Context-Aware Spelling Corrector for Sentiment Analysis

Fazal Masud Kundi[1], Aurangzeb Khan[2], Muhammad Zubair Asghar[1], Shakeel Ahamd[1]

[1]Institute of Computing and Information Technology, Gomal University, Pakistan
[2]Institute of Engineering and Computer Sciences, University of Science and Technology Bannu, Pakistan

Abstract: *One of the most thrived features of the Web 2.0 era is the fastest growing of user-generated content in the shape of blogs and reviews, with unmatched speed and size. These reviews contain poor, text quality and structure which results spelling mistakes as well as out-of-vocabulary words. This paper presents a Context-Aware Spelling Corrector for Sentiment Analysis based on similarity measures and statistical language model. The paper also presents some compelling statistics about spelling errors. The comparative results show that the proposed framework outperforms the related systems, features wise and in accuracy.*
Key words: *Spelling Corrector, Context-aware, Language Model, Sentiment Analysis, Similarity measures*

1. Introduction

The Web 2.0 has dramatically changed the way of producing and consuming information. Emergence of the social network services are the significant impact of social Web. Social media sites became a world's largest simulated community where people express their views about products, events or services globally (Fazal MK et al, 2014). Textual information in shape of reviews and blogs are generated with unmatched speed and size full of opinionated text. The influence of the social media on people communications is evident from the fact that the Oxford dictionary added over 1000 new words and meaning including words used in the Web environment such as 'lolz' and 'tweeps' (Oxford Dictionary, 2014).

Most of the user's comments and reviews are generated without any regard to the general rules and standards of any language, due to which the text is poorly written, have spelling mistakes and contains out-of-vocabulary words. On Twitter, almost one out of every 150 English words is spelt incorrectly. Facebook users write just one in every 323 words incorrectly and one in every 238 by Google+ users (The Telegraph, 2014).

Recently, research has focused on developing algorithms which are capable of recognizing a misspelled word itself is in the vocabulary, based on the context of the surrounding words. Majority of the typographical errors (80 to 95%) found in very large text documents differ from the correct spelling in one of the following four ways (Damerau FJ,1964; Pollock JJ and Zamora A, 1984): One letter mistyped (sentoment), omitted (sentment), inserted (sentimment) or transposition of two adjacent letters (sentiment).

This paper presents a context-aware spelling corrector for sentiment analysis (CASC). The framework uses hybrid approach of similarity measures to generate a candidate list of words and statistical language model (noisy channel) for choosing most likely spelling correction.

2. Related Work

The task of spelling correction has a long and interesting history; more than three decades passed on research of detection and correction of spelling errors (Peterson JL, 1980a, Peterson JL, 1980b). Looking up every word in a dictionary for detecting errors is the most popular method; any word not present in the dictionary is taken as error (Spellchecking, 2014). Risen et al. (Riseman EM and Hanson AR, 1974) used dictionary indirectly by generating table of trigrams of all dictionary words. Using this table the spelling checker divides the target text into trigrams and searches them in the table; if any trigram is not found the word is taken as misspelled. This technique has limitation due to the low proportion of any impossible (not present in the table) trigrams. The method proposed in (Morris R and Cherry LL, 1975) does not use a dictionary at all, rather it divides the text into trigrams, and calculate index of peculiarity for each word based on trigrams. The advantage of this method is its Language-independency and spotting typing errors but it would fail to identify a high proportion of ordinary spelling errors.

The majority of spelling checkers are dictionary based. To save the storage space the method presented by (McIlroy MD, 1982) stores only the

(DOI: dx.doi.org/14.9831/1444-8939.2014/2-5/MAGNT.1)

stems of words. This system can accept new words that are acceptable in the text but at the same time it can accept some words that does not exist. The spelling checker can cause two types of errors: identifying a word as incorrect when in fact it is correct for example all proper nouns and identifying incorrect word as correct. The use of larger dictionaries can be used to reduce this false positive rate but no real solution exists for handling proper noun. Many spelling checkers enhance the dictionary with additional words to minimize the Type 1 error (Spellchecking, 2014). Problems become more complicated when the misspelled word matches the dictionary word, as in "*Their are two books*". Such type of problems exist mostly with larger dictionaries due presence of large number of obscure words. Small dictionary also raises too many false alarms.

