International Journal on Emerging Technologies **11**(2): 817-824(2020)

ISSN No. (Print): 0975-8364
ISSN No. (Online): 2249-3255

LemmaChase: A Lemmatizer

***Rupam Gupta*[1] *and Anjali G. Jivani*[2]**
[1]*Associate Professor, Department of MCA, SVIT, Vasad (Gujarat), India.*
[2]*Associate Professor, Department of Computer Technology, MSU, Baroda (Gujarat), India.*

(Corresponding author: Rupam Gupta)
(Received 03 January 2020, Revised 14 February 2020, Accepted 17 February 2020)
(Published by Research Trend, Website: www.researchtrend.net)

ABSTRACT: Text Mining is a discovery or technique through which interesting information and hidden knowledge is automatically extracted from un-structured or semi-structured text. The critical part of understanding the textual data and giving an appropriate output as per the user requirement needs an initial important task of text pre-processing. Amongst the different pre-processing steps is an important one called Stemming. An improvisation to Stemming is Lemmatizing. This paper proposes a Lemmatization model which attempts to eliminate the shortcomings of the currently available popular Lemmatizers like the Stanford LemmaProcessor, Spacy Lemmatizer, LemmaGen, MorphAdorner etc. This model takes into account the nominalised/derived words for which correct lemmas are currently not generated by any available Lemmatizer. To develop a lemmatizer, the foremost challenge lies in understanding the morphological structure of any input English word and especially comprehending the derivational word's structure. Another important challenging task for a lemmatizer is identification of derivational suffix from morphed words and then extraction of dictionary base word from that derived word. Existing famous and popular lemmatizers are not handling such derivative words to extract their base words. The proposed lemmatizer – LemmaChase, extracts the base word correctly considering the pre-requisite knowledge of the word's Part of Speech, different class of suffix rules and efficiently implementing the recoding rules using the WordNet Dictionary. LemmaChase successfully generates the base word form from all its derivational / nominalized word forms available in any standard English dictionary.

Keywords: Information Extraction, lemmatization, nominalization, stemming, Text Mining.

Abbreviations: IE, IR, POS.

I. INTRODUCTION

Stemming algorithms are applied as a prerequisite task of any Text Mining application or natural language text analysis for extracting root words or stemmed tokens from different but semantically similar morphed words. Morphological variants of a single root word carry similar semantic meaning in a particular text. So with the help of stemming algorithm, single token is generated from different morphological variants of a word, derived from a single root word, present within a text. This will help to extract and to retrieve core information from any un-structured and semi-structured text. Modified version of stemming technique is called lemmatizing which mostly applies morphological finite state transducer method to generate dictionary root word from different morphological variants. Unlike stemming where the root word generated is not necessarily a meaningful dictionary word, lemmatizing ensures that the root word is always a meaningful dictionary word. The challenge lies in finding this meaningful dictionary word called the lemma. There are many other approaches for the lemmatization process to get dictionary root word from morphological variants of words.

Major approaches have been observed through the study conducted in the task of lemmatization which is a pre-requisite task of Information Extraction and Information Retrieval. Other than morphological analyzer; lemmatizers based on Edit Distance and the Ripple Down Rule are the other lemmatization processes which are popular in Text Mining and Natural Language Processing.

The available approaches are handling suffixes of words which are nouns in plural form and words which exist as verbs in all tenses. The limitation of those lemmatizers is that those models and tools are not able to generate the correct dictionary lemmas of the derivational words, especially of the nominalized words.

Nominalization or nominalisation is the use of a word form which actually does not exist as a noun but is being used as a noun, or as the head of a noun phrase, with or without morphological transformation. This word could actually be a verb, an adjective or an adverb. This derivational affix is added for this word to convert it to a noun form e.g. the noun "application" has been produced from the verb "apply". These words are being treated as separate independent words in the English dictionary without connecting them to their actual root words or lemmas. These word formations are also called 'nouning'. Some examples of nominalized words are: Nouns formed from adjectives are: legalization (from legal), intensity (from intense). Word "Safety" came from word "safe". Nouns formed from verbs are: failure (from fail), nominalization (from nominalize). The Nominalized word show only a single Part of Speech (POS) in any English dictionary, though its root word basically exists in different POS form and root word's morphological structure also differs from nominalized word's structure (judgmental to judge/ applicable to apply/ angrily to angry).

