ISSN 2007-9737

A Fuzzy System for Identifying Partial Reduplication

Apurbalal Senapati

Central Institute of Technology,
Department of CSE,
India

a.senapati@cit.ac.in

Abstract. Reduplication is a common feature more or less in almost all languages. It is a linguistic process that has been studied since back in both a morphological as well as a phonological process. Literary reduplication is the repetition of tokens in various forms like morpheme, phrase, word, etc. In most cases, it affects the syntactic or/and semantic meaning of the original words. But in several cases, it is not the exact reduplication rather partial reduplication. In the exact or total reduplication, it exactly reiterates a word or a phrase (e.g. fifty-fifty, bye-bye in English) while in partial reduplication reiteration partially (e.g. flip-flop in English). The morphological structure of a total or partial reduplication is relatively simple for English-like languages, but it is complex in the case of African, Austronesian, or South-Asian languages. Earlier researchers tried to address this problem using the various heuristic approaches. This paper presented a set of fuzzy-based approach to deal with partial reduplication in the Bengali language.

Keywords. Fuzzy, rule-base, reduplication, language, morphology, natural language processing.

1 Introduction

Identifying the reduplications from the text corpora is a specific string matching or a pattern matching problem based on the morphological [1] as well as a phonological [2] criterion. Whereas string matching or pattern matching is a fundamental operation in computer science. It plays an essential role in various real-world applications, including Natural Language Processing (NLP). Various string matching algorithms have been developed like Rabin-Karp’s algorithm [4], Knuth-Morris-Pratt's (KMP) algorithm [5], automata-based algorithm [6]. The fuzzy-based approximate string matching algorithms are also used in various applications like spellcheckers, search engines, Bioinformatics, etc. But none of these are suitable for partial reduplication detection in the run of text [3] in the context of NLP applications. Reduplication detection in a text is a common problem in NLP. None of the conventional string matching algorithms are suitable for partial reduplication detection. Researchers tried to address such a problem using the ruled-based heuristic approach, but the rules are very much language-dependent and have low coverage. This situation motivated us to develop a fuzzy rule-based system to capture the partial reduplication from the corpus.

2 Types of Reduplications

Based on the morphological construction the reduplications are of two types [7], total and partial reduplication. The total reduplication is the repetition of the whole word such as “bye-bye”, “quack-quack” etc. in English. On the other hand, partial reduplication is defined as the repetition of part of the word such as "flip-flop", “criss-cross” in English.

The reduplication is relatively less frequent in English-like languages and their morphological construction is also simple.

But the languages of African, South-Asian, etc. are complex and difficult to identify from the text corpus. There is complex morphological construction on partial reduplication.

It may be with suffixes, prefixes, sub-string, etc. Sometimes it co-occurs with the eco-word, synonyms, antonyms, etc. [8]. From the computational point of view, handling the total reduplication is relatively simple. However, the main concern is in partial reduplication, which has been addressed in this paper.

3 Importance of Reduplications

Reduplication is a morphological operation of repetitions. Sometimes it affixing reflects certain phonological characteristics of the language. It is also used in inflection to convey the grammatical dimension like the number and sometimes derived a new word. Furthermore, reduplication is often used when a speaker adopts a tone that is more expressive or figurative speech and it is also often but not exclusively iconic in meaning. Sometimes reduplication is used in echoing and found in onomatopoeia. In some cases, it is used to emphasize e.g. no-no in English. Quite a few reduplications can be found in child language, such as bobo, dada, etc.

What exactly the syntax or semantic changes depends on a particular language [9]. It has been identified that there are forty-five functions in 108 languages [10]. For example, it is used to express augmentation, intensification, diminution, attenuation, etc [11].

Augmentation implies the increase of quantities, activity, or greatness. Augmented through reduplication adds to the semantic value of a word in various languages. In the Indonesian language, the reduplication of *buku* (book) is *buku-buku* (plural form of a book). In Bengali কে / *ke* means **who** and its reduplicated কে - কে / *ke-ke* impels **plural form of who**.

