TELKOMNIKA, Vol.17, No.1, February 2019, pp.95~102
ISSN: 1693-6930, accredited First Grade by Kemenristekdikti, Decree No: 21/E/KPT/2018
DOI: 10.12928/TELKOMNIKA.v17i1.10183

Comparison of stemming algorithms on Indonesian text processing

Afian Syafaadi Rizki*[1], Aris Tjahyanto[2], Rahmat Trialih[3]
[1,2]Department of Information System, Institut Teknologi Sepuluh Nopember
FTif Building, ITS Raya Road, Surabaya 60111, Indonesia
[3]Department of Information System, University of Brawijaya
FILKOM Building, 8 Veteran Street, Malang 65145, Indonesia
*Corresponding author, e-mail: afianrizki@gmail.com[1], atjahyanto@gmail.com[2], rtrialih@gmail.com[3]

Abstract

Stemming is one of the stages performed on the process of extracting information from the text. Stemming is a process of converting words into their roots. There is an indication that the most accurate stemmer algorithm is not the only way to achieve the best performance in information retrieval (IR). In this study, seven Indonesian stemmer algorithms and an English stemmer algorithm are compared, they are Nazief, Arifin, Fadillah, Asian, Enhanched confix stripping (ECS), Arifiyanti and Porter. The data used are 2,734 tweets collected from the official twitter account of PLN. First, the aims are to analyze the correlation between stemmer accuracy and information retrieval performance in Indonesian text language. Second, is to identify the best algorithm for Indonesian text processing purpose. This research also proposed improved algorithm for stemming Indonesian text. The result shows that correlation found in the previous research does not occur for the Indonesian language. The result also shows that the proposed algorithm was the best for Indonesian text processing purpose with weighted scoring value of 0.648.

Keywords*: confix stripping stemmer, Indonesian stemmer, information retrieval, text clustering*

1. Introduction

Stemming is a process of converting a word into its root [1]. Stemming was commonly used in application developments, especially in the case of IR [2]. In addition, stemming was widely used in text mining, and natural language processing systems to improve its performance [3]. Stemming can improve the performance of text processing because this process will merge words which have the same root. If words have the same roots, they are considered to have a semblance of meaning. Therefore, documents that have words with the same roots are considered relevant so the stemming process will reduce the features dimension of the documents [4]. The stemming algorithms were divided into two types. They were statistic-based stemmer and rule-based stemmer. Statistic-based stemmer was an unsupervised algorithm that uses training data to form models in performing stemming, while rule-based stemmer was an algorithm that uses a number of rules that have been defined to perform stemming. The advantage of statistic-based stemmer was it can be applied to different languages more easily, allowing for multilingual stemming. The drawback of statistic-based stemmer was it requires large amounts of data to get good accuracy. So it is less suitable for small amounts of data and unable to handle words outside statistical models [5].

Comparing stemming algorithms had been done before. Brychcín and Konopík [5], proposed a statistical-based stemmer algorithm called High Precision Stemmer (HPS). It was compared with graph-based stemmer (GRAS), YAAS, Linguistica, and some Rule-based Stemmer. Flores and Moreira [6], comparing stemming algorithms for four languages. They were English, French, Portuguese and Spanish. The conclusions of the study show that the most accurate stemmer algorithm was not the only way to gain improvement in the IR case. The experiment results also show that low stemmer although simple but it can also provide improved performance. Further research can be done to find out if the correlation found in that study will apply to other languages that have different characteristics [6].

In this research, the researchers test the correlation between stemmer performance and IR performance to find out whether the correlation found in previous studies [5] will apply to the

Received June 22, 2018; Revised October 24, 2018; Accepted September 12, 2018

Indonesian language. The researchers also compared the Indonesian stemmer algorithms with the aim to identify the most accurate algorithm, and the best algorithm for IR in Indonesian text. The weighted scoring method [7] was used to select which was the best stemmer algorithm for Indonesian text processing purpose. Justification of weighting values will be explained in more details in the research methods section. The results of this research can become a source of study in the development of IR applications in Indonesian language.