"Computational Morphology and Natural Language Parsing are the two important as well as essential tasks required for a number of natural language processing application including machine translation" [20].

Morphological structure of any English word is talking about the form of a word including inflectional and derivation. At the basic level, words are made of "morphemes". Morphemes are the smallest units of meaning: roots and affixes (prefixes and suffixes) [6, 10]. Derivational morphology is discussing about the process of nominalization.

This paper basically discusses the work done to handle this major challenge in finding the correct lemma of nominalized words. The first part of the proposed model is to find out and eliminate the correct suffix and then to recode the ending one or two characters of what are left-over of the token to form a valid dictionary word. Finally, the newly formed word's POS is identified and then its lemma is extracted and that lemma is the most probable root word of the input-nominalized word. The work has been compared with that of Stanford lemmatizer, Lemmagen lemmatizer, Spacy lemmatizer WordNet lemmatizer and MorphAdorner. The output of these lemmatizers has been compared with the proposed LemmaChase lemmatizer's output. A very detailed analysis after empirically executing each lemmatizer has been discussed further in this paper.

All above mentioned available approaches of lemmatizers, are not focusing on nominalised or derivational input words to generate correct dictionary base words. Some of the mismanaged words are confusion, judgmental, applicant, examiner, etc. to name a few. But, in this proposed lemmatizer, single class bounded nominalised words are merged into their corresponding class's single base word. For e.g. application/applicable/applicant are merged into "apply". This proposed lemmatizer will impart major satisfactory role for generating correct expected output in the area of text mining, text simplification, text summarization (part of information extraction) and in the area of information retrieval.

II. RELATED WORK IN LEMATIZERS

Unlike stemming, where a lot of work has been done and many stemmers have been popularized, there is still scope in designing of a lemmatizer as discussed before. However as a part of the literature study and background work, it has been observed that broadly three different categorical approaches have been used in designing of lemmatizers.

The first approach, "Edit Distance" (also known as Levenshtein Distance) has been implemented on a dictionary-based algorithm in order to get the automatic induction of the normalized form (lemma) of regular and blandly irregular words without direct supervision. The algorithm is a composition of two alignment approaches based on the string similarity and presence of the most frequent inflectional suffixes [11].

The Levenshtein Distance of two strings A and B (denoted as $d_L(A,B)$) is computed on minimum number of single character insertions, deletions and substitutions required to convert string A to string B. The greater the Levenshtein distance, more mismatched strings, they are. The $d_L(A, B)$ can be easily calculated which is given below [11]:

$$d_L[i, j] = \min (d_L[i,j-1]+c[\varepsilon,B_j], d_L[i-1,j]+c[A_i,\varepsilon], d_L[i-1,j-1]+c[A_i,B_j])$$

for $i \geq 1$ and $j \geq 1$,

with $d[0, 0] = 0$ and $d_L[i,-1] = d_L[-1, j] = \infty$

where: A_i is the i^{th} element of string A,

B_j is the j^{th} element of string B

Fig. 1. Levenshtein Distance Calculation.

Here, lexical entries are saved as lemmas in a dictionary. In order to meet the goal, two files (one for the Modern Greek language and one for the English language) are maintained which contain over 30,000 lemmas for searching. When this calculation is ended, the algorithm sends back a collection of lemmas including the minimum edit distance from the source word. E.g. close → closing, stir → stirred. One input source string is converted into targeted output string which will be expected to obtain into dictionary-lemma list after minimum substitution of alphabets from input-string, is the purpose of this lemmatization process [11].

The second approach is the Morphological Analyzer which is based on "Finite State Automata (FSA)". Morphological Analysis is the process of providing grammatical information of a word given its suffix. A morphological analyzer is a program for analyzing the morphology of an input word, it detects morphemes of any word [3-6]. Around 1981, Koskenniemi [3], Karttunen and Wittenburg [7]; Kaplan and Kay [13] had focused on morphological approach to generate lemma from morphed word. They worked on surface word/morphed word to get lemma, based on finite state transducer or finite state machine. Around 2001, John Goldsmith [6], University of Chicago adopted unsupervised learning of the morphological segmentation of European languages, using corpora ranging in size from 5,000 words to 500,000 words.

