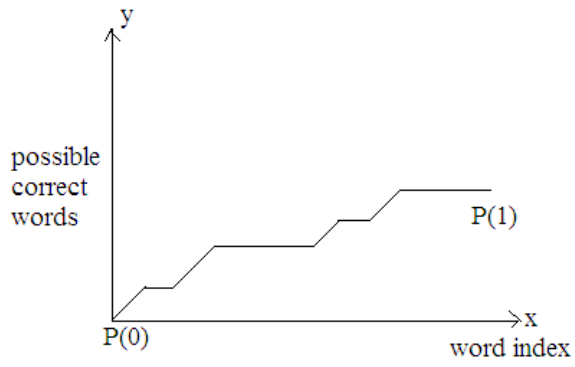


Fig 5. Execution process

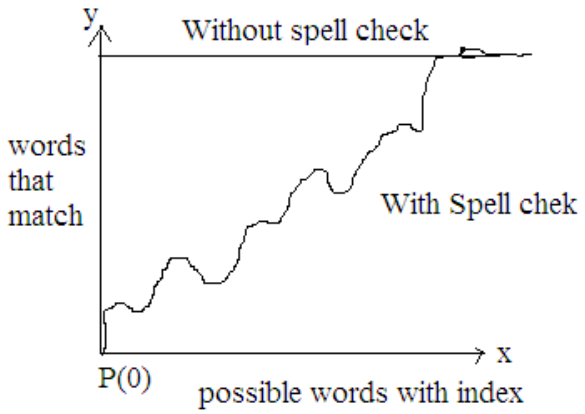
Figure 5 represents how the input key word should be classified and compared within dictionary and display results based on probability wise.

Fig 6. Output results

Figure represents final output how the user gets from search engine. It will display appropriate result.



Graph 1 : Possible words for wrong input data



Graph 2: Performance issue with spell checker and without

Graph 1 explains how the word divided and checks for possible words in dictionary. User can get their accurate word in final state. Graph 2 explains the comparison of present system and previous system. Without spell check line indicates, previous system, it can retrieve data without checking any spellings. A simple line it will shown. With spell checker will take so many steps to get accurate result. In each step it will get some expected word, in final stage user gets accurate data. In clearly, input keyword is passing to search engine for checking availability in data base, if it is in data base, it should display word possibilities, if it is not in data base, again forward to search engine, then search reporter will handle with dictionaries, which holds index files, data files. The wrong input keyword is to be broken based on n gram length(3) word length, then look up into indexed files for availability of matching the words, then it will forward the result to search reporter to search engine , and data base will be updated with details. The big advantage of this approach over dictionary-based spell checking is that the corrections are motivated by data in the search index. On Google, there is no automatic correction, presumably because of huge amount of data in data base. Both Yahoo and Google perform context-sensitive correction. For instance, the query frod (an Old English term from the German meaning wise or experienced) has a suggested correction of ford (the automotive company, among others), whereas the query frod baggins has the corrected query frodo baggins (a 20th century English fictional character). That's the Yahoo behavior. Google doesn't correct frod baggins, even though there are about 785 hits for it versus 820,000 for Frodo Baggins. On the other hand, Google does correct frdo andfrdo baggins. Amazon behaves similarly, but MSN corrects frd baggins to ford baggins rather than frodo baggins.

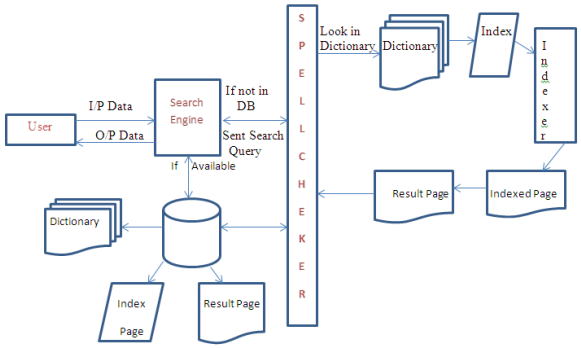


Fig 7. System Architecture

Fig 7 shows the current system architecture, how the search engine works for getting accurate result. It works from searching in dictionaries with appropriate indexes .if it is available in dictionary, the system will display the result page from database.