In the intensification increase the degree e.g. in the Indonesian language the reduplication of *mue ko* (book) is *mue-mue ko* (big book). In Bengali জোরে / *jore* means **fast** and its reduplicated জোরে- জোরে / *jore-jore* impels **very fast**.

Diminution says the decrease of quantity, e.g. In Indonesian *anak* means **child** where reduplicated *anak-anak* means **adopted child**.

The above examples show that Several semantic and syntactic properties are connected with reduplication amongst the languages [12]. All such features of the reduplications are not encapsulated in the various NLP applications. In order to achieve the high precision result, it needs a special treatment of reduplication in various applications. Consider an example i.e. S1 and S2 in Bengali:

S1: ঘরে বেকার যুবক / *ghare bekar yubaka* (Unemployed youth at home).

S2: ঘরে ঘরে বেকার যুবক / *ghare ghare bekar yubaka* (Unemployed youth at every house).

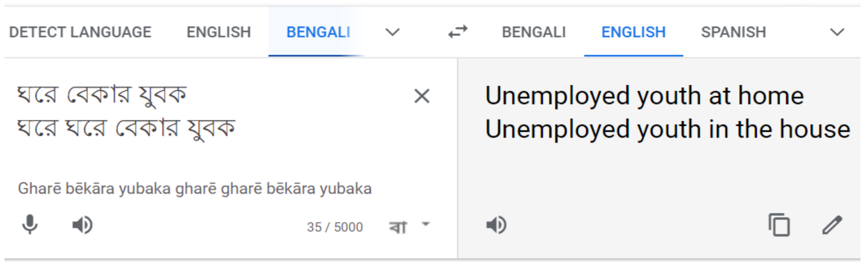
In sentence S1, there is a word ঘরে / *ghare*, which means in house, but its reduplicated forms appear in sentence S2, ঘরে ঘরে / *ghare ghare*, which means every house. It clearly shows that how reduplication changes the semantics. Figure 1 shows that the Bengali-English google machine translation system fails to capture the reduplicated meaning. It is just an instance but it happens in most cases. These examples show the importance of reduplication in language processing applications.

4 Reduplications in Bengali

The Bengali language belongs to the Indo-Aryan language spoken most widely spoken language of Bangladesh and the second most widely spoken of the 22 scheduled languages of India [13]. It is one of the state languages of the Indian states of Tripura, West Bengal as well as some parts of Assam it is the second most spoken language in the Indian subcontinent. The Bengali-speaking population is more than 250 million native and more than 300 million speakers in total, making Bangla the sixth language [14] after English, Chinese, Hindi, Spanish, Arabic.

4.1 Existing Work and Frequency of Reduplication in Bengali

The literature shows that most of the existing works on reduplication are found in the domain of linguistic. It can be found in various Indic languages like Bengali [3], Sanskrit, Tamil [15], Kannada [16], Manipuri, or other South Asian languages [17]. Some of the researchers tried to address the issue in computational aspects [18]. But it is hardly found that tried to explore its quantization or addressed all forms and aspects.



ঘরে বেকার যুবক
ঘরে ঘরে বেকার যুবক

Gharē bēkāra yubaka gharē gharē bēkāra yubaka

35 / 5000

Unemployed youth at home
Unemployed youth in the house

Fig. 1. The Bengali-English google translation could not capture the reduplication semantics (access: 20-07-2021)

Senapati et. al. [3] tried to develop a heuristic-based generic approach, but that does not cover all types of partial reduplications.

Reduplication is a common feature in Bengali and is frequently used in spoken and written text.

A corpus-based study shows that [3] there is 0.71% of the word in a corpus are reduplicated and 9.4% of the sentences contain reduplicated words in the TDIL corpus [19]. These statistics show the importance of the reduplicated feature and it cannot be ignored in any NLP applications. The rich morphology of the language makes it more productive and complex.