Based on literature survey, it was studied that Harris, (1955) had first discussed about structural phonemes, utterance, and successor variety and word boundaries in structural and linguistics domain [1].

Hafer and Weiss described a method for automatically segmenting words into their stems and affixes. The process uses certain statistical properties of a corpus (successor and predecessor letter variety counts) to indicate where words should be divided [2].

The approach used in this study is based on Z. S. Harris' process for breaking phonetic text into its constituent morphemes.

Based on Harris and Hafer's approaches, two-level KIMMO model [3, 4], GoldSmith's unsupervised morphological model [6] and Stanford morphological analyser (LemmaProcessor) [7, 8, 9] were developed. G.J. Russell and S.G. Puhnan had also developed a morphological analyzer for generating morpheme from complex word [10].

The popular Lemmagen [12], the third approach uses "Ripple Down Rule" data structure which allows to retrieve possible lemma of a given inflected or derivational form. Ripple-down rules are an incremental approach to knowledge acquisition.

III. THE PROPOSED LEMATIZER: THE LEMMACHASE

After pre-processing the text file, the proposed model with its methods for different input are applied for getting lemma from its derived or morphed words.
Steps for File Pre-processing:
– Remove all unnecessary characters.
– Convert all words into lower case.
– Split sentences into words/tokens.
– Remove all stop words from the word/tokens list.
– Generate a Unique Word List and tag all unique words using Stanford POS tagger based on PEN bank POS List.

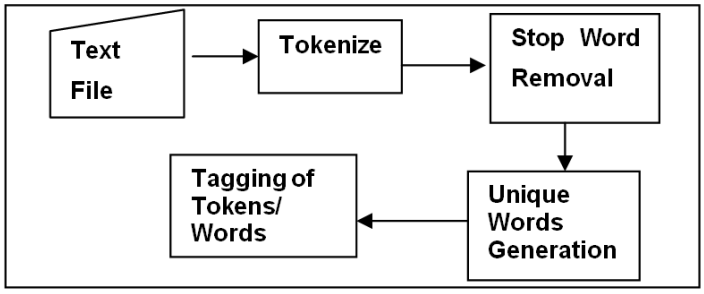

Fig. 2. Flow of pre-processing task of LemmaChase.

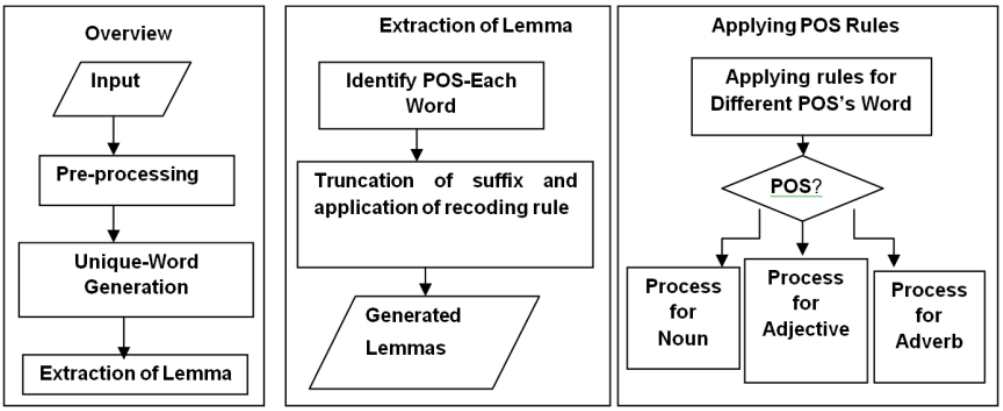

Fig. 3. Flowchart of LemmaChase.