The stemmer algorithms used in this research were: Nazief algorithm [8], Arifin algorithm [9], Fadillah algorithm [10], Asian algorithm [11], ECS [12], Arifiyanti algorithm [13], and Porter confix stripping algorithm [14]. Most of Indonesian stemmer algorithms uses the rules of cutting the word, so the algorithms are classified as rule-based stemmer. Porter confix stripping algorithm was also tested in this research to know how its effect on Indonesian language text. This algorithms is a stemmer algorithms for English language that was used as a basis in developing Indonesian stemmer algorithms.

Early stemming algorithm for Indonesian language still had some errors [8], so there were some research to improve the stemmer performance. Asian [11] has proved that the confix stripping algorithm had the best results, although there are still shortcomings. Asian stemmer algorithm added rules to deal with plural words and rules to remove prefixes first instead of suffixes. Arifin et al, [12] revised the rules for prefixes and named the algorithm the enhanced confix stripping while Arifiyanti [13] added rules for nonstandard languages. However, there were some terms which could not be transformed into their root using Arifiyanti algorithm. Therefore, the researchers proposed stemmer algorithm which was a refinement of Arifiyanti algorithm. To compare Indonesian stemmer algorithms, it was used several measurements. They were davies bouldin index value (DBI) [15] for measuring clustering performance, overstemming and understemming values [16], and data processing time. The data used in this research were twitter data from State Electricity Company (PLN). PLN is a company owned by the government of the Republic Indonesia which was engaged in electricity industry in Indonesia.

2. Related Works and Proposed Stemmer Algorithm

This section will discuss previous studies related to comparing the stemming algorithms. Secondly, it discussed the Indonesian stemmer algorithms and the proposed algorithm.

2.1. Indonesian Stemmer Algorithms

Stemmer algorithm for Indonesian language has been developed by Nazief [8]. The nazief stemmer algorithm was developed based on the porter algorithm [14]. This algorithm was done by applying some affix removal rules to the word which its root was to be searched. The affixes were grouped into four categories: inflection particles, possessive pronouns, derivation suffixes, and derivation prefixes. This algorithm uses word root dictionary to validate the stemming results.

Arifin and Setiono [9] proposed a stemmer algorithm similar to Nazief, but with the addition of a process which combines affix to the word which executed if the entire affix removal rules were failed. The results show that this algorithm was effective to find the word root. Unlike nazief and arifin algorithm, Fadillah [10] proposed a new stemmer algorithm that did not use word root dictionaries. It used syllable numbers to check whether the word has been converted to its root. That study conducted a comparison between the proposed algorithms with the Nazief algorithm. The results show that the accuracy of the proposed algorithm was lower, but the processing time was much faster [10].

Asian [11] not only compared Indonesian stemmer algorithms, but also Malay stemmer algorithms. The researcher added some affixes removal rules, plural word rules and a rule precedence on the nazief algorithm. The conclusion was that the stemmer that used dictionary had better performance. From the test results it was also known that most of the errors were caused by the roots which did not exist in the dictionary. Arifin et al. [12] refined the Asian algorithm by adding some prefix removal rules. In addition, they also added a process of returning suffixes which was similiar with their previous work [9]. The new algorithm was called enhanched confix stripping (ECS) stemmer. The results of their study showed an increase in data classification performance compared with using the Asian algorithm.

Arifiyanti [13] also refined the Asian algorithm. She added several new rules for the affixes removal and rule precedence in the Asian algorithm. The new added rules were adjusted for semi-structured Indonesian language in twitter data. The results show that her proposed stemmer algorithm provides improved performance in the IR.

2.2. Proposed Stemmer Algorithm

The Arifiyanti algorithm [13] was developed based on the Asian algorithm [11]. In this study, some new rules were added in rule precedence checking step into Arifiyanti algorithm. The new rules were *be-i, di-pun, se-pun, s-an, di-nya*, and *te-ah*. For the affixes removal process and recoding process, the rules were not changed. This modified Arifiyanti algorithm is called proposed stemmer algorithm.