4.2 Peculiarities of Bengali Reduplication

Based on the morphological construction there are different types of reduplication like total or partial as mentioned earlier. But in the case of Bengali, the partial reduplication is more complex and lots of peculiarities are there. The morphology of reduplication is of the form X X (or X-X or XX) i.e. repetition of a word or token. In the case of partial it is of the form X Y (or X-Y or XY), where X is part of Y or Y is a part of X. Sometimes X is the prefixing or suffixing Y. Sometimes X, Y have the opposite meaning of each other but X Y together forms a reduplication. Some of such peculiarities are explained below with examples.

When X is a prefix of Y: in another word it can be expressed as Y is the inflection form of X. Example, ধব ধবে / *dhab dhabe* (pure white color). টক টকে / tak take *(pure red color).* Here each individual word is not a valid term, but the reduplicated expressions give a significant meaning.

When X is a suffix of Y: in this case, X appears as a suffix of Y. Example, ন্যায় অন্যায় / *nayai onayai* (justice or injustice). ধাতব অধাতব / *dhataba adhataba* (metallic or non-metallic). জ্ঞানে অজ্ঞানে / *gayene agayene* (consciously or unconsciously). Here the meaning each pair of words are opposite to each other and together they give a separate meaning. In such cases, the reduplicated gives the meaning completeness i.e. meaning gives a whole concept.

When X Synonym Y: when two words X, Y are lexically different but their meaning is same or almost similar together form a reduplication. For example, চুরি চামারি / *churi chamari* (robbery) where the meaning of চুরি / *churi* is theft, চামারি / *chamari* means illegal work. চাল চুলো / *chal chulo* (economically poor) where the meaning of চাল / *chal* is rice and চুলো / *chulo* means cooking burner.

When X Antonym Y: when two words X, Y are lexically different, but their meaning is opposite to each other that forms a reduplication. For example, আগু পিছু / *aagu pichu* (back and forth), কাছে দূরে / *kachhe dure* (near and far), দেনা পাওনা / *dena pawna* (loan and owing), etc.

When Y is the eco word in X: An echo word is a word or phrase that does not have any meaning and does not exist separately. It co-occurs with another word to form a reduplication. Generally, it imitates the same sound as the associate word and has the same morphological construction. The eco word changes its meaning as the like duplications, etc. Example, খাবার দাবার / *khabar dabar* (varieties food etc.) where the meaning of খাবার / *khabar* is food and দাবার /

dabar is eco-word. Similarly, জল টল / *jal tal* (water, beverage etc.) where the meaning of জল / *jal* is water and টল / *tal* is an eco-word. In this case, the first token has a specific meaning, and when it scans with the eco-word it changes its meaning.

The above examples show that just using the string matching algorithm it is difficult to identify such reduplication from the corpus. Some researchers tried to identify such pairs using the heuristic approaches. But such an approach does not give high precision accuracy. Not even fuzzy-based string matching algorithms retrieve all such partial reduplications but it gives many false positive instances.

For example, the in-build Python-based fuzzy string matching library Fuzzywuzzy returns the matching score of 67 in both the cases ('এহাত / *ehat*' (this hand), 'ওহাত / *ohat'* (that hand)) and ('এহাত / *ehat'* (this hand), 'কহাত / *khat'* (how many hands)). But in first case is a valid partial reduplication but not in the second case. So, this matching score cannot be used to identify partial reduplications.

In our approach, we have considered the other morpho-linguistic features in the matching criterion like the length of characters, the length of characters in the vowel modifiers, ordered sequence of the characters, ordered sequence of the vowel modifiers, position of mismatches, rhythmic similarity, eco words, antonym words. Each feature is described in brief with examples.

Length of characters: length of characters is an important feature in reduplication. In the case of English reduplication (XY) in almost all cases the length (X) = length (Y). But in case of Bengali reduplication it follows either length (X) = length (Y) or length (X) = length (Y) + 1. For example, গুছিয়ে গাছিয়ে / *guchhiye gachhiye* (well-organized way) where it is length (X) = length (Y). And কারণে অকারণে / *karone okarone* (all the time) where it is length (X) = length (Y) + 1.