4.1 Algorithm

PHASE I: TRAINING Set

Model File: SpellCheck.model

N-gram Length: 3

Writing model to file→SpellCheck.model

Writing lucene index to →lucene

Reading model from file→SpellCheck.model

PHASE II: CORRECTION

Constructing Spell Checker from File

5. CONCLUSION

In this paper we improve the performance issue with usability of keyword query. Users need not to know schema knowledge. The development of MySQL Yog to make easier data base operations like insertion, updation, deletion and takes less time to perform operations. Easy to recognize tables as well data entities. In this paper, particularly we aimed to get appropriate data from database. We increase the accuracy of the data as well performance of the system. We believe our system should suggest trust value based on users actions within data base. We presented IQP—a novel system, which enables construction of structured queries from keywords. We presented an algorithm for generating optimal query construction plans, which enables the user to obtain the intended structured query with a minimal number of interactions. Our experimental results and user study show that IQP is highly helpful when user intended structured queries cannot be found within the top-ranked results.

6. FUTURE SCOPE

We would develop the code for storing the False Hit Keywords given by the Users. False hit Keyword means the keywords which are not available in the existing database table. Those keywords will be stored in separate table and these keywords will be shown to the Admin page for updating the details if the database to be updated by the user.

7. ACKNOWLEDGMENT

Our thanks to the management members and principal of Kakatiya Institute of Technology and Science-Warangal who have facilitated resources to read and compute in order to develop this model and narrate this article and our sincere thanks to Head of the Department Prof.P.Niranjan who encouraged us research and publish this paper.

8. REFERENCES

[1] Probabilistic scheme for keyword based incremental query construction - Elena Demidova, Xuan Zhou, and Wolfgang Nejdl, Member, IEEE Computer Society.

[2] G. Zenz, X. Zhou, E. Minack, W. Siberski, and W. Nejdl, "From Keywords to Semantic Queries—Incremental Query Construction on the Semantic Web," J. Web Semantics, vol. 7, no. 3, pp. 166-176,2009.

[3] J.R. Quinlan, "Induction of Decision Trees," Machine Learning,vol. 1, no. 1, pp. 81-106, Mar. 1986.N Grams.

[4] Probabilistic language models, http://dingo.sbs.arizona.edu/~hammond/ling178-sp06/mathCh8.pdf

[5] Spelling correction using N Grams, http://polibits.gelbukh.com/2009_40/40_06.pdf

[6] X. Zhou, G. Zenz, E. Demidova, and W. Nejdl, "SUITS: Constructing Structured Queries from Keywords," technical report, L3S Research Center, Hannover, Germany, http://www.l3s.de/~demidova/suits-TR.pdf, 2008.

[7] Q. Zhou, C. Wang, M. Xiong, H. Wang, and Y. Yu, "SPARK: Adapting Keyword Query to Semantic Search," Proc. Sixth Int'l the Semantic Web and Second Asian Conf. Asian Semantic Web Conf. (ISWC), 2007.

[8] An Improved Error Model for Noisy Channel Spelling Correction Eric Brill and Robert C. Moore Microsoft Research One Microsoft Way Redmond, Wa. 98052 {brill,bobmoore}@microsoft.com.

[9] D. Braga, A. Campi, and S. Ceri, "XQBE (XQuery By Example): A Visual Interface to the Standard XML Query Language," ACM Trans. Database Systems, vol. 30, no. 2, pp. 398-443, 2005.

[10] K. Arnold and J. Gosling, The Java Programming Language, 1996: Addison Wesley.

9. AUTHOR'S PROFILE

M.Sandhya Currently persuing Master of Technology in Computer Science and engineering with specilization in Softwatre Engineering.Computer Science and Engineering Department, Kakatiya Institute of Technology & Science (KITS), Kakatiya University-Warangal.A.P.,India.

B.Hanmanthu obtained his Bachelor's degree in Computer Science and Engineering from JNT University of India. Then he obtained his Master's degree in Computer Science and Engineering with specialization Software Engineering from JNT University Hyderabad, and he is also life member of ISTE. He is currently Assistant Professor of Computer Science and Engineering, Kakatiya Institute of Technology & Science (KITS), Kakatiya University-Warangal. His specializations include Data mining and Data warehousing, Databases and networking.

B.Raghuram obtained his Bachelor's degree in Computer Science and Engineering from JNT University of India. Then he obtained his Master's degree in Computer Science and Engineering from Pondicherry Central University Pondicherry, and he is also life member of ISTE. He is currently Assistant Professor of Computer Science and Engineering, Kakatiya Institute of Technology & Science (KITS), Kakatiya University-Warangal. His specializations include Data mining and Data warehousing, Databases and networking.