3. Research Methods

This section discusses the methods used in this research. They were methods to perform data pre-processing, data clustering and methods to evaluate the results. The data were collected from the official twitter account of PLN, @pln_123. The data were collected from August to September 2016. After cleaning retweets, the data were divided into two. First, the data for clustering purposes named the tweet’s data which contain 2,734 tweets. Second, were the group data which contain 936 words and compiled into 200-word groups. The group data were created manually by selecting and grouping words from the tweet’s data which have the same root.

The tweet’s data were processed using data pre-processing methods to minimize noise. Data noise would cause the process to become more difficult, and longer [17]. Therefore the following steps were executed. First, account names were deleted, then tokenizing process was done to break the sentence into words. Next, it was performed casefolding and words normalization. For normalization step, the naraadhipa method was used [18]. After the words were normalized, stopwords removal was performed and stemming processes was done. All eight algorithms including the proposed algorithm used in these stemming processes. Furthermore, the tf-idf method was applied and the clustering algorithm was performed. For the data clustering, k-means algorithm was used. K-means was very often used because it was an efficient and simple method [19]. In this research, some k values ranged from 2 to 20 were used on k-means parameter. After the clustering step was done, the result was evaluated using the DBI method to get the DBI values.

Meanwhile, the group data were handled separately. Only the stemming processes applied to the data and the results were evaluated using Paice method [16] to get the overstemming and understemming values. In order to measure the significance of using stemmer algorithms in data clustering, it was performed t-test with a confidence level of 95% against DBI values. Afterward, the correlation between the results obtained from the Paice evaluation method and the performance of data clustering was tested using the Pearson method [20].

Finally, the weighted scoring was performed after other tests and evaluations were finished. This method used to determine the best stemming algorithm using several criteria, they were understemming value, overstemming value, process time, and DBI value. To determine the weight value of each criterion, it needs justifications based on several past research which discussed in the following paragraphs.

Stemmer algorithms were usually used in development of IR [2]. Those were also broadly used to increase performance of text mining and natural language processing systems [3]. While Dalwadi et al. [21] had stated that the development of their algorithm could be useful dictionary search and machine translation. Another study stated that stemmer’s algorithms were widely used in IR systems [22], word processing, and spelling checkers [23]. It can be concluded that IR is the main concern, so we gave a 45% weight score for the DBI value because it is representing the IR. Overstemming and understemming justification were based on Flores and Moreira’s study [6]. They were conducting correlation test between understemming, overstemming, and mean average precision (MAP). The results showed that overstemming was negatively significant for most languages. While the value of understemming correlated positively insignificant. It can be concluded that overstemming error was more serious than understemming error. Hence, we gived 25% weight score for overstemming and

20% for understemming. The study that measured processing time and data compression was performed by Jaafar et al. [2], while Mahmud et al. [24] were discussing about the processing time and system memory used. Not many studies focused on processing time, so the processing time was weighted by 10%. After assigning a weight value to each criterion, it was normalizing using normalization function [25].

4. Results and Analysis

In this section the results of the tests will be presented. The test results include stemmer algorithms performance evaluaiton, clustering performance evaluation, and correlation test result.

4.1. Performance Evaluation Result of Stemmer Algorithms

Table 1 presents the results of stemmer algorithms performance evaluation using Paice method. The index values used were overstemming, Understemming, and stemming weight.