Length of vowel modifier: in Bengali, 10 vowels appear as signs act as a modifier is known as vowel modifier. These are {া / আ, ি / ই, ী / ঈ, ে / এ, ৈ / ঐ, ো / ও, ৌ / ঔ, ু / উ, ূ / ঊ, ৃ / ঋ}. These vowel modifiers play an important role in the morphology and construction of the reduplications. Our observation is that the length of the vowel modifier is an important feature in identifying the partial reduplicated word.

It is observed that in most of the cases the mismatched character is either in an alphabet or in a vowel modifier but not in both cases.

Example, in মালিশ টালিশ / malis talis (message etc.) the alphabet set are < ম ল শ > and < ট ল শ> and their vowel modifier sets are < া ি > and < া ি >. In this case, there is a mismatch in an alphabet. সামনা সামনি / samna samni (nearby), here the alphabet set < স ম ন > and < স ম ন > but their vowel modifier sets are < া া> and < া ি >. In this case, the mismatch is in the vowel modifier.

Position of mismatches: in partial reduplication, the repetitive word has a partial match or in other words, we can say there is a mismatch. The position of mismatch is an important feather in partial reduplication.

For example, গুছিয়ে গাছিয়ে / *guchhiye gachhiye* (well organized), the mismatch is at the very beginning of the characters. খরচ খরচা / *khorach khoracha* (expanses), the mismatch is at the end position. It is noted that in most of the cases the mismatch is at either beginning or at the end position.

5 Fuzzy Inference System Architecture

A Fuzzy Inference System (FIS) is a technique that is used for mapping from an input space to an output space using fuzzy logic.

A FIS tries to formalize the reasoning of a set of rules of human language employing fuzzy logic [20-21]. The architecture of the FIS includes four components, these are Fuzzification, Knowledge base, Inference engine, and Defuzzification.

The architecture of the FIS is illustrated in Figure 2. Each component is illustrated below. The fuzzification [22-23], choosing of membership function [24-25], fuzzy knowledge base [26-27], aggregation rules [28], defuzzification [29] are the important components of the fuzzy inference system.

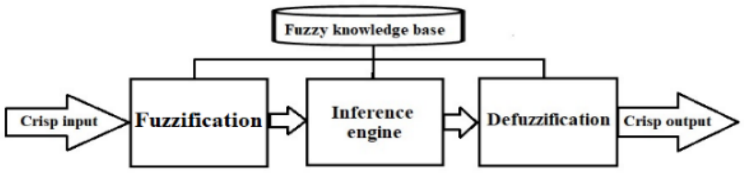


Fig. 2. Fuzzy inference system architecture

6. Rule-Base for Identifying for Partial Reduplication

The rule-base of identifying the partial reduplication of Bengali is comprised of the features that are described in section 4. We described them in two sets, the crisp rule base and fuzzy rule base are described next.

6.1 Crisp Rule Base

This set consists of three rules. Rules are defined as a crisp rule because rules are hard decisions like yes or no. That means, if it satisfied a predefined condition then it is considered as a reduplicated form otherwise not.

Rule 1:

IF *X is synonym of Y* THEN *(X Y) is reduplicated*

Rule 2:

IF *X is antonym of Y* THEN *(X Y) is reduplicated*

Rule 3:

IF *Y is a eco word* THEN *(X Y) is reduplicated*

In the implementation point of view, to find the synonym and antonym a Bengali WordNet has been used that build with a similar scheme of BanglaNet [30]. On the other hand, to check the *eco-word* find the entry of the word in the Bengali WordNet. If the entry is not found, then it is considered an *eco-word*. Because the eco word is invalid or it does not have any meaning.

6.2 Fuzzy Rule Base

This set also consists of nine fuzzy rules. These rules are dealing with morpho-linguistic features like the number of mismatch characters (i.e. length comparison of characters), the number of mismatches of vowel modifiers (i.e. length comparison of vowel modifiers), the position of mismatch.

Before we formulate the actual fuzzy rules, define the fuzzy linguistic variables of similarity score and some terminologies used in the rules:

Score = {high, medium, low}

charMismatch(X, Y) returns the number of mismatch characters of the strings X and Y.

vmMismatch(X, Y) returns the number of mismatches of vowel modifiers in the strings X and Y.