When an uncommon word appears in a text it has much more possibility to be a correct spelling of a rare word than a misspelling of other word (Damerau FJ and Mays E, 1989). Sixteen percent of typing errors produce another dictionary word for instance mistype of word "*bed*" can produce "*bad*", "*bud*", "*bod*" and so on (Peterson JL, 1986). When spelling errors as well as typing errors are handled at the same time the problem becomes much more alarming.

There are two aspects of the spelling checker and corrector; producing correct spelling and deciding which word was intended. People face trouble in correcting the spelling but feel easy to select the suitable word in most of the cases. For example someone writes "*sychology*" and checker shows error flag. If the user does not know how to spell "*psychology*" he will stuck, but in the sentence "*I went water*", the human can easily know which word was intended "*want*" or "*went*". In contrast it is very easy for computer to retrieve a correct spelling from dictionary but very hard to decide about the intended word (Spellchecking, 2014). A wide reviews of the literature about spellchecker are given in (Peterson JL, 1980a/1980b; Kukich K, 1992a/1992b). In the following lines we present some recent research literature reviews of spellchecker.

Comparison of spelling corrector for mobile instant messages for N-Gram similarities is presented in (Butgereit L and Botha RA, 2013). Four similarity measures (Jaccard, Cosine, Sorensen and Overlap) were investigated and evaluated using historical data of mathematical terms. They achieved 83-90% accuracy on different similarity measures. Spelling corrector for Web sentiment analysis that handles cross-word errors was presented by (Jadhav SA et al. ,2013). Two datasets of tweets named "*barack Obama*" and "*microsoft*" were used in this work and achieved maximum accuracy of 91.26%. M. Kim et al. (Kim M et al. 2103) presents "Statistical Context-Sensitive Spelling Correction" using confusion sets. Confusion sets help in finding and correcting context-sensitive spelling errors using conditional probability based reliability between each word.

This study proposes a context-sensitive spelling corrector for sentiment analysis. The framework uses hybrid approach of similarity measures to generate a candidate list of words and statistical language model (noisy channel) for choosing most likely spelling correction.

3. Proposed Framework

The proposed framework for detecting and correction of typographical errors is depicted in Fig 1. It consists of three major modules.

3.1 Tokenization and Error detection

In order to retrieve sentence structure and its complete sense, it is needed to break the sentence into small parts called tokens (Muhammad ZA et al, 2013).This is the initial module of the framework used to break up the sentences into tokens (words) and identifies the typographical errors (focal words) using dictionary.

3.2 Generating Candidates

Jaccard index and Levenshtein distance (Edit distance) were used to generate the candidates list. In the first phase every dictionary and focal word was expressed into uni-grams or bi-grams depends on Edit distance. If the Edit distance was greater than one, word was expressed into uni-gram otherwise into bi-grams.

$$n-grams(w1,w2)=\begin{cases} uni-grams(w1,w2) & if\ lev_{W1,W2}(m,n)>2 \\ bi-grams(w1,w2) & otherwise \end{cases} \quad (1)$$

Where $lev_{W1,W2}(m,n)$ is Levenshtein distance, m = |*w1*| and n = |*w2*|.

The candidate list is further pruned by filtering some candidates to improve the performance of language model. Following function is used for pruning.

$$cfilter(cw, flc) = \begin{cases} remove(cw) & if\ cw \notin DB \\ remove(cw)\ if\ fc(cw) = fc(w_e) and\ flc = True & \\ cw & otherwise \end{cases} \quad (2)$$

Where *cw and* w_e represent the candidate and misspelled words respectively. The second argument *flc* is used to specify whether the first letter of the misspelled word is correct? The function *fc()* extracts the first letter of the word.