In this proposed lemmatization model, following steps are designed to get lemma from their surface words.
Proposed model's steps are described below: Based on POS of words, the following different rules are applied for restructuring all input-words. The Flow Chart of Lemmatization process is given in Fig. 3.
1. Word with Proper Noun tagging, is not processed further, same input word will be identified as its lemma. e.g.Word "India" has lemma as "India".
2. If word's POS is identified only as verb in any tense from WordNet dictionary, [16] lemma for above mentioned input from the WordNet is selected as final freezed lemma e.g. words "achieved", "went" and "programmed" exist as only verb POS in dictionary. Lemma "achieve", "go" and "program" are selected as the lemma for the given input.
3. If the input word is identified as both Noun and Verb POS form, lemma of Verb form's word is selected as lemma if and only if length of lemma <= length of input word. E.g. "programming" and "left" exists as both VERB and NOUN.
If input word always exists as a noun in any dictionary and its extracted lemma is same as that word itself, then such a condition indicates that this lemma could be correct or incorrect. For nominalized noun, correct lemma cannot be extracted using WordNet dictionary. The following steps should be implemented in such cases. Lovins' stemming "Longest-match" principle is applied on input-word for suffix/endings identification (e.g. -al, -al after –l, -al with duplicate characters, -ance, -ence, -ion, -ication,-ion, -ism, -ship, -(p)tion, -ure, -ment, -age, -cation, -ief) and recoding rules are applied for restructuring input words.
— "The longest-match principle states that within any given class of endings, if more than one ending provides a match, the one which has longest match, should be removed." In case of suffix "-ion" and "-ation", suffix "-ation" will get more priority to be removed. After removing endings/ suffix from input word, stem token's length should be maintained at minimum 2 for forming a valid word/token, according to Lovins' stemming algorithm [16].
— If trimmed stem-token exists as VERB POS word in WordNet dictionary, its lemma is extracted and selected as lemma of input-word. If the trimmed stem-token does not exist as a valid dictionary word in any POS, in WordNet dictionary, then new token is generated using the recoding rule given in Table 1. The recoding rules are different for different class of endings for generating a new token. If, the token exists as a valid word in WordNet dictionary and also exists either as a VERB or as a NOUN form, the lemma of the word which is in VERB form is selected as the final lemma of input-word. If it is only in NOUN form(e.g. history, world), then

the input word is selected as the final lemma. If, the token is not existing as a valid word in WordNet dictionary, original input word is treated as a lemma of input word.

4. If, input word is identified as both Adjective and Verb POS form, lemma of Verb form's word is selected as proposed model's lemma. If input word is only identified as Adjective (e.g. nice, happy, and historical) and its extracted lemma is identical with input-word, such condition indicates that this lemma is either correct or incorrect. For nominalized adjectives, correct lemma cannot be extracted using WordNet dictionary [15]. Lovins' stemming [16] "Longest match" principle is again applied for adjectives (e.g. -able, -en, -ious, -ful, -AL, -ary, -ic, -ical, -y, -ed) [14,16]. Same type of recoding rules are also applied for adjectives, which is done on words of Noun POS. Suffix list and recoding rules based on suffix are depicted on Table 2.

Table 1: Noun with suffix list and Recoding rule.

Seq. No.	Suffix/ endings	Recoding [14, 16]	Nominalized Noun	Root word
1.	-al	-e	Removal, approval	Remove, approve
2.	-al after -i	-y	Denial	Deny
3.	-al with Dup char	—	Dismissal	Dismiss
4.	-ance	-/-e	Appearance, insurance	Appear, insure
5.	-ence	—	Preference	Prefer
6.	-ion	-/-e	Education, rotation	Educate, rotate
7.	-ication	-/-y	Identification, application	Identify, apply
8.	-ion	-/-e	Education,	Educate
9.	-(p)tion -ption	-/-e/-be /<-en>--> <-ain>	Infection, donation, description, retention	Infect, donate describe, retain,
10.	-ure	-/-e	Departure, enclosure	Depart, enclose
11.	-ment	-/-e	Employment, argument	Employ, argue
12.	-age	-/<-i>-><-y> / <-gg>-> <-g>	Breakage, passage, baggage, marriage	Break, pass, bag, marry
13.	-ery	-/<-e>	Creamery, machinery	Cream, machine
14.	-ief	-iev	Belief	Believe
15.	-ty	-	Safety, cruelty	Safe, cruel
16.	-ity	-e/-	Activity, stupidity	Active, stupid
17.	-ism	-/-e	Absenteeism, absolutism	Absentee, absolute
18.	-EE	-/-e	Appointee, payee, divorcée	Appoint, pay, divorce
19.	-ist	—	Tourist, typist, machinist	Tour, type, machine
20.	-sion	-se/s/-t, <-s>-><-t>	Revision, extension, expression, emission	Revise, extend, express, emit
21.	-y	<-c>-> <-te>/-	Privacy, difficulty	Private, difficult
22.	-TH	<-e>/<-d>/ <-ir>-><-orn>/<-eng>-><-ong>	Width, death, birth, length, growth	wide, dead, born, long, grow
23.	-OR	-/-e	Actor, collector, dictator	Act, collect, dictate
24.	-ster	-	Youngster, oldster	Young, old