Rule 1:

IF *charMismatch(X, Y) = 0* THEN *Score = high*

Rule 2:

IF *charMismatch(X, Y) = 1* THEN *Score = medium*

Rule 3:

IF *charMismatch(X, Y)* ≥ 2 THEN *Score = low*

Rule 4:

IF *vmMismatch(X, Y) = 0* THEN *Score = high*

Rule 5:

IF *vnMismatch(X, Y) = 1* THEN *Score = medium*

Rule 6:

IF *vmMismatch(X, Y)* ≥ 2 THEN *Score = low*

Rule 7:

IF *misPosition (X, Y) = first* THEN *Score = high*

Rule 8:

IF *misPosition (X, Y) = mid* THEN *Score = low*

Table 1. Rules are in tabular form

	IF		THEN
Rule	**Mismatch character / vowel modifier**	**Mismatch character in position**	**Score**
1	0	NA	high
2	1	NA	medium
3	≥ 2	NA	low
4	0	NA	high
5	1	NA	medium
6	≥ 2	NA	low
7	NA	first	high
8	NA	mid	low
9	NA	last	high

Rule 9:

IF *misPosition (X, Y) = last* THEN *Score = high*

A summarization of the rules is given in Table 1.

7 System Architecture

Figure 3 shows the system architecture of our sys-tem of identifying the partial reduplication from the corpus.

As described above, our rule base is consisting of two sets of rules, the crisp rule-base, and the fuzzy rule-base. The crisp rule base consists of three crisp rules which are treated as a separate component in our architecture (Fig. 3).

The fuzzy rule-base consists of the nine fuzzy rules and a fuzzy inference system is used for this rule-base (Fig. 3). So, our system is a hybrid architecture that consists of a crisp rule-base along with a fuzzy inference system (Fig. 3). Its execution process is in parallel mode, i.e. for a pair of a token (X Y), it is considered input for both, the fuzzy inference system as well as a crisp rule base. And it is considered a reduplicated pair if any one of the rule-base outputs as a reduplicated pair.

7.1 Fuzzification in our System

The fuzzification is the method of transforming a crisp value into a fuzzy value. Here it is described the exact implementation in our system.

This fuzzification is also defined in two ways. For the fuzzy Rule 1 – Rule 6, the linguistic variables are considered as low, medium, and high.

But for the fuzzy Rule 7 – Rule 9 and hence the fuzzification is done for Rule 1 – Rule 6, in one way and for Rule 7 – Rule 9 in different ways.

The fuzzification function for Rule 1 – Rule 6, is like the trapezoidal function is shown in Fig. 4.

The value along the x-axis is considered as the similarity measured in percentage. The 0-50% matching is considered as low, 40-80% matching is considered as medium and 50-100% matching is considered as high.

These values are set from the concept domain study of various literature. Now for a pair (X Y) if their matching percentage is x_1, and the membership value corresponding to the high and medium is $\mu_1(x_1)$ and $\mu_2(x_1)$ then its resultant value is considered as $\max\{\mu_1(x_1), \mu_2(x_1)\}$ (shown in Fig. 8). A similar concept is used for Rule 7 – Rule 9 is shown in Fig. 5.

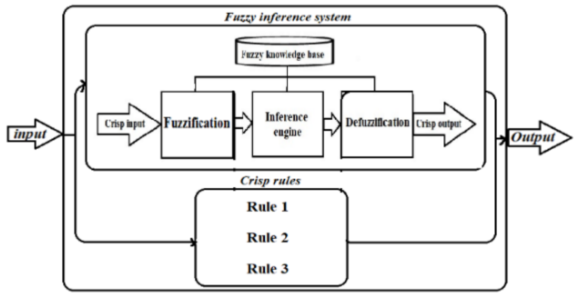


Fig. 3. System architecture of our system

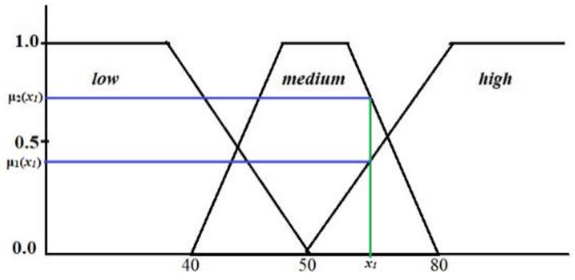


Fig. 4. Fuzzification mechanism (for Rule 1 – Rule 6)