In the second phase Jaccard index was employed to calculate the similarity index between dictionary words and focal word to generate n candidates. The above eq. (1) uses Edit distance to take advantage of uni-grams due to the high coverage with all similarity measures when edit distance is greater than one. Jaccard index can minimize the number of candidates by taking top n measures. Other similarity measures serve as base-line. The framework works also with the assumption that the first letter of the mistype word is correct, because it has been found that the first letter is usually correct (Yannakoudakis EJ and Fawthrop D, 1983).

3.3 Language model

This module selects the best choice among the candidate words using statistical language model. A best choice is that one which has highest noisy channel probability.

S_e: Sentence with typographical error
S_i: Sentence intended by writer
S: Intended sentence with highest likelihood

Where *cw and* w_e represent the candidate and misspelled words respectively. The second argument *flc* is used to specify whether the first letter of the misspelled word is correct? The function *fc()* extracts the first letter of the word.

In the second phase Jaccard index was employed to calculate the similarity index between dictionary words and focal word to generate n candidates. The above eq. (1) uses Edit distance to take advantage of uni-grams due to the high coverage with all similarity measures when edit distance is greater than one. Jaccard index can minimize the number of candidates by taking top n measures. Other similarity measures serve as base-line. The framework works also with the assumption that the first letter of the mistype word is correct, because it has been found that the first letter is usually correct (Yannakoudakis EJ and Fawthrop D, 1983).

3.4 Language model

This module selects the best choice among the candidate words using statistical language model. A best choice is that one which has highest noisy channel probability.

S_e: Sentence with typographical error
S_i: Sentence intended by writer
S: Intended sentence with highest likelihood

$$S = \arg\max_{S_i} P(S_i|S_e)$$

$$= \arg\max_{S_i} \frac{P(S_e|S_i)}{P(S_e)}$$

$$= \arg\max_{S_i} P(S_e|S_i).P(S_i) \quad (3)$$

This work is based on Bigram Language Model (BLM). Eq. (4) and (5) present, before BLM and after BLM respectively for sentence of *m* words with w_f focal word. Algorithm of CASC is shown in Fig 2.

$$P(w_1, w_2, \dots \dots w_f, \dots \dots \dots w_m) = P(w_{f-1}w_f) = \frac{n(w_{f-1}w_f)}{N} \quad (4)$$

$$P(w_1, w_2, \dots \dots w_f, \dots \dots \dots w_m) = P(w_f w_{f+1}) = \frac{n(w_f w_{f+1})}{N} \quad (5)$$

Where $n(w_{f-1}w_f)$ and $n(w_fw_{f+1})$are the number of times that bi-grams appeared in the source text.

4. Experimental Setup

4.1 Datasets

Following datasets were used in this research work. (i) Hotel reviews dataset (Natural Language, 2014), which contains 3000 reviews (1500 positive and 1500 negative) (ii) OpinRank review dataset of cars and hotels reviews collected from Tripadvisor and Edmunds (Ganesan K and Zhai C, 2012). The hotels dataset contains full reviews of hotels in 10 cities. (iii) Artificial dataset of 100000 misspelled words generated from dictionary. Python "random()" function was used to generate the data differs in four ways (insertion, deletion, substitution or transposition) from the correct data. (iv) Natural language corpus downloaded from (Corpus, 2014).

4.2 Similarity Measures

Following similarity measures were used in this study.