Table 1. Performance Evaluation of Stemmer Algorithms using Paice Methods

Number	Stemmer Algorithm	Overstemming Value	Understemming Value	Stemming Weight
1	Porter Stemmer	0.0	0.226615	0,0
2	Nazief Stemmer	0.001124	0.104166	0.010796
3	Arifin Stemmer	0.001103	0.110969	0.009948
4	Fadillah Stemmer	0.001110	0.132228	0.008400
5	Asian Stemmer	0.001152	0.087585	0.013155
6	ECS Stemmer	0.001166	0.068452	0.017034
7	Arifiyanti Stemmer	0.001166	0.086309	0.013509
8	Proposed Stemmer	0.001172	0.083333	0.014075

Porter confix stripping stemmer algorithm gets the highest understemming value of 0.225. This means that the ability to merge a group of words into one root was not good. For example: "*akhirnya* (in the end), *pengakhiran* (the end), *diakhiri* (to end)" should be transformed to "*akhir* (end)”. This was caused by many porter confix stripping algorithm's rules that do not match the affixes that existed in Indonesian language. Conversely, since there were no words that have different root words merged into a single root, the overstemming value becomes 0.

The second algorithm tested was nazief stemmer. This algorithm was developed based on the porter algorithm, whose rules were redesigned for Indonesian language. The nazief algorithm has better value than the porter algorithm. It can be caused by the use of a dictionary that serves to validate word roots, as a result of stemming process. Nevertheless, stemming errors were found in the word *berarti* (means) which should transformed in to *arti* (mean). This was caused by the sequence of affixes removal which incorrectly transformed the word *berart* into *rart*, which was not the correct root.

The third algorithm discussed was arifin stemmer algorithm. It was also using dictionary like the nazief stemmer. This algorithm had understemming value lower than porter algorithm but higher than nazief algorithm. This was due to some errors that happened in stemming process. For example, the word *keberangkatan* (departure), which transformed in to the word *angkat* (lift), while the correct root was *berangkat* (depart). This was caused by the word *berangkat* (depart) and *angkat* (lift) which were both existed in the dictionary. When the inputed word was *berangkat* (depart), the algorithm would firstly check the dictionary and found it. But, when the word *keberangkatan* (departure) was inputed, the algorithm incorrectly removed the sufixes and transformed the word into *angkat* (lift).

Fadillah stemmer algorithm was also developed from porter confix stripping algorithm. It was the only algorithm that did not use dictionary. Instead, it used syllables whose function was the same as the measure in the porter algorithm. The fadillah algorithm mistake occured in transforming the word *mengakui* (admitted) into *kaku* (stiff), while the correct root was *akui* (admit). When the word *mengakui* was inputed, the algorithm would directly run three processes. They were removing particles, possesive pronoun, and delete first order prefix. In the case of removing possessive pronoun, the word *mengakui* (admitted) was unchanged, but at the stage of removing first order prefix, the word *mengakui* was transformed into *kakui*. Here, the suffix removal stage was done. Then the suffix removal stage was executed and transformed the word *kakui* into *kaku* (stiff).

Next was the asian stemmer algorithm. It was the refinement of the nazief stemmer algorithm. In asian algorithm there was a process of checking rule precedence, which will change the order of affixes removal steps. When observed on the test results, then it was concluded that the type of error existed in the asian algorithm and nazief algorithm were almost the same. However, there was an error that occurred in the nazief algorithm but it was solved in the asian algorithm. This was caused by the rule precedence which removed prefix first instead of suffix. The seventh algorithm was ECS stemmer algorithm. It was an improvement of asian algorithm. In this algorithm, some rules were added for removing the prefix. It also added the process of returning suffix similiar to arifin algorithm. This algorithm was able to overcome some of the errors that occurred in both algorithm of its predecessor, for example, the word *berarti* (means), which could not be stemmed by the asian algorithm.

The next agorithm was arifiyanti algorithm. It was the development of asian stemmer algorithm that was deliberately adjusted to perform stemming especially on twitter data. For example, from testing result, the word *diperbaikin* which was non standard word, was transformed succesfully into *baik*. Our proposed algorithm showed an improved performance over arifiyanti algorithm. For example, the word *seharipun* which could not be stemmed into its root by asian and arifiynti algorithm, it was transformed successfully into *hari* (day). This was caused by the new rule *se-pun* was added in rule precedence checking step.