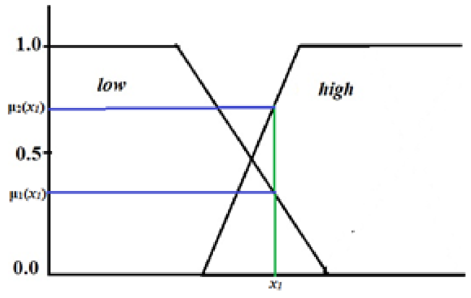


Fig. 5. Fuzzification mechanism (for Rule 7 – Rule 9)

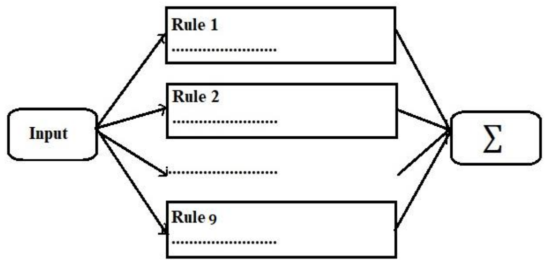


Fig. 6. Aggregation in fuzzy rule-base

7.2 Aggregation and Defuzzification in our System

In the fuzzy rule base, all rules are executed parallelly. In order to make a final decision, the fuzzy sets that represent the outputs of all rules are needed to aggregate into a single output that is also a fuzzy set (Fig. 6). There are various methods are for aggregation, but we have considered the max-min method in our system [31].

7.3 Defuzzification

Now the final step is defuzzification. Here it needs to convert the output of aggregated fuzzy member-ship value (Fig. 6) to the crisp value. This is done by the max-membership method i.e. it considered maximum x value corresponding to that fuzzy membership value. The x-value represents the similarity measured (described in section 7.1) of two strings (X Y) in percentage. In our cases, a threshold value is defined and the crisp output exceeds that threshold value it is considered as reduplication and in our experiment, it is set as 75.

8 Experiment and Result

8.1 Test Data Set

The system is tested on the Technology Development for Indian Languages (TDIL) Bengali corpus [19]. The corpus is created under the Department of Electronics, Govt. of India for various Indian languages including Bengali. This corpus content is with texts from diverse areas like Fine Arts (5%), Literature (20%), Social Sciences (15%), Commerce (10%), Natural Sciences (15%), Mass media (30%), etc. The corpus content 3,34,260 sentences and 44,29,574 words.

8.2 Results

Though the system is mainly focused only on partial reduplication, it is capable capture the total

Table 2. System accuracy

TDIL corpus	Correctly Identified (True positive)	Wrongly Identified (False positive)	Accuracy
Total Re-duplication	2,789	40	0.98
Partial Re-duplication	1,058	169	0.86
Total + Partial Re-duplication	3,847	209	0.95

reduplication too. Hence the result is described in three following ways shown in Table 2.

9 Error Analysis

The error analysis is not done in fuzzy implementation and fuzzy rule-bases. It is primarily tried to focus only to investigate source of false-positive errors. In cases of total reduplication, the output is 100% accurate with respect to the lexical level but there is an error in 40 cases in the semantic level.

For example, the system identified “2-2” as a total reduplication, which is lexically correct but semantically it is not correct. So all the 40 false-positive cases the source of errors are similar. Some of other instances are দিদি / didi (elder sister), শিশি / *sisi* (glass bottle), কাকা / kaka (uncle), etc.

In the case of partial reduplication, there are 169 false-positive cases. There are some words (word pair) used in Bengali, their morphological structure looks like a reduplication but it is not and these are the sources of this error. For example, কোকা কোলা / *coca cola*, মায়ের গায়ের / *mayer gayer*, etc.