Jaccard Index

The Jaccard index (Jaccard P, 1901), also also known as Jaccard similarity coefficient is a single measure used to calculate the similarity and diversity of sets. Jaccard similarity coefficient between two words (string of characters) can be calculated as follows:

$$J(W1, W2) = \frac{|W1 \cap W2|}{|W1 \cup W2|} \qquad (6)$$

Where $0 \leq J(W1, W2) \geq 1$

Cosine Similarity

Cosine similarity (Salton G, 1989) measures the cosine of the angle between two vectors. Similarity 1 means same orientation and 0 similarity means that the angle between vectors is 90^0. Cosine similarity between two words can be calculated as follows:

$$Cosine(W1, W2) = \frac{|W1 \cap W2|}{\sqrt{|W1||W2|}} \quad (7)$$

Sørenson-Dice

Sørenson-Dice (Manning CD, 999) is a statistic used to compare the similarity between two sets. Sørenson-Dice can be calculated using the following formula:

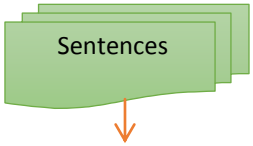
$$SD(W1, W2) = \frac{2|W1 \cap W2|}{|W1|+|W2|} \qquad (8)$$

Levenshtein distance

Levenshtein distance (Nerbonne J et al., 1999) is a string distance function also known as Edit distance. It takes two inputs and return value equivalent to the number of substitutions and deletions needed to transform one input string into another. The Edit distance between two words *W1* and *W2* is the minimum number of single-character edits (i.e. insertions, deletions or substitutions) required to change *W1* into *W2* or vice versa. Mathematically it is defined as follows:

$$lev_{W1,W2}(m,n) = \begin{cases} \max(m,n) & if\ \min(m,n) = 0 \\ min \begin{cases} lev_{W1,W2}(m-1,n-1) + 1 \\ lev_{W1,W2}(m,n-1) + 1 \\ lev_{W1,W2}(m-1,n-1) + 1\ (W1_m \neq W2_n) \end{cases} & otherwise \end{cases} \qquad (9)$$

Where $m = |W1|$, $n = |W2|$, $1(W1_m \neq W2_n)$ is the indicator function and equal to zero when $(W1_m = W2n$, equal to 1 otherwise.



(DOI: dx.doi.org/14.9831/1444-8939.2014/2-5/MAGNT.1)

