Artificial Intelligence Review (2020) 53:5559–5588
https://doi.org/10.1007/s10462-020-09828-3

Empirical evaluation and study of text stemming algorithms

Abdul Jabbar[1] · **Sajid Iqbal**[2] · **Manzoor Ilahi Tamimy**[1] · **Shafiq Hussain**[3] · **Adnan Akhunzada**[1]

Published online: 15 April 2020
© Springer Nature B.V. 2020

Abstract
Text stemming is one of the basic preprocessing step for Natural Language Processing applications which is used to transform different word forms into a standard root form. For Arabic script based languages, adequate analysis of text by stemmers is a challenging task due to large number of ambigious structures of the language. In literature, multiple performance evaluation metrics exist for stemmers, each describing the performance from particular aspect. In this work, we review and analyze the text stemming evaluation methods in order to devise criteria for better measurement of stemmer performance. Role of different aspects of stemmer performance measurement like main features, merits and shortcomings are discussed using a resource scarce language i.e. Urdu. Through our experiments we conclude that the current evaluation metrics can only measure an average conflation of words regardless of the correctness of the stem. Moreover, some evaluation metrics favor some type of languages only. None of the existing evaluation metrics can perfectly measure the stemmer performance for all kind of languages. This study will help researchers to evaluate their stemmer using right methods.

Keywords Natural language processing · Information retrieval · Text mining · Stemming algorithms · Stemmer evaluation methods · Urdu stemming

✉ Sajid Iqbal
sajidiqbal.pk@gmail.com

Abdul Jabbar
a.jabbar73@gmail.com

Shafiq Hussain
shafiqhussain@bzu.edu.pk

Adnan Akhunzada
akhunzadaadnan@gmail.com

[1] Department of Computer Science, COMSATS University Islamabad (CUI), Main campus, Park Road, Tarlai Kalan, Islamabad 45550, Pakistan

[2] Department of Computer Science, Bahauddin Zakariya University Multan, Multan, Punjab, Pakistan

[3] Bahauddin Zakariya University Multan (Sahiwal Sub-Campus), Multan, Punjab, Pakistan

1 Introduction

Performance evaluation is the primary method to find the effectiveness of algorithms and methods developed to solve different scientific problems. Efficient evaluation methods can boost the applicability of the solution. Performance evaluation methods describe and determine the extent to which the solution can achieve its intended goals.

It is an open challenge for computational linguistics researchers to assess the performance of NLP applications (Cambria and White 2014). Existing evaluation methods can be divided between two main categories: intrinsic and extrinsic methods (Gaidhane et al. 2015). In the intrinsic evaluation, performance measure of NLP applications and methods are compared with some gold standard results that are calculated using manual methods. For instance, a stem produced by a stemmer is compared with relevant dictionary stem developed by human experts. Whereas, in the extrinsic evaluation method, the performance is measured directly in any realistic scenario. For example, two stemmers are tested on the same dataset and achieved accuracy is compared in relative way.

An evaluation of the text stemming system has always a long debate (Brychcín and Konopík 2015). In past studies, various text stemming evaluation (TSE) methods are used by researchers. However, the manual assessment of stemmers require human effort that make it a challenging task (Suryani et al. 2018). The state-of-the-art TSE methods can be direct or indirect (Singh and Gupta 2016). Direct evaluation refers to text stemming error analysis, conflation ratio, and compression factors as mentioned by Abainia et al. (2017). Indirect evaluation refers to stemmer evaluation with respect to specific NLP application. Such methods usually use machine learning (ML) methods like K-Nearest Neighbor (KNN), Naïve Bayesian (NB) and Support Vector Machines (SVM) are used for text classification (Saeed et al. 2018a, b). Presently neural network based methods are showing better performance than tranditional ML methods. Performance evaluation methods used in Information Reterival (IR) domain are considered as indirect methods (Mustafa and Rashid 2018).

To perform stemming in different langauges, various stemmers have been proposed (Jabbar et al. 2018a, b). To evaluate a stemmer from multiple aspects, it is required to evaluate it through different methods i.e. direct and indirect ways. Direct evaluation methods only consider the conflation ratio for performance measurement (Brychcín and Konopík 2015; Sirsat et al. 2013). These methods measure their performance based on correct stemming ratio. They did not focus on false positives, false negatives and untouched words. This shortcoming leads to limited applicability of designed stemmers. In this work, we address and review state of the art TSE methods. We experimently prove that current evaluation methods provide a partial picture of stemmer performance.

The contributions of this work are three fold. Firstly, we present an extensive comparative study of existing stemmer evaluation methods to highlight their merits and demerits. Secondly, we perform various experiments to find the performance of different stemmers designed for Urdu language. Thirdly and lastly, through our experiments, we show that current evaluation methods provide partial view and some of the evaluation methods are language specific.

Remaing part of this paper is organized as follows. Section 2 provides the background of this study. In Sect. 3, different text stemming applications are discussed. Different stemming evaluation methods are described in Sect. 4. A deep analysis of evaluation methods is given in Sect. 5. In Sect. 6, we discuss challenges associated with text stemming evaluation

methods and future research directions in this context. Finally, the findings and conclusion is provided in Sect. 7.