10 Conclusion

The primary focus of this paper is a successful implementation of a FUZZY-NLP system for partial reduplication identification from the text corpus. Earlier researchers tried to address this issue by heuristic approaches on the basis of string matching. In this context, it is one of the pioneering attempts that concretize a fuzzy-based or fuzzy string matching system. This approach incorporated some morphological features like an ordered sequence of characters’/vowel modifiers, lengths, the position of mismatch, etc.

Some of these features are not considered earlier. The system is easily imported to identify the reduplications for other morphological-rich languages. Similar concepts can be used for other similar problems like morphological analysis, spell checker, etc.

The system can be fitted in any language just replacing the rule base module. This system also introduced three crisp rules, those rules address the exceptional reduplications in the Bengali languages. The yield of these rules is high precessions on the availability of rich language resources like a digital dictionary, WordNet, etc.

References

1. **Abbi, A. (1990)**. Reduplication in Tibeto burman languages of south Asia. Japanese Journal of Southeast Asian Studies, Vol. 28, No. 2, pp. 171–181. DOI: 10.20495/tak. 28.2_171.
2. **Wilbur, R. B. (1973).** The phonology of reduplication. Bloomington: Indiana University Linguistics Club.
3. **Senapati, A., Garain, U. (2015).** A computational approach for corpus-based analysis of reduplicated words in Bengali. International Conference on Intelligent Text Processing and Computational Linguistics (CICLING), pp. 456–466. DOI: 10.1007/978-3-319-18111-0_34.
4. **Karp, R.M., Rabin, M.O. (1987).** Efficient randomized pattern-matching algorithms. IBM Journal of Research and Development, Vol. 31, No. 2, pp. 249–260. DOI: 10.1147/ rd.312.0249.

