International Journal of Multidisciplinary Engineering in Current Research - IJMEC
Volume 7, Issue 10, October-2022, http://ijmec.com/

A Comprehensive Survey of Stemming Methods in Information Retrieval

Dr. Kirti Wanjale[1], Dr. Abhijit Chitre[2], Parth Patil[3], Ronish Parmar[4], Siddhant Raka[5], Samruddhi Ghattuwar[6]
[1]Associate Professor, Computer Engineering Department, [2]Associate Professor, Electronica and Telecommunication Department.
[3,4,5,6]Students, Computer Engineering Department, Vishwakarma Institute of Information Technology, Pune.

Abstract:

In applications for text mining, stemming is seen as a pre-processing stage. Additionally, it serves as a general requirement for the features of natural language processing. It is crucial in the majority of information retrieval systems. Getting back to a word's root form from its various grammatical forms, such as its noun, adjective, verb, adverb, etc., is the basic objective of stemming. We can argue that stemming's objective is to reduce a word's various forms, including those derived from other words, to its fundamental form. In this essay, we have discussed many approaches to determining a word's stem and contrasted them according to their benefits and drawbacks. It is also discussed how stemming and lemmatization differ from one another.

I. INTRODUCTION:

Word stemming is a vital feature supported by current indexing and search systems. Indexing and searching are, in turn, a part of Text Mining applications, Natural Language Processing (NLP) systems, and Knowledge Retrieval (IR) systems. The best way to improve recall is to handle word endings automatically by reducing words to their word roots when indexing and searching. Recall went up without changing how accurately the documents were searched. Before index terms are added to the index, any suffixes and prefixes (referred to as "affixes") are typically removed. The stemming technique will eventually lead to an increase in the number of documents discovered by the IR system because the root of a word is a more abstract concept than the word itself. This translation is necessary as part of the pre-processing that occurs before any algorithm is actually utilised for text grouping, categorization, and summarization.

II. WORKING OF STEMMER:

The majority of the time, it has been observed morphologically distinct word forms have comparable semantic interpretations and can be treated as being equivalent for the sake of IR applications. It is vital to match each word form with its base form because the meaning is the same, but the word form varies. Numerous stemming algorithms have been developed to accomplish this. Every algorithm tries to map a word's morphological variations, such as introduction, introducing, intro, etc., to the word "introduce."Some algorithms might translate them to "introduce," but this is OK as long as they all translate to the same word form, also known as the stem form. As a result, stems represent key phrases in queries and documents rather than the actual words themselves. The goal is to cut down on the total number of distinct terms in a document or query, which will reduce how long it takes to process the output.

III. ERRORS IN STEMMING:

Over-stemming and under-stemming are the two basic stemming mistakes. When two words with distinct stems stem from the same root, this is known as over-stemming. A false positive is another name for this.

When two words that should have the same root are not, this is known as under- stemming. It is referred to as a false negative as well. Paice has established that light stemming increases under-stemming errors while decreasing over-stemming errors. Heavy stemmers, on the other hand, increase the over-stemming errors while decreasing the under-stemming errors.

IV. CLASSIFICATION OF ALGORITHMS

In general, stemming algorithms fall into two categories: truncating methods, statistical approaches or mixed methods. Finding the word variations' stems is a typical method for each of these groupings. The Fig.1 displays these techniques and the algorithms presented alongside them in this research.

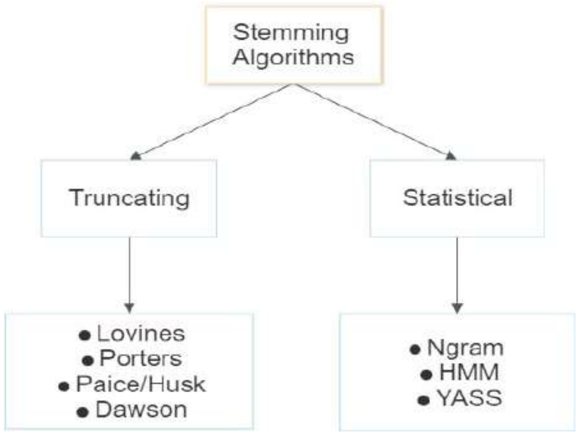


FIG.1 TYPES OF STEMMING ALGORITHMS

V. Truncating Methods (Affix Removal):

As the name implies, these procedures are concerned with eliminating suffixes or prefixes (usually a word sometimes referred to as affixes). The Truncate (n) stemmer, which kept the first n letters of a word and removed the rest, was the simplest type of stemmer. Words that are shorter than n are left alone using this strategy. When the word length is short, there is a higher likelihood of over-stemming. The S-stemmer algorithm, which combines the singular and plural forms of English nouns, was another straightforward method. Donna Harman proposed this algorithm. The algorithm provides rules for stripping plural suffixes and changing them to singular versions.