2 Background

Stemming is a computational procedure in which affixes are truncated from conflated word to extract the root. This process may differ from application to application based on their role in that application. In text stemming, stemmers minimize the document index size to improve the efficienty of computation. An index of a document containing English words such as "accepts", "accepted" and "acceptance" can be mapped to one common root i.e. "accept". Text stemming is an integral part of many NLP applications and an evaluation of these systems is a very crucial and tedious task (Dahab et al. 2015). This section presents necessary definitions and descriptions, required to understand stemming evaluation process.

Morpheme Morpheme is the smallest grammatical unit of a language that cannot be further divided into smaller meaningful parts and are combined together to form meaningful words. This combination could be through inflection, derivation, and composition (Aronoff and Fudeman 2011). For example, a morpheme may consist of a word such as "acceptance" in which "accept", is a meaningful piece of a word, whereas; "ance" is a meaningless morpheme. It cannot be further divided into smaller meaningful parts.

Affix Affix is a morpheme that can be defined as word or letters which are attached to the root or stem on any position i.e. it may be at the end /start /on both sides of the word or anywhere in the middle of the word (Aronoff and Fudeman 2011). Affixes are used to produce inflections and derivative forms of a word. For instance "accepts", "accepting" and "accepted" in which "s", "ing" and "ed" are affixes and stem is "accept".

Inflectional and derivational affixes Inflectional morphemes refer to the modification of words and that change the grammatical categories such as tenses, singulars, plurals, masculine, feminine and neutrals. The derivation is the reverse of inflection; it constructs new words by adding affixes to a root word (Qureshi et al. 2018).

Stem It is a base morpheme to which other morphemes like affixes are attached (Aronoff and Fudeman 2011).

Root A root is like a stem, but it contains only two morphologically simple units. For instance, '*disagree'* is the stem of '*disagreement*' because it is the base morpheme to which '*ment'* affix is attached, but '*agree'* is the root (Aronoff and Fudeman 2011). With this definition, we will be using both "stem" and "root" alternativilly. If required, the context will clear the particular meanings.

Lemma and Lexeme.

Lexemes indicate a common morpheme in a variant form of a word. On the other hand, a lemma is a definite form that is chosen from the lexemes collection to characterize the lexeme. The lemma is a valid dictionary word (Singh and Gupta 2016). For instance, the words *'write', 'writing', 'wrote', 'writes'* and *'written'* are lexeme and *'write'* is *a* lemma.

2.1 Overview of stemming algorithms

In literature, researchers have categorized stemming algorithms in different ways. Al-Sughaiyer and Al-Kharashi (2004) studied stemmers for English and Arabic languages and

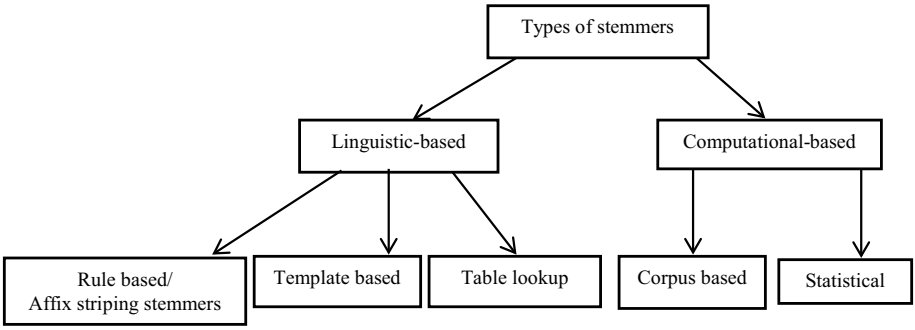

Fig. 1 Classification of stemming algorithms in terms of types

have classified the English stemmers into three main categories i.e. table lookup, linguistics, and computational. They classified Arabic stemmers into four groups i.e. table lookup, linguistics, computational and pattern based. According to Paik et al. (2011), stemmers can be categorized as rule-based and statistical stemmers. Jivani (2011) examined the English stemmers and classified them into three subcategories i.e. truncating, statistical and mixed. Zhou et al. (2012) divided the stemmers into two main categories (i.e., rules-based stemmers, and statistical stemmers). Moghadam and MohammadReza (2015) described the Persian stemmers and classified them into three classes: structural stemmers, table lookup stemmers and statistical stemmers. Singh and Gupta (2017) divided the stemming approaches into linguistic rule-based methods and language independent/statistical methods. Jabbar et al. (2018a, b) classified the stemmers into three classes: (a) linguistic-based stemmers, (b) corpus-based stemmers, and (c) hybrid stemmers. In general, the stemming algorithms can be classified as linguistic-based and computational-based stemmers as shown in Fig. 1. In linguistic-based stemmers, handcrafted grammatical rules are designed to derive the stem. For instance Saeed et al. (2018a, b) presented rule-based stemmer for the Kurdish language, and Suryani et al. (2018) developed Sundanese rule-based stemmer. On the other hand, computational stemmer performs some statistical or non-statistical computations. Corpus-based stemming measures the co-occurrence of variants word forms [e.g., Alotaibi and Gupta (2018)]. Similarly, in statistical stemmers, researchers have applied statistical and machine learning based procedures/techniques to extract the stem (Bölücü and Can 2019; Pande et al. 2018).