5. **Knuth, D., Morris, J., Pratt, V. (1977)**. Fast pattern matching in strings. SIAM Journal on Computing, Vol. 6, No. 2, pp. 323–350. DOI: 10.1137/0206024.
6. **Melichar, B. (1995).** Approximate string matching by finite automata. International Conference on Computer Analysis of Images and Patterns (CAIP), Lecture Notes in Computer Science, Vol. 970, pp. 342–349. DOI: 10.1007/3-540-60268-2_315.
7. **Inkelas, S., Zoll, C. (2005).** Reduplication: doubling in morphology. Cambridge Studies in Linguistics, Cambridge University Press, Vol. 106.
8. **Thompson, H.R. (2010).** Bengali: A comprehensive grammar. Routledge, pp. 663–672.
9. **Fischer, O. (2011).** Cognitive iconic grounding of reduplication in language. Semblance and Signification, Vol. 55, pp. 55–82. DOI: 10.1075/ill.10.04fis.
10. **Li, Y., Ponsford, D. (2018)**. Predicative reduplication: Functions, their relationships and iconicities. Linguistic Typology, Vol. 22, No. 1, pp. 51–117. DOI: 10.1515/lingty-2018-0003.
11. **Kajitani, M. (2005).** Semantic properties of reduplication among the world's languages. LSO Working Papers in Linguistics, Proceedings of WIGL´05, Vol. 5, pp. 93–106.
12. **Inkelas, S. (2014).** Non-concatenative derivation: Reduplication. **Lieber R., Štekauer, P. (eds.)** The Oxford Handbook of Derivational Morphology. DOI: 10.1093/ oxfordhb/9780199641642.013.0011.
13. **Office of the Registrar General & Census Commissioner (2021).** Census of India 2011. Ministry of Home Affairs, Government of India. https://censusindia.gov.in/2011Census /C-16_25062018_NEW.pdf.
14. **Szmigiera, M. (2021).** The most spoken languages worldwide.
15. **Ananthanarayana, H.S. (1976).** Reduplication. Sanketi Tamil OpiL, Vol. 2, pp. 39–49.
16. **Murthy, C. (1972).** Formation of Echo-Words in Kannada. Proceedings of the Second All India Conference of Dravidian Linguistics.
17. **Abbi, A. (1992).** Reduplication in south Asian languages: An aerial. Typological and Historical Study. South Asia Books.
18. **Chakraborty, T., Bandyopadhyay, S. (2010).** Identification of reduplication in Bengali corpus and their semantic analysis: A rule-based approach. Proceedings of the Multiword Expressions: from Theory to Applications (MWE), pp. 73–76.
19. **Jha, G.N. (2010).** The TDIL program and the Indian language corpora initiative (ILCI). Proceedings of the Seventh Conference on International Language Resources and Evaluation (LREC), pp. 17–23.
20. **Zarandi, M.H.F., Mohammadhasan, N., Bastani, S. (2012).** A fuzzy rule-based expert system for evaluating intellectual capital. Advances in Fuzzy Systems, Vol. 2012. DOI: 10.1155/2012/823052.
21. **Mendel, J.M. (1995).** Fuzzy logic systems for engineering: A tutorial. Proceedings of the IEEE, Vol. 83, No. 3, pp. 345–77. DOI: 10.1109/5.364485.
22. **Manikopoulos, C.N., Zhou, M.C., Nerurkar, S.S. (1995).** Design and implementation of fuzzy logic controllers for a heat exchanger in a water-for-injection systems. Journal of Intelligent and Fuzzy Systems, Vol. 3, No. 1, pp. 43–57. DOI: 10.3233/IFS-1995-3105.
23. **Hellendoorn, H., Thomax, C. (1993).** Defuzzification in fuzzy controllers. Journal of Intelligent & Fuzzy Systems, Vol. 1, No. 2, pp. 109–123. DOI: 10.3233/IFS-1993-1202.
24. **Xiao, Z., Xia, S., Gong, K., Li, D. (2012).** The trapezoidal fuzzy soft set and its application in MCDM. Applied Mathematical Modelling, Vol. 36, No. 12, pp. 5844–5855. DOI: 10.1016/ j.apm.2012.01.036.
25. **Jorba, L., Adillon, R. (2017).** A Generalization of trapezoidal fuzzy numbers based on modal interval theory. Symmetry, Vol. 9, No. 10. DOI: 10.3390/sym9100198.
26. **Zimmermann, H.J. (1999).** Practical applications of fuzzy technologies. The Handbooks of Fuzzy Sets Series, Kluwer Academic Publishers. DOI: 10.1007/978-1-4615-4601-6.
27. **Wu, K., Zhou, M., Lu, X.S., Huang, L. (2017).** A fuzzy logic-based text classification method for social media data. IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 1942–1947. DOI: 10.1109/SMC.2017.8122902.
28. **Tomlin, L., Anderson, D.T., Wagner, C., Havens, T.C., Keller, J. M. (2016).** Fuzzy integral for rule aggregation in fuzzy inference systems. Information Processing and Management of Uncertainty in Knowledge-Based Systems. IPMU´16. Communications in Computer and Information Science, Vol. 610, pp. 78–90. DOI: 10.1007/978-3-319-40596-4_8.
29. **Talon, A., Curt, C. (2017).** Selection of appropriate defuzzification methods: Application to the assessment of dam performance. Expert Systems with Applications, Vol. 70, pp. 160-174. DOI: 10.1016/j.eswa.2016.09.004.
30. **Rahit, K.M., Hasan, K.T., Amin, M., Ahmed, Z. (2018).** BanglaNet: Towards a WordNet for bengali language. Proceedings of the 9th Global WordNet Conference, pp. 1–9.
31. **Bobillo, F., Straccia, U. (2013).** Aggregation operators for fuzzy ontologies. Applied Soft

Computing, Vol. 13, No. 9, pp. 3816–3830. DOI: 10.1016/j.asoc.2013.05.008.

32. **Naaz, S., Alam, A., Biswas, R. (2011).** Effect of different defuzzification methods in a fuzzy based load balancing application. International Journal of Computer Science Issues (IJCSI), Vol. 8, No. 5, pp. 261–267.

Article received on 29/07/2021; accepted on 30/09/2021.
Corresponding author is Apurbalal Senapati.