LOVIS STEMMER

In 1968, Lovins proposed the first widely accepted and successful stemmer. It conducts a lookup on a table of 294 endings, 29 conditions, and 35 transformation rules organized according to the most extended match principle. The Lovins stemmer eliminates a word's longest suffix. The word is recoded using a different table that makes numerous alterations to turn these stems into acceptable words after the ending has been deleted. It is a one-pass method, so it can never remove more than one suffix from a word. The advantages of this method are its speed and ability to handle the conversion of various irregular plurals, such as mouse and mice, index and indices, etc, from double letters to single letters. The Lovins method has limitations because it requires a lot of time and data. Additionally, a large number of suffixes are absent from the table of endings. It frequently fails to match the stems of words with similar meanings or build words from their stems, which sometimes makes it incredibly untrustworthy.

Porter Stemmer

One of the most often used stemming algorithms nowadays was first proposed in 1980 by Porter. The fundamental algorithm has undergone numerous adjustments and improvements, as well as suggestions. It is predicated on the notion that the roughly 1200 suffixes in the English language are primarily composed of a collection of smaller and simpler suffixes. It consists of five steps, and rules are applied within each step until one of them meets the requirements. The suffix is deleted appropriately and the subsequent action is taken if a rule is accepted. At the conclusion of the fifth stage, the resultant stem is given back.

The rule appears as follows:

<condition> Suffix + new suffix

For instance, the rule (m>0) If a word has at least one vowel and one consonant in addition to the EED ending, modify the ending to EE. As a result, "agreed" changes to "agree," while "feed" stays the same. This algorithm is relatively simple to understand and contains roughly 60 rules.

Porter created a thorough stemming framework known as "Snowball." The framework's main objective is to give programmers the freedom to create custom stemmers for different character sets or languages. There are currently implementations for a large number of Romance, Germanic, Uralic, Scandinavian, English, Russian, and Turkish languages.

Paice came to the conclusion that the Porter stemmer produces fewer error rates than the Lovins stemmer based on the stemming errors. The Lovins stemmer, on the other hand, is a heavier stemmer that yields superior data reduction. Due to its extremely long endings list, the Lovins method is substantially larger than the Porter algorithm. But it does have one benefit: it moves more quickly. With its vast suffix collection, it only requires two significant steps to delete a suffix, as opposed to the Porter algorithm's five. This effectively trades space for time.

PAICE/HUSK STEMMER

The Paice/Husk stemmer is an iterative technique where the last letter of a suffix is used to index a single table with around 120 rules. It searches for a relevant rule by the word's final character on each iteration.

Each rule details the substitution or deletion of an ending. If there isn't a rule like that, it ends. It also comes to an end if a word begins with a vowel and only has two letters left, or if a word begins with a consonant and only has three characters. If not, the rule is followed and the procedure is repeated. The benefit is that it has a straightforward structure, and each iteration handles both deletion and replacement according to the rule in effect. The drawback is that the method is highly complex and excessive stemming may happen.

DAWSON STEMMER

This stemmer is an extension of the Lovins technique, but it covers a more extensive list of approximately 1200 suffixes. Like Lovins, it is a one-pass stemmer and, as such, moves reasonably quickly. The suffixes are kept in reversed order, with their length and last letter serving as indexes. For quick access, they are put up as a collection of branching character trees. The benefit is that it executes quickly and covers more suffixes than Lovins. Its complexity and absence of a standardized, reusable implementation are drawbacks.

STATISTICAL METHODS

They are the stemmers that rely on statistical methods and analysis. The majority of techniques delete the affixes, but only after using a statistical procedure.

N-Gram Stemmer

This is a language-independent and highly fascinating method. Here, word inflation is reduced to its stem using a string-similarity method. A collection of n, typically contiguous, characters called an n-gram is taken from a segment of continuous text. An n-gram, to be precise, is a collection of n consecutive characters taken from a word. This method's key presumption is that words that are similar to one another will share a large percentage of n-grams. The words that are extracted when n is 2 or 3 are referred to as digrams or trigrams, respectively. For instance, the word "INTRODUCTIONS" generates the following digrams:

*I, IN, NT, TR, RO, OD, DU, UC, CT, TI,
IO, ON, NS, S*

while the trigrams:

I, *IN, INT, NTR, TRO, ROD, ODU,DUC, UCT, CTI, TIO, ION, ONS, NS*, S

Where "*" indicates a space for padding.