A good number of rule based stemmers for English language are proposed in literature (Lovin 1968; Porter 1980). Many researchers have attempted to improve the performance of Marting Porter's Stemmer (Bimba et al. 2016), due to its effectiveness for many NLP applications (Chintala and Reddy 2013; Patil and Patil 2013). Examples of rules and resource development for rule based stemmers are given by Jabbar et al. (2016) where the authors have developed some resources for Urdu language processing. According to table lookup/ references lookup (also called brute force approach), the word and corresponding stem are saved in the form of a table (Hussain et. al. 2017). This approach is suitable for those languages which have very complex linguistic structures. In statistical stemmers, several statistical features are extracted from given dataset. By using these features, the stem of the query word is obtained using statistical classifiers. Statistical methods are also known as language independent methods. Hybrid stemmers are constructed by combining two or more stemming methods. For instance, Bimba et al. (2016) stemmer combined the

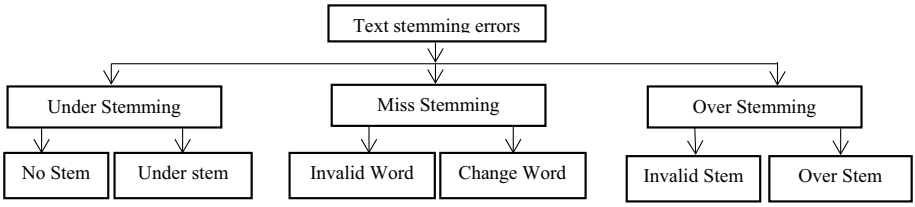

Fig. 2 Overview of stemming errors

Table 1 Examples of under stemming errors

Input word	Actual stem	Produced stem	Types of error
Acceptance	Accept	Acceptance	No stem
Acceptances	Accept	Acceptance	Under stem

Table 2 Example of over stemming errors

Input word	Actual stem	Produced stem	Types of error
receiving	receive	receiv	Invalid stem
consistently	consist	consistent	Over stem

Table 3 Example of mis-stemming errors

Input word	Actual stem	Produced stem	Types of error
Red	Red	r	Invalid word
kneel	Kneel	knee	Change word

rule base and table lookup approaches for the Hausa language. Jabbar et al. (2018a, b) also constructed an Urdu stemmer using a hybrid approach.

2.2 Text stemming errors

Recognizing the types of errors, a stemmer may produce, is the first step to measure the effectiveness of given stemmer. These types of errors can help to find the answers of questions like when and why they are occurred and what is their affect on stemmer performance? Fig. 2 gives the categorization of stemming errors.

Under stemming errors (USE) It refers to the fact when a stemmer strips the letters under acceptable level. In this type of errors, the stemmer either produces the word as it (no stem) or the process of affix removal generate the word with changed meaning as shown in Table 1.

Over stemming errors (OSE) OSE is an error in which a stemmer truncates more characters than required. OSE error leads toward invalid stem or out of vocabulary (OOV) word (Table 2).

Mis-stemming errors (MSE) The term "Mis-stemming errors" refers to those errors in which the stripped characters do not make proper affix (Table 3).

3 Applications of stemming algorithms

A stemming is a morphological analysis and necessary preprocessing step for NLP applications (Hassani and Lee 2016). To model a language, researchers extract different type of features (manual or automatic) from given data (Brychcín and Konopík 2015). There are variety of NLP applications and each type of application may require different type of features. For example, a language expert may utilize stemmer for vocabulary learning and development (Mochizuki and Aizawa 2000). Sarma and Purkayastha (2013) and Dang et. al. (2013) have used stemming for word classification and wordnet development respectivelly. Domain specific words extraction is performed by Rehman et. al. (2013) and Nguyen and Leveling (2013). Vocabalry mismatch problem can also be solved with the help of stemming (Singh and Gupta 2016). Applications like Information Extraction (IE), Information Reterival (IR), Text Classification (TC), Text Clustering (TClu), Question Answering (QA), Text Summarizations (TS), Machine Translation (MT), Text Segmentation (TS), Indexing (Ind), Automatic Speech Recognition (ASR) (Dahab et al. 2015; Singh and Gupta. 2016) and language generation (Mishra and Prakash 2012) require stemming as preprocessing step. In short, stemming improves the performance by reducing time and space complexity for several NLP applications (Boudchiche and Mazroui. (2015). A summarized view of applications of text stemming systems is provided in Table 4.

4 Stemming evaluation methods

In this section we review the representative stemming evaluation methods. To the best of our knowledge, we have included all available stemming evaluation methods and are summarized in Table 5.

Text stemming evaluation methods can be categorized as direct, indirect and gold standard evaluation metrices as shown in Fig. 3.

4.1 Direct evaluation methods

These evaluations methods used training datasets to extract required statistics known as features to perform stemming. These statistics may include over stemming index (OI), under stemming index (UI), Index compression factor (ICF), Average Words Conflation Factor (AWCF), and few more statistical significance metrics. In this subsection, we analyze all these different direct evaluation methods and make a discussion on our analysis.