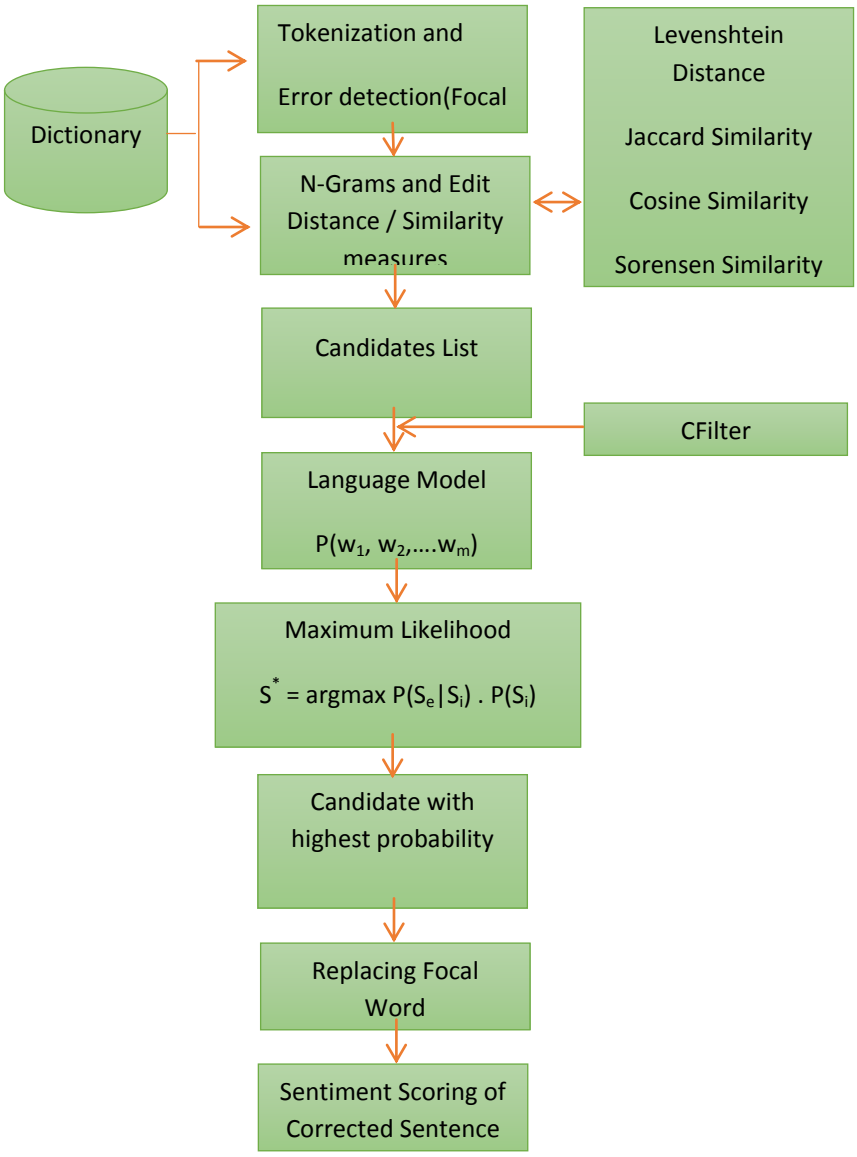


Fig. 1: Context-aware Spelling Corrector

4.3 Performance Evaluation

Precision, recall and F-score are the most widely performance measures to evaluate the stability of the classifier. Confusion matrix (Provost FJ et al. 1998) also known as error matrix is a tool used for prediction of classifier results. Table 1 shows the confusion matrix for binary classification. The purpose of these measures in this study is to measure the impact of spelling correction on accuracy of sentiment classification.

Fig. 2: Algorithm of CASC

Table 1. Confusion Matrix for Binary Classification

		Machine Says	
		Positive	**Negative**
Human Says	**Positive**	TP	FN
	Negative	FP	TN

True Positive (TP): Number of positive cases classified correctly.
False Positive (FP): Number of negative cases classified incorrectly as a positive.
True Negative (TN): Number of negative cases classified correctly.
False Negative (FN): Number of positive cases classified incorrectly as a negative.

Precision

Precision (Olson DL and Delen D, 2008) also called positive predicted value, measures the correctness of the model. Higher precision indicates less FP. Mathematically it is defined as:

$$Precision\ , P = \frac{TP}{TP+FP} \quad (10)$$

Recall

Recall (Olson DL and Delen D, 2008) also known as sensitivity, measures positive cases correctly classified by the model, large recall value means few positive cases misclassified as a negative. Recall can be calculated using the following formula.

$$Recall, R = \frac{TP}{TP+FN} \quad (11)$$

F-Score

F-score or F1-measure (Olson DL and Delen D, 2008) is the harmonic mean of precision and recall. F-score can be calculated as follow:

$$F - Score = \frac{2rp}{r+p} = \frac{2TP}{2TP+Fp+FN} \quad (12)$$

5. Results and Discussion

We performed wide range of experiments on misspelled words for getting empirical evidence of the performance of the proposed framework. The proposed framework was evaluated in two ways: (i) Coverage and number of candidates. If the candidate list has large number of intended words then it has high coverage. Small number of candidates contributes to efficiency. (ii) Accuracy of the language model in selecting intended word.

Table 2 shows the coverage of three different similarity measures. It was observed that Bi-grams has highest coverage in first three type but low coverage in case of "transpose", where Uni-grams has highest coverage for all three measures. Jaccard and Sorensen-Dice have same results for all types. Levenshtein distance (edit 2) has word coverage of 96% but generates a very large number of candidates. For example edit 2 generates 179 candidates on average with standard

(DOI: dx.doi.org/14.9831/1444-8939.2014/2-5/MAGNT.1)

deviation of 137, which is computationally expensive for the language model to evaluate the sentence for the intended word. Table 3 shows the word coverage (90 to 94%), number of suggested candidates (15-40), accuracy of the framework in selecting appropriate word and comparative performance. The framework has a capability to reduce the number of candidates up to 89% using the candidates filter, because it has been found that the first letter is usually correct, as statistics are shown in table 4.