In a word with n characters, there are n+1 such digrams and n+2 such trigrams. Language-specific stemmers comprise the majority. Typically, n is chosen to be between 4 and 5. Textual data or document is then examined for every n-gram. A word root obviously appears less frequently than its morphological form in general. This implies that a word typically has an affix attached to it. They can be located using a standard statistical study based on the inverse document frequency (IDF). This stemmer has the benefit of being language-neutral, making it particularly beneficial in many applications. The drawback is that it is not a very practical solution because it needs a lot of memory and storage to create and store the ngrams and indexes.

HMM STEMMER

This stemmer is based on the notion of a finite-state automaton called a Hidden Markov Model (HMM), which uses probability functions to control state transitions. The new state emits a symbol with a specific probability at each transition. Melucci and Orio proposed this paradigm.

This approach uses unsupervised learning and doesn't require any prior linguistic familiarity with the dataset. The Viterbi coding in the automata graph is used in this method to determine the most likely path by computing the probability of each path.

When using HMMs for stemming, a word's letter sequence can be thought of as the product of the concatenation of two subsequences: a prefix and a suffix. An HMM that divides the states into two distinct sets—the first set can be the stems only, and the latter set can be the stems or suffixes—is one technique to simulate this process. Word-building processes are defined by transitions between states. This approach allows for the following assumptions:

1. Initial states are exclusive to the stem set; every word begins with a stem.
2. A word can only be made up of a stem anda suffix when they are concatenated. Hence transitions from states of the suffix set to states of the stem set always have a zero probability.
3. Final states are included in both sets; a stem may have a variety of derivations, or it may not have a suffix. The split point (a change from roots to suffixes) is produced by the most likely route from beginning to final states for any particular word.
4. The characters that came before this point can then be regarded as a stem.
5. This method has the benefit of not requiring language proficiency because it is unsupervised.
6. Its complexity and potential for word overstuffing are drawbacks.

YASS STEMMER

In the name, it stands for Yet Another SuffixStriper. According to the authors, the performance of a stemmer developed by clustering a lexicon without any linguistic input is comparable to that obtained using conventional, rule-based stemmers such as Porter. This stemmer falls within the statistical and corpus-based categories. It does not require language knowledge. The authors' retrieval tests on datasets in Bengali, English, and French demonstrate that the suggested strategy works well for languages that are largely suffixing in nature.

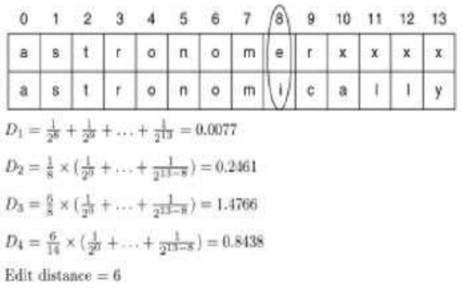
A hierarchical technique and distance measurements are used to build the clusters. The centroids of the generated clusters are regarded as the stems, and the clusters themselves as equivalence classes. The edit distance and YASS distance estimates fortwo string comparisons are displayed in Figs.In accordance with the information provided there. The Boolean function pi for penalty serves as the foundation for the YASS distance measures D1, D2, D3, and D4. It's outlined as:

$$p_i = \begin{cases} 0 \text{ if } x_i = y_i \quad 0 \le i \le \min(n, n') \\ 1 \text{ otherwise} \end{cases}$$

Where X and Y are two strings, X = x0x1x2
. . . xn and Y = y0y1y2 . . . yn. If the strings are not the same length, we add nullcharacters to the shorter string to make themthe same length.

A smaller distance measurement indicates the greater similarity between the strings.

The number of actions necessary to change one character string into another determines the edit distance between two sequences of characters.