4.1.1 Paice's (1994, 1996) evaluation methodology

Paice (1994, 1996) proposed first stemmer which is based on error counting and predefined groups of words that are related to each other either morphologically or semantically. A good stemmer conflates as many words as possible in a predefined group and avoids conflating different class words or words that are semantically distinct. Using this method, performance of proposed stemmer is measured with under-stemming index (UI), over-stemming index (OI) parameters, their ratio and the stemming weight (SW). To determine

Table 4 State-of-the-art applications of text stemming algorithms

No. 1	Cited	Application	Description
1	Schofield and Mimno. (2016), Hassani and Lee (2016)	Language modeling	Stemming may be viewed as a system of smoothing and as a way of better statistical estimation
2	Schofield and Mimno (2016)	Topic modeling	Stemmers can reduce the vocabulary size and topic modeling depends upon sparse vocabulary. So, it is also leveraged in topic modeling as a preprocessing step
3	Boukhalfa et al. (2018)	Plagiarism detection	Stemming enhances the performance of the similarity detecting system
4	McCormick (2016)	Word embedding	Two words have similar context must have the same word vector such as "ant" and "ants" have a similar context that is possible when a stemmer stem "ants" to "ant"
5	Sarma and Purkayastha (2013)	Word classification	Stemming also improves the efficiency of word classification applications
6	Dang et al. (2013)	WordNet development	Stemming deals with a variant form of words and each form of word belongs to a specific part of speech that is helpful in WordNet development
7	Nguyen and Leveling (2013)	Domain-specific words extract from the text	Domain-specific words have a specific form or specific affix and stemming facilitates to identify these words form or affix
8	Saeed et al. (2018a, b) Ismailov et al. (2016) Karimi et al. (2015)	Text mining	The goal of text mining is to extract meaningful information from text data
9	Dey et al. (2014)	Named entity recognition (NER)	Named entity recognition (NER) system seeks and extract the predefined proper and common nouns entities from the natural language text
10	Rehman et al. (2013)	Word segmentation	Stemming can be viewed as word tokenization from the continuous text
11	Dahab et al. (2015), Singh and Gupta (2016)	Information extraction (IE)	Stemming also improves the efficiency of an information extraction system
12	Flores and Moreira (2016), de Oliveira and Junior (2018)	Information retrieval (IR)	IR system uses a variant form of the words via stemmer

Table 4 (continued)

No. 1	Cited	Application	Description
13	Rani et al. (2015), Ali et al. (2018), Saeed et al. (2018a, b)	Text classification (TC)	Stemming can be viewed as text classification mechanism because it groups the words that share the same morphological root
14	Khalid et al. (2016), Dahab et al. (2015)	Text clustering (TClu)	A bag of words can be grouped by the stemming system
15	Giachanou and Crestani. (2016), Yadollahi et al. (2017)	Sentiment analysis	Pre-processing includes recognition and deletion of stop words, slangs, abbreviations, stemming and correction
16	Khalid et al. (2016)	text compression	Text stemming compresses the vocabulary by reducing conflicted words to their common root form
17	Dahab et al. (2015), Singh and Gupta (2016)	Question answering (QA)	A variant form of questioning words stems that enhances the performance of the QA system
18	Dahab et al. (2015), Singh and Gupta (2016)	Text summarization (TS)	Variant forms of words with different meanings can be reduced to their common root. It makes easy for TS system to perform better
19	Fattah et al. (2006)	Machine translation (MT)	It is the variant form of words which are not present in the tagset language. Subsequently, in such cases, stemmer provides the stem that is helpful for translation
20	Rashid and Mohamad (2016)	Detecting wicked website	In wicked information filtering and detecting wicked website, text stemming is used to extract features that ultimatly improve the performance of the system

Table 5 Summary of state-of-the-art stemmer's evaluation metrics

No	Work	Stemming method	Languages	Dataset size	Evaluation methods
1	Mishra and Prakash (2012)	Rule-based	Hindi	2265 words	Manual
2	Ababneh et al. (2012)	Rule-based	Arabic	Sample terms list	Manual
3	Karaa (2013)	Rule-based	English	30,000 words	Paice (1994)'s evaluation method
4	Thangarasu and Manavalan (2013)	Statistics (cluster analysis)	Tamil	7,000 words	Manual
5	Husain et al. (2013)	Statistics (N-gram)	Urdu and Marathi	1,200 Urdu words 1,200 Marathi words	Manual
6	Sulaiman et al. (2014)	Rule-based	Malay	1,200 words	Paice (1994)'s Evaluation Method
7	Abu-Errub et al. (2014)	Rule-based	Arabic	1100 words	Manual
8	Al-Omari and Abuata (2014)	Rule-based (linguistic and mathematics rules)	Arabic	6,225 words	Manual
9	Rashidi and Lighvan (2014)	Hybrid	Persian	Small data from Hamshahri collection	Manual
10	Dianati et al. (2014)	Corpus base approach	Persian	1,250 Persian words	Manual
11	Al-Kabi et al. (2015)	Rule-based and pattern base	Arabic	6,081 words	Manual
12	Khan et al. (2015)	Rule-based and template matching	Urdu	66,200 words	Precision, recall and F-measure
13	Brychcín and Konopík (2015)	Statistical	Czech, Slovak, Polish, Hungarian, Spanish and English languages	Large date set for each language	Precision, recall and F-measure
14	Bimba et al. (2016)	Rule-based	Hausa language	1,723 words	Paice (1994)'s evaluation method Sirsat's evaluation method (Sirsat et al. 2013)
15	Momenipour and Keyvanpour (2016)	Statistical	Persian	PER-Tree-Bank words Bijankhan distinct words Hamshahri test collection	Manual, precision, recall
16	El-Defrawy et al. (2016)	Rule-based	Arabic	International Corpus of Arabic (ICA)	Precision, recall and F-measure