Table 2. Coverage of Three Similarity Measures

	Cosine		Jaccard		Sorensen-Dice	
	Uni-grams	**Bi-grams**	**Uni-grams**	**Bi-grams**	**Uni-grams**	**Bi-grams**
Insertion	0.8785	0.9452	0.7120	0.9507	0.7120	0.9507
Deletion	0.7661	0.8354	0.6640	0.8463	0.6640	0.8463
Replace	0.5322	0.7941	0.4133	0.8222	0.4133	0.8222
Transpose	1.0	0.4201	0.9682	0.4516	0.9682	0.4516
Mean	0.7942	0.7487	0.6894	0.7677	0.6894	0.7677

Table 3. Comparative Performance of CASC

Method	**Coverage (%)**	**Suggested Words**	**Accuracy (%)**
Base line	78	15	70
(SC Model 2014)	--	--	86.6
(Jadhav SA et al, 2013)	--	--	91.26
CASC	90 to 94	15 - 40	92

Table 4. Statistics of Mistype Words

Source	**Words**	**First Letter is Correct (%)**
Artificial Dataset	12392	82
Artificial Dataset	124930	90
(NL Corpus, 2014)	7841	96
Mean		89

Table 5. Impact of Spelling Correction on Sentiment Analysis (Hotel Reviews)

	Precision	**Recall**	**Accuracy**
W/O Correction	0.79	0.85	0.75
Corrected	0.80	0.85	0.80

The number of candidates can be reduced further by excluding all candidates not found in the database of source text. It was observed during this study that 25% candidates can be reduced in this way. Finally table 5 shows the impact of spelling correction on sentiment analysis using hotel reviews.

6. Conclusion and Future Work

Textual information in shape of reviews and blogs are generated with unmatched speed and size full of opinionated text (Muhammad ZA et al, 2013; Muhammad ZA et al, 2014). In this paper we proposed a framework for context-aware spelling corrector for sentiment analysis and achieved satisfactory results in both, candidate's generation and context. The existing work can be enhanced in many ways. Error detection can be expanded to other type of spelling errors such as homophones. Candidate generation process can be improved by including some other sources of information such phonetic and semantic properties. Different other statistical language models can be applied with some feedback to improve the accuracy context-wise.

7. REFERENCES

1. Butgereit L and Botha RA. A comparison of different calculations for N-gram similarities in a spelling corrector for mobile instant messaging language. In *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference*, pp. 1-7, (2013).

2. Kundi, FM., Ahmad, S., Khan, A., & Asghar, M. Z. Detection and Scoring of Internet Slangs for Sentiment Analysis Using SentiWordNet. Life Science Journal, **vol. 11 no. 9 (2014)**.

3. Damerau FJ. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, **Vol. *7***, No. 3, pp. 171-176, (1964).

4. Damerau FJ and Mays E. An examination of undetected typing errors. *Information Processing and Management*, **Vol. 25**, No. 6, pp. 659-664, (1989).

5. Ganesan K and Zhai C. Opinion-based entity ranking. *Information retrieval*, **Vol. 15**, No. 2, pp. 116-150, (2012).

6. Jaccard P. Distribution de la flore alpine dans le bassin des Dranses et dans quelques régions voisines. Bulletin de la Société Vaudoise des Sciences Naturelles **37**, pp. 241-272, (1901).

7. Asghar, M. Z., Khan, A., Ahmad, S., & Kundi, F. M. PREPROCESSING in natural language processing. Editorial board, **pp. 152 (2013).**

8. Jadhav SA et al. Topic dependent cross-word Spelling Corrections for Web Sentiment Analysis. In *Advances in Computing, Communications and Informatics (ICACCI), 2013 International Conference on*, pp. 1093-1096, (2013).