0	1	2	3	4	5	6	7	8	9	10	11	12	13
a	s	t	r	o	n	o	m	e	r	x	x	x	x
a	s	t	r	o	n	o	m	i	c	a	l	l	y

$D_1 = \frac{1}{2^8} + \frac{1}{2^9} + \ldots + \frac{1}{2^{13}} = 0.0077$

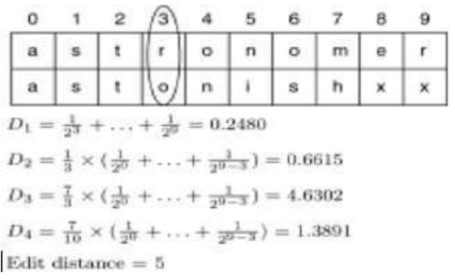
$D_2 = \frac{1}{8} \times (\frac{1}{2^0} + \ldots + \frac{1}{2^{13-8}}) = 0.2461$

$D_3 = \frac{6}{8} \times (\frac{1}{2^0} + \ldots + \frac{1}{2^{13-8}}) = 1.4766$

$D_4 = \frac{6}{14} \times (\frac{1}{2^0} + \ldots + \frac{1}{2^{13-8}}) = 0.8438$

Edit distance = 6

FIG.2 CALCULATION OF DISTANCE MEASURES-1



0	1	2	3	4	5	6	7	8	9
a	s	t	r	o	n	o	m	e	r
a	s	t	o	n	i	s	h	x	x

$D_1 = \frac{1}{2^3} + \ldots + \frac{1}{2^9} = 0.2480$

$D_2 = \frac{1}{3} \times (\frac{1}{2^0} + \ldots + \frac{1}{2^{9-3}}) = 0.6615$

$D_3 = \frac{7}{3} \times (\frac{1}{2^0} + \ldots + \frac{1}{2^{9-3}}) = 4.6302$

$D_4 = \frac{7}{10} \times (\frac{1}{2^0} + \ldots + \frac{1}{2^{9-3}}) = 1.3891$

Edit distance = 5

Fig.3 Calculation of Distance Measures-2

As can be seen from the distances D1, D2, D3, and D4, astronomer and astronomically are more comparable than astronomer and astonish. The new distance measurements are more precise because the edit distance seems exactly the opposite.

VI. COMPARISON BETWEEN THE ALGORITHMS

Truncating (Affix Removal) Methods

LOVINS STEMMER

Advantages:	**Disadvantages:**
1) Fast – algorithm with only one pass. 2) Handles exclusion of the double letters in words such as 'getting' being transformed to 'get'	1) Time consuming. 2) Not all suffixes available.

PORTERS STEMMER

Advantages:	**Disadvantages:**
1) Its outcome is the best compared to other stemmers. 2) Failure rate is less.	1) Not every one of the stems that are made are real words. 2) It takes time because it includes at least five steps and sixty rules.

PAICE/HUSK STEMMER

Advantages:	Disadvantages:
1) Simple form. 2) Each time through, elimination and renewal are taken care of.	1) A fair amount of script. 2) It's possible that there's over stemming.

DAWSON STEMMER

Advantages:	Disadvantages:
1) Covers more suffixes than Lovins. 2) Fast in execution.	1) Very complex. 2) Lacks a standard implementation.

STATISTICAL METHODS

N-Gram Stemmer

Advantages:	Disadvantages:
1) Based on the idea of n-grams and string correlations. 2) Language independent.	1) Not time efficient. 2) The formation and classification of n-grams take up a significant amount of space.

HMM Stemmer

Advantages:	Disadvantages:
1) Based on the concept of Hidden Markov Model. 2) Unsupervised method and so is available in multiple languages	1) A complex method for implementation. 2) Over-stemming may occur in this method.

YASS Stemmer

Advantages:	Disadvantages:
1) Based on hierarchical clustering approach and distance measures. 2) It is also a corpus-based method.	1) It's hard to decide on a threshold for making groups. 2) Requires a significant amount of processing power

VII. Conclusion:

So, from given paper and study it clearly shows that all the stemming algorithms are nearly like each other but none of them is 100 percent fit in any situation if one algorithm is better in one field then other is better in some other field.
The Over-stemming and Under-stemming can be reduced if syntax, POS and semantic of words is taken into consideration.

VIII. References

[1] Eiman Tamah Al-Shammari, "Towards An ErrorFree Stemming", in Proceedings of ADIS European Conference Data Mining 2008, pp. 160-163.
[2] Frakes W.B. "Term conflation for information retrieval". Proceedings of the 7th annual international ACM SIGIR conference on Research and development in informationretrieval. 1984, 383-389.
[3] Frakes William B. "Strength and similarity of affix removal stemming algorithms". ACM SIGIR Forum, Volume 37, No. 1. 2003, 26-30.
[4] Faunchun Peng, Nawaaz Ahmed, Xin Li and Yumao Lu. "Context sensitive stemming for web search". Proceedings of the 30th annual international ACM SIGIR conference.