Table 5 (continued)

No	Work	Stemming method	Languages	Dataset size	Evaluation methods
17	Abainia et al. (2017)	Rule-based	Arabic	ARASTEM data set[a]	Paice (1994)'s evaluation methodology
18	Taghi-Zadeh et al. (2015)	Hybrid	Persian	4,689 words 26,913 words	Manual
19	Singh and Gupta (2017)	Statistical	English Marathi Hungarian Bengali	173,252 WSJ documents (English) 99,275 documents (Marathi) Magyar Hirlap collection of 49,530 documents (Hungarian) FIRE 2010 collection containing 123,047 documents (Bengali)	Precision, recall and F-measure
20	Mateen et al. (2017)	Hybrid	Punjabi	85,152 words	Manual
21	Jaafar et al. (2017)	Rule-based	Arabic	Quranic Arabic Corpus	Frakes and Fox (2003) Evaluation mechanism
22	Jabbar et al. (2018a, b)	Hybrid	Urdu	76,074 words	Precision, Recall and F-measure Frakes and Fox (2003)
23	Alotaibi and Gupta (2018)	Statistical	English Marathi Hungarian Bengali	WSJ documents (English) Sakal and Maharashtra Times (Marathi) Magyar Hirlap corpus (Hungarian) FIRE 2010 collection (Bengali)	Precision, Recall and F-measure
24	Suryani et al (2018)	Rule-based	Sundanese	4,453 words	Paice (1994)'s evaluation methodology
25	Saeed et al. (2018a, b)	Rule-based	Arabic	4007 documents	Precision, recall and F-measure
26	Ali et al. (2019)	Rule-based	Urdu	32,000 words	Manual

Table 5 (continued)

No	Work	Stemming method	Languages	Dataset size	Evaluation methods
27	Bölücü and Can (2019)	Statistical	Turkish, Hungarian, Finnish, Basque, and English	Turkish = 5,620 sentence and 53,798 tokens Hungarian = 24 K words Finnish = 19,000 sentences and 1,62,000 words Basque = 24 K words English = 24 K words	Frakes and Fox (2003) evaluation mechanism

[a] https://abainia.net

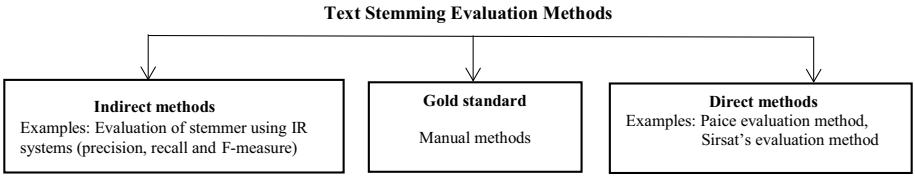

Fig. 3 Classification of text stemming evaluation methods

these values, a list of groups of semantically and morphologically related words are formed and then submitted to the stemmer. A stemmer commits under stemming error if it produces more than one unique stems for the same group or class of words. On the other hand if produced stem of one group also occurs in another group (same stem is produced for two groups of words) then the stemmer has committed over-stemming error. An ideal stemmer conflates this group to the same stem and has low UI and OI indexes. UI and OI can be calculated using Eqs. (1) and (2). Following four parameters are used to calculate over-stemming and under-stemming indexes.

Global Desired Merge Total (GDMT)
Global Desired Non-Merge Total (GDNT)
Global Unachieved Merge Total (GUMT)
Global Wrongly Merged Total (GWMT)

$$Under\ Stemming\ Index\ (UI) = \frac{GUMT}{GDMT} \tag{1}$$

$$Over\ Stemming\ Index(OI) = \frac{GWMT}{GDNT} \tag{2}$$

These parameters are defined in following section.