5. If the input word is identified as both Adverb and Verb POS form, lemma of Verb form's word is selected as lemma. If input word only exists as an Adverb POS in any dictionary (e.g. gracefully, gently) and it may exist as a nominalized adverb. For nominalized adverb, same suffix truncation rules and recoding rules are applied for getting correct lemma for input adverb words. Some suffix list (e.g. -ly, -ally,-ably, -ically, -wise, -y) and recoding rule detail are given in Table 3. If input word is not nominalized adverb, word (e.g. together, instead) itself is selected as lemma.

With the help of above mentioned steps and using the suffix and recoding rules mentioned in the Table 1, 2, 3, maximum allied nominalized words as well as allied morphed words are merged into their corresponding dictionary base words or morphological root words which is called lemma. With the help of this proposed model, new valid word generation is also possible from any morphological input word. Normal (bird-birds/box-boxes) and unusual (fungus-fungi/woman-women) singular-plural, verb in any tense (run-ran/go-went) and nominalised words (acceptance-acceptability/application-applicability/ add-addition) can accurately extract their root word/lemma using this proposed model.

Table 2: Adjective with suffix list and Recoding rule.

Seq. No	Suffix/ endings	Recoding [14]	Nominalized Adjective	Root word
1.	-able	-/-e	Explainable, advisable, believable	Explain, advise, believe
2.	-able	-ic->-y	Applicable, achievable	Apply, achieve
3.	-ible	-/-e	Accessible, flexible, forcible, sensible	Access, flex force, sense
4.	-ant	– /-e/-y	Pleasant, reliant, resistant	Please, rely, resist
5.	-en	-e	chosen	Choose
6.	-ious	-y	envious	Envy
7.	-ful	—	Hateful, beautiful	Hate, beauty
8.	-AL	—	Accidental	Accident
9.	-ary	—	Customary	Custom
10.	-ic	-/-e/ <-tif>-><ce>/ -y	Athletic, basic, scientific, academic	Athlete, base, science, academy
11.	-y	-/<->-> <>	Rainy, messy, funny	Rain, mess, fun
12.	-ed	-/-e	Amused, Relaxed	Amuse, relax
13.	-IVE	-/<-t>-><-d>	Attractive, possessive, attentive	Attract, posses, attend
14.	-ly	-	Lively	Live
15.	-ion	-	Collection	Collect
16.	-ical	-/-icine/-ics	Musical, medical, political	Music, medicine, politics
17.	-ful	-/-e, <-i>-><-y>	Beautiful, wonderful, Awful	Beauty, wonder, awe
18.	-al	-e	Universal	Universe
19.	-ical	-ic/-ice/-ics/-y	Magical, practical, statistical, historical, alphabetical	Magic, practice, statistics, history, alphabetic
20.	-ous	-/<e>-><-y>/<-e>	Poisonous, courteous, mysterious, nervous	Poison, courtesy, mystery, nerve
21.	-ish	—	Foolish, sheepish, childish, selfish	Fool, sheep, child, self
22.	-ly	-/<-i>-> <-y>	Friendly, daily, costly	Friend, day, cost

Table 3: Adverb with suffix list and Recoding rule.

Seq. No	Suffix/ ending	Recoding [16]	Nominalized Adverb	Root word
1.	-ly	<-i>-> <-y>, <-le>	Angrily, completely, busily, gently	Angry, complete, busy,gentle
2.	-ly	<-ful>-><->, <-l>-<-le>	Gracefully, wholly	Grace, whole
3.	-ly	<-y>-><-y>	Shyly	Shy
4.	-ly	-le	Nobly, definitely	Noble, definite
5.	-ally	-	Academically	Academic
6.	-ably	-	Fashionably	Fashion
7.	-ically	-ic/-ics	Classically, politically,	Classic, politics,
8.	-wise	-	Clockwise, edgewise	Clock, edge
9.	-ily	-ar->-eer	Voluntarily	Volunteer
10.	-y	-	Holey, smiley	Hole, smile

IV. DISCUSSION OF OUTPUT

Table 4: Sample Output of LemmaChase for Nominalized words.