4.2. Clustering Performance Evaluation Result

This section discusses the results of clustering performance measured using DBI. It also presents statistical *t*-test results among stemmer algorithms. The following Table 2 shows the average DBI values obtained by each stemmer algorithm. Meanwhile, Table 3 shows the results of *t*-tests. Significance was indicated by the symbol "+", meaning that the algorithm on the line was better than the algorithm in the column. While the symbol "-" states otherwise, that the algorithm on the line has a significantly worse performance than the algorithm in the column. If there was no significance, then it used the symbol "=".

Table 2. Clustering Performance Evaluation Result

Algorithm	No Stemmer	Arifiyanti	ECS	Asian	Fadillah	Arifin	Nazief	Porter	Proposed Stemmer
Average DBI	9.041	8.673	8.805	8.696	8.926	8.845	8.759	9.214	8.618

Table 3. Significance Test Result

	No Stemmer	Arifiyanti	ECS	Asian	Fadillah	Arifin	Nazief	Porter
No Stemmer		-	=	-	=	=	-	=
Arifiyanti			=	=	=	=	=	+
ECS				=	=	=	=	+
Asian					=	=	=	+
Fadillah						=	=	+
Arifin							=	=
Nazief								+
Proposed Stemmer	+	=	=	=	+	=	=	+

From the results of tests, it can be inferred that the best algorithm that improved clustering performance was the proposed algorithm. It was shown by its lowest DBI value of 8.618, which mean the produced clusters have the best structure. On the other hand the other algorithms did not show significant improvement. This was because the added affixes removal rules were in accordance with semi-structured twitter data. In addition, the addition of a non-standard dictionary also helps the proposed algorithm in stemming non-standard words.

The proposed modified algorithm could improve more in clustering performance compared with arifiyanti algorithm, but the increase was not significant. The difference between the two algorithms which was caused by the existence of words that meet the new rules on the modified algorithm was not sufficient enough.

From the results and analyses that have been discussed, it can be concluded that stemmer in general will provide improved performance of data clustering. However, this does not occur in porter algorithm. Besides the stemmer errors that occur in the porter algorithm, it was also due to the rules which did not match the Indonesian language.

4.3. Correlation Test Result between Stemmer Performance and Clustering Performance

This section discusses the results of correlation testing between the stemmer algorithm performances and clustering performance using Pearson method. Table 4 shows the results of Pearson correlation testing.

Table 4. Correlation Test Result between Stemmer Performance and Clustering Performance

Correlation	df	Correlation value	p-value	Conclusion
Understemming and DBI	5	-0.90091	0.00562	negative correlation
Overstemming and DBI	5	0.9354698	0.00196	positive correlation

Based on the results of correlation test in Table 5, it can be seen that if overstemming value is getting lower, then the DBI value will decrease. Conversely, if the understemming value is getting lower, then the DBI value will increase. It can be concluded that for IR case in Indonesian, the overstemming was more important to improve IR performance. The test results also shows that the results of Flores and Moreira study [6] which states that overstemming values are negatively correlated with the performance of IR does not occur for the Indonesian language.

4.4. Processing Time

This section presents the test results of processing time for each stemmer algorithm on processing the data. The following Table 5 is the length of time needed to process data by the stemmer in seconds.

Table 5. Processing Time of Stemmer Algorithms

Algorithm	Arifiyanti	ECS	Asian	Fadillah	Arifin	Nazief	Porter	Proposed Stemmer
Time(s)	2416.499	2702.450	1967.301	0.218	3624.804	1961.879	0.627	2995.695

From the results obtained, it can be inferred that the porter algorithm has a very fast processing time. This is because porter algorithm used only affixes removal rules, and did not use words root dictionary. However, the fadillah algorithm could run faster because its affix removal rules have been adapted to the Indonesian language. Porter algorithm which has many incompatible rules for the Indonesian language caused the algorithm to do more rule checking. So, its total time becomes longer than the fadillah algorithm. The longest processing time is the arifin algorithm. This is due to the affixes combining processes that were executed if all standard affixes removal rules failed. In this process, any prefixes and suffixes that have been removed from the roots will be re-attached according to the affixes combination rules.