Under stemming index (UI) The Desire Merges Total (DMT) represents the total number of word forms in the group, and it can be calculated by the Eq. (3)

$$DMT_g = \frac{1}{2} n_g \left(n_g - 1\right) \tag{3}$$

where n_g = possible number of morphological forms in particular group having same stem

GDMT is equal to the sum of DMT values for all word groups as in Eq. (4)

$$GDMT = \sum_{i=1}^{N} DMT_g \tag{4}$$

Unachieved Merge Total (UMT) represents the failure of a stemmer to merge all query words to the same root in a specific group. Unachieved Merge Total (UMT), u represents the total number of distinct stems in a group that are produced by the stemmer, and it can be calculated as follows in (Eq. 5)

$$UMT_g = \tfrac{1}{2} \sum_{i=1}^{s} u_i \left(n_g - u_i\right) \tag{5}$$

where s = Number of distinct stems produced by stemmer (a stemmer may produce multiple stems for a particular group of words), n_g = possible number of morphological forms in particular group having same stem, u_i = ith stem produced by stemmer

GUMT can be calculated as sum of UMTs for each group using Eq. (6)

$$GUMT = \sum_{i=1}^{G} UMT_g \tag{6}$$

where G = Total number of groups in given corpus

Finally, the under-stemming index (UI) can be defined as given in (Eq. 1):

Over steaming index (OI) Wrongly Merged Total (WMT) represents the over stemming when words from two different groups are stemmed to one root. It can be calculated using (Eq. 7):

$$WMT_g = \frac{1}{2} \sum_{i,j=1}^{N} v_{ij}\left(n_i - v_{ij}\right) = \left(v_{11}, v_{12}, v_{13}, v_{21}, v_{22}, v_{23}, v_{31}, v_{32}, v_{33}\right) \tag{7}$$

where N = total number of groups involved in correct and wrong stemming, n_i = number of words in ith group, v_{ij} = Number of stems that should actually belong to group i but produced in group j.

In Eq. (7), we can see that v_{ij} is the number of stems that belong to group i and stemmer has produced in group j. If $i = j$, the stemmer has performed the job correctly wheras in case $i \neq j$, wrong stemming has been performed. In short, it tells that the stemming process for a particular group of morpholical variants is interfering with other groups.

Global Wrongly Merged Total (GWMT) is obtained by summing the WMT for all the groups by Eq. (8).

$$GWMT = \sum_{i=1}^{G} WMT_g \tag{8}$$

Desired Non-Merge Total (DNT) refers to the number of words in a certain group that can be conflated after stemming with words from some other semantic group. DNT can be calculated by the Eq. (9)

$$DNT_g = \frac{1}{2} n_g\left(w - n_g\right) \tag{9}$$

where w = Total number of words in the test dataset, n_g = Total number of words in particular group

The Global Desired Non-Merge (GDNT) is equal to the sum of DNT for all the groups and can be calculated by Eq. (10)

$$GDNT = \sum_{i=1}^{G} DNT_g \tag{10}$$

Hence, the Over-Stemming Index (OI) can be calculated by Eq. (2)

Stemming weight (SW) The SW parameter measures the strength of a stemmer. Lower value of SW identifies weak stemming whereas higher value indicates the strong stemming. It can be calculated using Eq. (11).

Table 6 Results of Paice evaluation methods

Stemmers	UI	OI	SW
Lovins	3.26×10^{-1}	16.3×10^{-5}	1.93×10^{-4}
Paice/husk	1.21×10^{-1}	1.18×10^{-4}	9.78×10^{-4}
Porter	3.74×10^{-1}	2.18×10^{-5}	7.4×10^{-5}

$$SW = \frac{OI}{UI} \tag{11}$$

Paice (1994) utilized the ClSI source (CISI Collection, University of Glasgow) which contain 184,659 words and the authors extracted two smaller word samples of size 1,527 distinct words. The author experimented with Lovin's (1968), Porter's (1980) and Paice/Husk's (1990) stemmers and concluded that Paice/Husk (Paice 1990) stemmer has the highest value of OI index; on the contrary, rest of the two stemmers show the lowest score. In the case of UI index, Porter's stemmer has the highest under-stemming errors than others. Paice/Husk (Paice 1990) has UI lowest score $\left[1.21 \times 10^{-1}\right]$ and Porter has highest UI $\left[3.74 \times 10^{-1}\right]$ whereas porter has lowest OI $\left[2.18 \times 10^{-5}\right]$ and Paice/Husk (Paice 1990) has highest OI $\left[1.18 \times 10^{-4}\right]$ as mentioned in Table 6. According to SW score $\left[9.78 \times 10^{-4}\right]$, Paice/Husk (Paice 1990) is the strongest stemmer, Lovin's (1968)' stemmer scored SW is $\left[1.93 \times 10^{-4}\right]$ and is at second place and Porter's (1980)'s stemmer with SW $\left[7.4 \times 10^{-5}\right]$ stands last as shown in Table 6.

Discussion Paice's Evaluation Methodology (PEM) has some problems. First, it is not trivial to create groups of morphologically related words, and if a group contains only one word then the value of DMT will be zero. Moreover, it is time-consuming because the manual check to find whether a resultant word is suffering from under-stemming or over-stemming. This methodology is not suitable to check a large volume of data set. It deals only with two types of stemming errors; however, a stemmer may commit some other errors such as generation of invalid words i.e. generated stem does not lie in any group. In some cases, the stemmer produces linguistically correct stem but incorrect in reality, as ولدين [two boys] by Khoja (1999) stemmer root لدن [soft], is derived, it is linguistically correct but not valid stem, ولد [give birth] is the correct stem (Nwesri and Alyagoubi 2015). And finally, this method is suitable only for the English language (AlSerhan and Alqrainy 2008).