9. Kim M, Jin J, Kwon HC, and Yoon A. Statistical Context-Sensitive Spelling Correction Using Typing Error Rate. In *Computational Science and Engineering (CSE), 2013 IEEE 16th International Conference on*, pp. 1242-1246, (2103).

10. Kukich K. Spelling correction for the Telecommunications Network for the Deaf. *Communications of the A.C.M.* **Vol. 35**, No. 5, pp. 80-90, (1992).

11. Kukich K. Techniques for automatically correcting words in text. *Computing Surveys*, **Vol. 24**, No. 4, pp. 377-439, (1992).

12. Manning CD. Foundations of statistical natural language processing. MIT press, (1999).

13. McIlroy MD. Development of a spelling list. *IEEE Transactions on*

Communications, **Vol. COM-30**, No. 1, pp. 91-99, (1982).

14. M. Zubair Asghar, Aurangzeb Khan, Shakeel Ahmad, Fazal Masud Kundi, "A Review of Feature Extraction in Sentiment Analysis", Journal of Basic and Applied Scientific Research, **vol. 4** no. 3, pp. 181-186, **(2014)**.

15. Morris, Robert and Cherry, Lorinda L. Computer detection of typographical errors, *IEEE Trans Professional Communication*, **Vol. PC-18**, No. 1, pp. 54-64, (1975).

16. Natural Language Corpus: http://norvig.com/ngrams/, (2014).

17. Natural Language Engineering Lab: http://www.dsic.upv.es/grupos/nle/resources/corpusPM.zip, (2014).

18. Nerbonne J, Heeringa W and Kleiweg P. Edit distance and dialect proximity. Time Warps, String Edits and Macromolecules: The Theory and Practice of Sequence Comparison, 2nd ed., pages 15, (1999).

19. Olson DL and Delen D. Performance Evaluation for Predictive Modeling in Advanced Data Mining Techniques. Springer-Verlag, Berlin Heidelberg, pp. 137-139, (2008).

20. Oxford Dictionary Adds "Retweet,": http://www.mobiledia.com/news/103738.html, (2014).

21. Peterson J L. Computer programs for spelling correction: an experiment in program design. Springer Berlin Heidelberg, pp. 1-129, (1980).

22. Peterson JL. Computer programs for detecting and correcting spelling errors. *Communications of the ACM*, **Vol. *23***, No. 12, pp. 676-687, (1980).

23. Peterson JL. A note on undetected typing errors. *Communications of the A.C.M.*, **Vol. 29**, No. 7, pp. 633-637, (1986).

24. Pollock JJ and Zamora A. Automatic spelling correction in scientific and scholarly text. *Communications of the ACM*, **Vol. 27**, No. 4, pp. 358-368, (1984).

25. Asghar, M. Z., Qasim, M., Ahmad, B., Ahmad, S., Khan, A., & Khan, I. A. (2013). Health miner: opinion extraction from user generated health reviews. International Journal of Academic Research, vol. 5 no. 6 (2013).

26. Provost FJ, Fawcett T and Kohavi R. The case against accuracy estimation for comparing induction algorithms. In Proceedings of the Fifteenth International Conference on Machine Learning, pp. 445-453, (1998).

27. Riseman EM and Hanson AR. A contextual post-processing system for error correction using binary n-grams. *IEEE Trans Computers*, **Vol. C-23**, No. 5, pp. 480-493, (1974).

28. Salton G. Automatic text processing: the transformation, analysis, and retrieval of information by computer. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, (1989).

29. Spellchecking by computer: http://www.dcs.bbk.ac.uk/~roger/spellchecking.html. (2014).

30. Spelling Correction Models: http://dustwell.com/PastWork/SpellingCorrectionLanguageModels.pdf

31. The Telegraph: http://www.telegraph.co.uk/technology/twitter/10086819/Twitter-users-cant-spell.html. (2014).

32. Yannakoudakis EJ, and Fawthrop D. The rules of spelling errors. *Information Processing and*

Management, **Vol. 19**, No. 2, pp. 87-99, (1983).