Sequence	Word Noun	Output of LemmaChase	Word Adjective	Output of LemmaChase	Word Adverb	Output of LemmaChase
1.	acceptance	accept	acceptable	accept	angrily	angry
2.	achievement	achieve	achievable	achieve	academically	academy
3.	action(Verb)	act	acting	act	rightly	right
4.	activity	act	actionable	act	busily	busy
5.	legalization	legal	academic	academy	legally	legal
6.	judgment	judge	judgmental	judge	hopefully	hopeful

7.	addition	add	additional	add	easily	easy
8.	adjustment	adjust	adjustable	adjust	idly	Idle
9.	admiration	admire	admirable	admire	artistically	artistic
10.	amazement	amaze	amazing (Verb)	amaze	rapidly	rapid
11.	annoyance	annoy	annoying (Verb, Noun)	annoy	tragically	tragic
12.	attention	attend	attentive	attend	classically	classic
13.	attraction	attract	attractive	attract	really	real
14.	advisement	advise	advisable	advise	luckily	Lucky
15.	avoidance	avoid	avoidable	avoid	magically	magical
16.	comfort(Verb)	comfort	comfortable	comfort	politically	politics
17.	cheerfulness	cheer	cheerful	cheer	maturely	mature
18.	confusion	confuse	confusable	confuse	voluntarily	volunteer

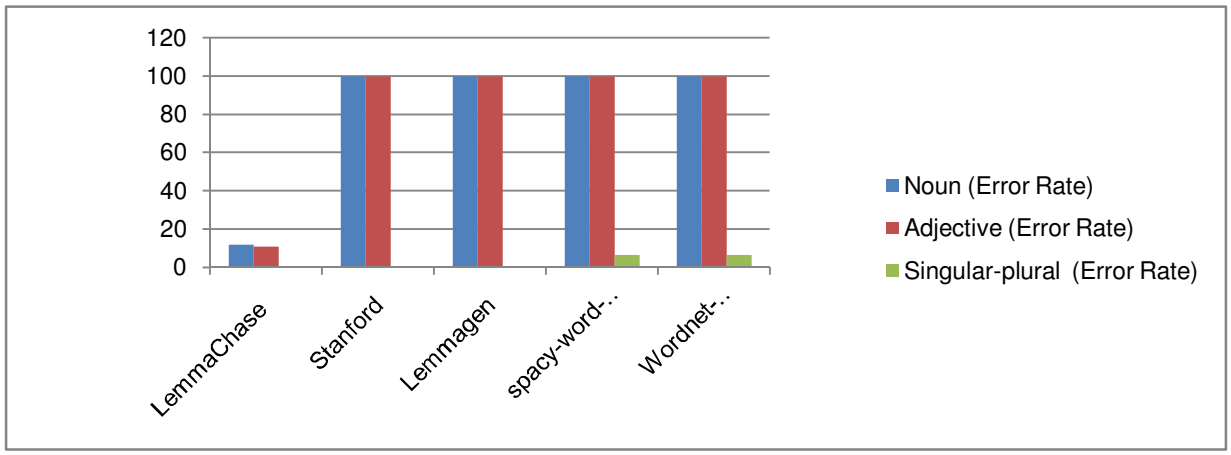

Fig. 4. Comparison between the rate of errors from LemmaChase and from other existing Lemmatizers.

Table 5: Existing Lemmatizers' Output for Nominalized words.

1. For LemmaGen Lemmatizer [released 2010, Online NLP Tool] :
Input: "Acceptance achievement action activity achievable judgment addition adjustment admiration amazement annoyance attention".
Output: "Acceptance achievement Action activity Achievable judgment addition adjustment admiration amazement annoyance attention." [No Change]

2. Text Analysis Result -- spaCy Word Lemmatize: [released 2016, Online NLP Tool]
Original Text
Acceptance achievement action activity achievable judgment addition adjustment admiration amazement annoyance
Analysis Result
acceptance achievement action activity Achievable judgment addition adjustment admiration amazement annoyance
[No Change]

3.Stanford CoreNLP XML Output [**released 2010**, java command **]**

Sentences

Id	Word	Lemma	Char begin	Char end	POS
1	Acceptance	acceptance	0	10	NN
2	Achievement	achievement	12	23	NN
3	Action	action	25	31	NN
4	Activity	activity	33	41	NN
5	Activeness	activeness	43	53	NN
6	Acceptable	acceptable	55	65	JJ
7	Achievable	Achievable	67	78	JJ
10	Activeness	activeness	98	108	NN
11	Addition	addition	110	118	NN
12	Adjustment	adjustment	120	130	NN
13	Admiration	admiration	132	142	NN

4. Lemmatization with Python NLTK: [Online NLP Tool] This is a demonstration of **stemming** and **lemmatization** for the 17 languages supported by the NLTK 2.0.4 **stem** package.