4.1.2 Hull's evaluation method

In the situations where the performance of two stemmers is slightly different from each other. it is very hard to say that the performance variation is enough or not, or it just happens by chance. For such cases, Hull (1996) proposed Analysis Of Variance (ANOVA) model for large, continuous and normally distributed sample size. It is observed that for English language, word inflectional forms are low, and the observed differences are limited. Hull (1996) performed the experiments over five stemmers [Remove s stemmer,[1] Lovins (1968),[2] Porter (1980), Inflec and Deriv stemmers (Xer 1994)] using the SMART text retrieval system originated at Cornell University (Buckley 1985). No stemming is used

[1] Built in SMART system.

[2] Extensively modified version Lovens (1968) included in SMART system.

Table 7 ICF values of Lovins and Porter stemmers

Stemmer	Lennon et al. (1988)	Frakes and Fox (2003)	Paice, (1994)	Harman (1991)
Lovin's (1968)	From 30.9–45.8%	29%	44.60%	38.23%
Porter's (1980)	From 26.2–38.8%	17%	38.90%	28.74%

in order to index the queries and compare them with their standard form. Hull (1996) concluded the average absolute improvement is smaller (up to 1–3%) in IR system only due to stemming.

4.1.3 Frakes's evaluation

The strength of stemmer indicates the degree of variation of the derived stem. Weak/light stemmers conflate only highly related words such as "consist", "consisted", and "consisting". In contrast, strong/heavy stemmers can handle more variation in morphological forms such as "consistency", "consistent", "consistently". Frakes and Fox (2003) proposed a criterion for determining the stemmer strength and similarity as described below.

The Mean number of words per conflation class (MWC) MWC refers to the mean number of words conflated per class. For example, "consist", "consisted", "'consisting" are conflated to "consist" which determine the value of MWC. In this case, it is three words. Higher value of MWC signifies the better performance of a stemmer. It is calculated using Eq. (12).

$$MWC = \frac{N}{S} \tag{12}$$

where N refers to the total number of unique words in a class and S refers to the number of unique stems obtained.

Index compression factor (ICF) A higher ICF value signifies the strength of the stemmer (Frakes and Fox 2003). Many experiments have proven that strong stemmers yield higher ICF value. Lennon et al. (1981) achieved ICF [30.9–45.8%] on Lovin's (1968)' stemmer and 26.2–38.8% using Porter's (1980)'s stemmer. Frakes and Fox (2003) depicted higher 29% ICF and 17% on porter's (1980) stemmer. Paice, (1994) and Harman (1991) proved Loven's (1968) has higher ICF 44.60% and 38.38 respectivly than porter's (1980) stemmer as shown in Table 7 last two columns.

The index compression factor is defined in Eq. (13)

$$ICF = \frac{n-s}{n} \tag{13}$$

where, n = the number of words in the corpus, s = the number of stems

For example, a corpus with 50,000 words (n) and 40,000 stems (s) would have an index compression factor of 20%.

The word change factor (WCF) WCF indicates the number of words that are left unchanged by the stemmer. For example, a stemmer might not alter the word "consist" as it is already a stem. Strong stemmers may often change such words than weaker stemmers. Normally, the higher value of WCF indicates the best stemmer. WCF can be calculated in (Eq. 14)

$$WCF = \frac{N-C}{N} \tag{14}$$

where N parameter is the number of unique words and C is the number of unchanged words.

The mean number of characters removed (MCR) It represents the average number of characters (in a group) removed to derive the stem. The strong stemmer truncates more letters than the weak stemmer to obtain the stem. As an example, for the word "helps", one character 's' is removed, for "helper" two letters 'er' are removed, in case of the word "helpful" three letters 'ful' are removed, similarly for "helpless" word four letters 'less' are stripped to extract the stem "help". Equation Eq. (15) computes the MCR score.

$$\begin{aligned} MCR &= \frac{Total\ no.\ of\ letters\ removed}{Total\ no.\ of\ words} \\ MCR &= \frac{(1+2+3+4)}{4} = 2.5 \end{aligned} \tag{15}$$

Frakes and Fox (2003) used the Moby Common Dictionary wordlist[3] to evaluate four stemmers ["S" stemmer (Harman 1991), Lovins (1968), Porter (1980), Paice/Husk(1990)] and claimed that Paice and Lovins stemming algorithms are the most similar, while the Paice and "S" stemmers are the most dissimilar.

MCR only measures the strength of a stemmer and gives metrics to check the similarity of two stemmers; however, it does not deal with the accuracy of the extracted stem. It also does not measure the correctly stemmed words. It does not provide information about invalid or modified stem production.

MCR does not measure the transformations of the stem and CI only check the compressions ratio of vocabulary size, not the correctness of the stem. Moreover, it is difficult to identify all the conflation classes and checking corresponding stem words manually for every conflation class is also a tedious task. Because some languages have high conflation and derivation morphology like Hungarian or Hebrew languages which have thousands of variant forms from a single word (Krovetz 2000). Consequently, its vocabulary Compression Index (CI) will be high. For instance, famous English stemmer Porter (1980) claimed to reduce initial vocabulary by one third and Jabbar et al. (2018a, b) proposed Urdu stemmer that reduced vocabulary size by 55%.