4.5. Weighted Scoring Result

After the values of each criterion in stemmer algorithm has been normalized, the next step was calculating the scores using weighted scoring method. The results of scores calculation are presented in Table 6.

Table 6. Weighted Scoring Result

Stemmer	Arifiyanti	ECS	Asian	Fadillah	Arifin	Nazief	Porter	Proposed Stemmer
score	0.620	0.535	0.616	0.450	0.439	0.554	0.349	0.648

From the results of weighted scoring, the proposed modified algorithm get the highest score. This is due to its low DBI values and good understemming value. The fadillah algorithm has a better score than the arifin algorithm. This is due to the excellent overstemming value and the very fast processing time. While the arifin stemmer algorithm, although its performance of clustering was good, its process time was the longest compared with other algorithms. In the last place is the porter confix stripping algorithm. This algorithm has excellent overstemming value, but clustering performance and understemming value are very poor compared with other algorithms, resulting in a low score.

5. Conclusions

In the case of study of data clustering, the best stemmer algorithm to improve IR performance was the proposed algorithm with average DBI value of 8.618. It was a significant improvement compared to Fadillah and Porter, It also excelling Arifiyanti algorithm which is the base of the proposed algorithm. The proposed stemmer algorithm got the best result because it had rules that were deliberately made to deal with twitter data. The proposed algorithm also solved the problem like the word *seharipun* which could not be stemmed by asian and arifiynti algorithm. Weighted scoring result also shows that the best algorithm for Indonesian text processing purpose was the proposed algorithm with score of 0.648. It is due to its low DBI values and good understemming value.

However, in the words root dictionary used by the stemmer algorithms, there were alphabet letters such as *a, b, c,* and so on. This may cause errors in stemming process. It is not yet known how the impact that occurs on the stemmer algorithm if the letters were not used as roots. Therefore, it is recommended to do stemming by excluding the alphabet letters.

Another conclusion is that the results obtained from the previous research [6] are not fully compatible with what happened in this study. Fadillah algorithm that was a simple stemmer, did not get a good performance. In addition, the correlation between understemming, overstemming with IR also does not match the correlation that occurred in the Indonesian language case. This may be due to the Indonesian language structure which is different from the languages tested in the previous research

References

[1] Hidayatullah AF, Ratnasari, CI, Wisnugroho, S. Analysis of Stemming Influence on Indonesian Tweet Classification. *TELKOMNIKA Telecomunication, Computing, Electronics and Control.* 2016; 14 (2): 665-673.

[2] Jaafar Y, Namly D, Bouzoubaa K, Yousfi A. Enhancing Arabic Stemming Process Using Resources and Benchmarking Tools. *Journal of King Saud University–Computer and Information Sciences.* 2017; 29: 164-170.

[3] Shrestha I, Dhakal SS. 2016. *A New Stemmer for Nepali Language.* 2nd International Conference on Advances in Computing, Communication, & Automation. Bareilly. 2016

[4] Rajput BS and Khare NA. Survey of Stemming Algorithms for Information Retrieval. *IOSR Journal of Computer Engineering.* 2015; 17(3): 76-80.

[5] Brychcín T, Konopík M. HPS: High Precision Stemmer. *Information Processing and Management.* 2015; 51: 68-91.

[6] Flores, FN and Moreira, VP. Assessing the Impact of Stemming Accuracy on Information Retrieval: A Multilingual Perspective. *Information Processing and Management.* 2016; 52: 840-854.

[7] Jadhav A, Sonar R. *Analytic Hierarchy Process (AHP), Weighted Scoring Method (WSM), and Hybrid Knowledge Based System (HKBS) for Software Selection: A Comparative Study.* Second International Conference on Emerging Trends in Engineering and Technology. Nagpur. 2009.

[8] Nazief B, *Confix Stripping: Approach to Stemming Algorithm for Bahasa Indonesia*. Internal Publication. Faculty of Computer Science, University of Indonesia. Depok. 1996.