Stem TextChoose stemmer WordNet Lemmatizer

acceptance
achievement
action
activity
activeness
acceptable
achievement
action
activity
activeness
addition
adjustment

Stem

16 amazement amazement 158 167 NN ***Sentence #1****Tokens*	**Stemmed Text:** acceptance achievement action activity activeness acceptable Achievable action activity activeness addition adjustment admiration advice mass amazement amusement annoyance approach attention attraction voidance belief blackness [No Change]
5. English Lemmatizer: MorphAdorner V2.0: (Last update: October 6, 2013 , Northwestern University Information Technology) [Online tool]: No change for above set of Input (Nominalized words) **Input : Output** Homogeneously : Homogeneous Microscopically : Microscopical	**6. BioLemmatizer based on MorphAdorner (released at 2012) for BioMedical words :** **Input : Output** anlagen, NN; : anlage spermatogonia, NN : spermatogonium

BioLemmatizer (2012), as shown in table 5 above is also a lemmatization tool which handles bio-medical terms as well as some derivational adjectives and adverbs, based on morphological analyzer, MorphAdorner2.0 [21,22]. Output of both, BioLemmatizer and MorphAdorner is shown in Table. 5. The figure 4 shows the comparison between the rate of errors of the proposed lemmaChase with other existing lemmatizers, clearly lemmaChase has very less error rate. Katharina Kann and Hinrich Schutze have also developed a slightly different analyzer named, morphological re-inflection (MRI) model which produces inflected form from given source word-form (e.g. “trees” from “tree) at Association for Computational Linguistics, August 2016 [23].

V. CONCLUSION

The output of LemmaGen, Stanford, spaCy and WordNet Lemmatizers are depicted in Table 5. It is obvious from the details described above that the proposed lemmatizer-LemmaChase handles nominalized words perfectly. All input words from dictionary or any text of corpus, when processed through LemmaChase, extracts the correct dictionary root word or lemma using WordNet dictionary and also taking care of input word’s POS at the time of processing. It handles most of the morphed and derived words (singular, plural in Noun, irregular singular-plural/ verbs in any tense/adjective/ adverbs), appeared as any POS form in a text. This model specifically focuses on nominalised words. What can be said to be a limitation of this model is it finds the lemma for the input words of filtered text individually after removing stop words, proper nouns and numeric values and using the WordNet dictionary. The research gap in this work lies in the fact that there needs to be some mathematical or statistical operation which assists in finding the lemma. Moreover the way Marine Schmitt and Mathieu Constant have gone further in implying that ‘apple pie’ (Multi-Word Expression: MWE) is actually to be evaluated as ‘pie made from apple’ is an aspect of natural language processing which could be incorporated in an Intelligent Lemmatizer. MWE model is successfully executed for Finnish, Polish and Italian languages and is published at Proceedings of the Joint Workshop on Multiword Expressions and WordNet (MWE-WN 2019), August, 2019 [24]. So the challenge now lies in developing such a leammatizer.

VI. FUTURE SCOPE

This work can be carried forward by including some statistical based computation, to create individual group for single class morphed words which can reduce the processing task and time of lemma searching. It should allow all single class morphed words to be merged into a single related lemma. Correct grouping of single lemma variant words will lead to accurate lemma generation.

REFERENCES

[1]. Harris Z. S. (1970). Distributional Structure. In: Papers in Structural and Transformational Linguistics. Formal Linguistics Series. Springer, ISBN978-94-017-6059-1.

[2]. Hafer, M. A., & Weiss, S. F. (1974). Word segmentation by letter successor varieties. *Information storage and retrieval*, *10*(11-12), 371-385.

[3]. Koskenniemi, K. (1983).Two-Level morphology: Model for morphological analysis A general Computational Model for word-form recognition and production.

[4]. Koskenniemi, K. (1984). A general computational model for word-form recognition and production. In *Proceedings of the 10th international conference on Computational Linguistics* (pp. 178-181). Association for Computational Linguistics.

[5]. Koskenniemi, K. (1984). Two-level morphology: A general computational model for word-form recognition and production. Publications / Department of General Linguistics, University of Helsinki, ISBN-0: 9514532015.

[6]. Goldsmith, J. (2001). Unsupervised learning of the morphology of a natural language. *Computational linguistics*, *27*(2), 153-198.

[7]. Karttunen, L., & Wittenburg, K. (1983). A two-level Morphological analysis of English: Texas Linguistic Forum 22, 217-228.

[8]. Karttunen, L., & Beesley, K. R. (2005). Twenty-five years of finite-state morphology. *Inquiries Into Words, a Festschrift for Kimmo Koskenniemi on his 60th Birthday*, 71-83.

[9]. Klein, D., & Manning, C. D. (2003). Fast exact inference with a factored model for natural language parsing. In *Advances in neural information processing systems*, 3-10.

[10]. Russell, G. J., & Puhnan, S. G. (1984). A Dictionary and Morphological Analyser for English: SERC/Alvey commencing Project, 277-279.

[11]. Lyras, D. P., Sgarbas, K. N., & Fakotakis, N. D. (2007). Using the levenshtein edit distance for automatic lemmatization: A case study for modern greek and english. In *19th IEEE International Conference on Tools* with Artificial Intelligence (ICTAI 2007), 2, 428-435. IEEE.
[12]. Juršic, M., Mozetic, I., Erjavec, T., & Lavrac, N. (2010). Lemmagen: Multilingual lemmatisation with induced ripple-down rules. *Journal of Universal Computer Science*, *16*(9), 1190-1214.
[13]. Kaplan, R. M., & Kay, M. (1994). Regular models of phonological rule systems. Computational linguistics, 20(3), 331-378.
[14]. Francis Katamba (1994). English Words: by Routledge: ISBN 0-415-10468-8 (pbk).
[15]. Morato, J., Marzal, M. A., Lloréns, J., & Moreiro, J. (2004). Wordnet applications. In *Proceedings of GWC,* 270–278.
[16]. Lovins, J. B. (1968). Development of a stemming algorithm. *Mech. Translat. & Comp. Linguistics*, *11*(1-2), 22-31.
[17]. Jivani, A. G. (2011). A Comparative Study of Stemming Algorithms, *et al.*, *Int. J. Comp. Tech. Appl.*, *2*(6), 1930-1938.
[18]. Gupta, R., & Jivani, A. (2014). Empirical analysis of affix removal stemmers. *Int. Computer Technology. Appl., 5*(2), 393-399.
[19]. Gupta, R. & Jivani, A. G. (2017). Analyzing the Stemming Paradigm: Information and Communication Technology for Intelligent Systems, 2, 333-342.
[20]. Antony, P. J., & Soman, K. P. (2012). Computational morphology and natural language parsing for Indian languages: a literature survey. *International Journal of Scientific and Engineering Research*, *3*.
[21]. Haibin Liu, Tom Christiansen. (2012). BioLemmatizer: a lemmatization tool for morphological processing of biomedical text. Published in J. Biomedical Semantics. Computer Science, Medicine Journal of Biomedical Semantics, DOI:10.1186/2041-1480-3-3. Corpus ID: 15618342.
[22]. MorphAdorner (2013). A Java Library for the Morphological Adornment of English Language Texts Version 2.0.1. Copyright © 2007, 2013 by Northwestern University.
[23]. Katharina Kann and Hinrich Schutze (2016). Single-Model Encoder-Decoder with Explicit Morphological Representation for Reinflection. Center for Information & Language Processing. LMU Munich, Germany. Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pages 555–560.
[24]. Marine Schmitt, Mathieu Constant (2019). Neural Lemmatization of Multiword Expressions. Proceedings of the Joint Workshop on Multiword Expressions and WordNet(MWE-WN2019), Association for Computational Linguistics, Florence, Italy, August 2, 2019. pages 142–148.

How to cite this article: Gupta, R. and Jivani, A. G. (2020). LemmaChase: A Lemmatizer. *International Journal on Emerging Technologies*, *11*(2): 817–824.