[9] Arifin AZ and Setiono AN, *Classification of Indonesian Language News Documents with Single Pass Clustering Algorithm (in Indonesia Klasifikasi Dokumen Berita Kejadian Berbahasa Indonesia dengan Algoritma Single Pass Clustering)*. Prosiding Seminar on Intelligent Technology and Its Applications (SITIA). Surabaya. 2002.

[10] Tala FZA. Study of Stemming Effects on Information Retrieval in Bahasa Indonesia. PhD Thesis. Amsterdam: Master of Logic Project. Amserdam. Institute for Logic, Language and Computation, Universiteit van Amsterdam. 2003.

[11] Asian J. Effective Techniques for Indonesian Text Retrieval. PhD Thesis. Melbourne: School of Computer Science and Information Technology, Science, Engineering, and Technology Portfolio, RMIT University Melbourne; 2007.
[12] Arifin AZ, Mahendra IPAK and Ciptaningtyas HT. *Enhanched Confix Stripping Stemmer and Ants Algorithm for Classifying News Document in Indonesian.* International Conference on Information & Communication Technology and Systems. Surabaya. 2009.
[13] Arifiyanti AA. Feature Extraction on Indonesian Language Twitter Social Network Content in Improving Sentiment Classification Performance. Master Thesis (in Indonesia Ekstraksi Fitur pada Konten Jejaring Sosial Twitter Berbahasa Indonesia dalam Peningkatan Kinerja Klasifikasi Sentimen). Master Thesis. Surabaya: Institut Teknologi Sepuluh Nopember. 2015.
[14] Porter MF. An Algorithm for Suffixes Stripping. Cambridge: Computer Laboratory, Corn Exchange Street, University of Cambridge. 1980.
[15] Rendón E, Abundez I, Arizmendi A, and Quiroz EM. Internal versus External Cluster Validation Indexes. *International Journal of Computers and Communication.* 2011; 5(1): 27-34.
[16] Paice, C.D. An Evaluation Method for Stemming Algorithms. Lancaster: Department of Computing, Lancaster University. 1996.
[17] Sulistiani H, Tjahyanto A. Heterogenous Feature Selection for Classification of Customer Loyalty Fast Moving Consumer Goods: Case Studey of Instant Noodle. *Journal of Theorethical and Applied Information Technology.* 2016; 94: 77-83
[18] Naraadhipa AR and Purwarianti. A. *Sentiment Classification for Indonesian Message in Social Media.* International Conference on Electrical Engineering and Informatics, Bandung. 2011.
[19] Li C, Sun L, Jia J, Cai Y, Wang X. Risk Assessment of Water Polluiton Based on An Integrated *K-means* Clustering and Set Pair Analysis Method In The Region of Shiyan, China. *Science of the Total Environment.* 2016; 557-558: 307-316.
[20] Pearson K. Note on Regression and Inheritance in The Case of Two Parents. 1895; 58: 240–242.
[21] Dalwadi B, Desai N. *An Affix Removal Stemmer for Gujarati Text*. International Conference on Computing for Sustainable Global Development. New Delhi. 2016.
[22] Meitei SP, Purkayastha BS, Devi HM. *Development of A Manipuri Stemmer: A Hybrid Approach.* International Symposium on Advanced Computing and Communication. Silchar. 2015.
[23] Gupta V, Joshi N, Mathur I. *Rule Based Stemmer in Urdu.* 4th International Conference on Computer and Communication Technology. Allahabad. 2013.
[24] Mahmud MR, Afrin M, Razzaque MA, Miller E, Iwashige J*A. Rule Based Bengali Stemmer.* International Conference on Advances in Computing, Communications and Informatics. New Delhi. 2014.
[25] Alam MJ, Oullet P, Kenny P, O'Saughnessy D. Comparative Evaluation of Feature Normalization Techniques for Speaker Verification. *Advances in Nonlinear Speech Processing.* 2011; 246-253.