4.1.4 Sirsat's evaluation method

Sirsat et al. (2013) criterion is very compelling for assessing the strength and accuracy of a stemming algorithm. The following parameters are used to evaluate the strength and accuracy of the stemmer.

Word stemmed factor (WSF) It refers to the average number of words stemmed from the stemmer. The threshold value is the minimum (50%). The larger value of WSF signifies the strength of the stemmer. It can be calculated by Eq. (16)

$$WSF = \frac{WS}{TW} \times 100 \tag{16}$$

[3] https://antiflux.org/dictionary?dict=moby-thesaurus

Table 8 Results of Sirsat's evaluation method

Stemmers	ICF	WSF	CSWF	AWCF
Paice/Husk	**64.63**	70.99	28.73	**19.26**
lovins	56.52	**73.35**	27.80	– 24.8
Porter1	51.88	67.17	31.97	– 8.52
Porter2	53.72	66.58	**34.76**	8.6

where WS = No. of stem words, TW = Total number of words in a sample

Correctly stemmed words factor (CSWF) It indicates the mean number of words correctly stemmed by the stemmer. The higher percentage of CSWF indicates the higher strength and accuracy of the stemmer. Minimum threshold value of CSWF is 50% and it can be calculated by the following Eq. (17)

$$CSWF = \frac{CSW}{SW} \times 100 \tag{17}$$

where CSW = Number of correctly stemmed words, WS = Total number of stemmed words.

Average words conflation factor (AWCF) This refers to the mean value of variant words of a different conflation group/s that are correctly stemmed. To calculate AWCF, we must compute the number of distinct words after conflation, which is calculated by Eq. (18):

$$NWC = S - CW \tag{18}$$

where S = Number of distinct stems after stemming, CW = Number of correct words which are not stemmed

Finally, AWCF is obtained by Eq. (19)

$$AWCF = \frac{CSW - NWC}{CSW} \times 100 \tag{19}$$

The higher value of AWCF indicates the higher strength and accuracy of the stemmer. Sirsat et al. (2013) carried out the experiments over four stemmers [lovins (1968), porter 1(1980), porter 2 (2006), Paice/Husk(1990)] and concluded that the Paice/Husk stemmer is slightly better than other stemmers in terms of ICF [64.63] and AWCF [19.26]. Lovins (1968)'s stemmer has higher WSF 73.35 and porter 2 better with CSWF 34.76 as shown in Table 8.

The value of AWCF may be zero or negative when the number of incorrect stems is larger than the correctly stemmed words.

4.1.5 Jaafar évaluations mechanism

Jaafar et al. (2017) used the execution time and accuracy of stemmer to determine the performance of a stemmer using formula given in Eq. (20)

$$GS_{score} = \frac{\alpha . \sum T_w}{\beta . \sum Acc_w} \tag{20}$$

where, GS_{score} = Global Stemming Score, T_w = Execution time to get a stem for a word, Acc_w = Correctness of stem word w, α, β = Variables to give the weights of time taken by

Table 9 Stemmer's performance in the IR system (Karaa 2013)

Used stemmer in IR	Precision	Recall
Without stemming	0.661	0.671
Original porter stemmer	0.732	0.775
New porter stemmer	0.852	0.884

the stemmer to extract the stem and accuracy. These values reflect which is more important, accuracy or execution time, if accuracy is matter, then the value of α is set higher than β.

T_w and Acc_w are two different measurements. T_w is measured in the form of $n^{1,2,\cdots}$ and Acc_w can be from 1 to 100. The relation between accuracy and execution time is always inverse to each other. Ridiculus results may be obtained if weights are not properly assigned.

4.2 Gold standard assessments

In this evaluation approach, the correctness of a system is manually checked by experts. For this purpose, input is given, and its corresponding output is checked manually. Many statistical and rule-based stemmers are evaluated manually such as Ali et al. (2019) Urdu stemmer, Al-Kabi et al. (2015) Arabic stemmer and Persian stemmer (Taghi-Zadeh et al. 2015). The accuracy of the stemmer by gold standard assessment can be calculated as given in Eq. (21)

$$Accuracy = \frac{Total\ No.\ of\ correct\ stem\ obtained}{Total\ No.\ of\ words\ given\ to\ stemmer} \times 100 \tag{21}$$

This method is good for small sized dataset however it is not suitable for large-scale evaluation. This method reflects the ratio of correct stems produced by a stemmer, but is silent about already stem words given to a stemmer. TP and TN both are important to determine the performance of a stemmer.

4.3 Indirect evaluation

Stemming reduces the dimensionality of the text data and improves the performance of information retrieval (IR) system (de Oliveira and Junior 2018). The performance of a stemmer can also be evaluated in the context of specific NLP applications such as Alotaibi and Vishal Gupta (2018) evaluated their proposed stemmer in an IR system. Ali et al. (2018) tested their stemmer for text classification and Boukhalfa et al. (2018) proposed Arabic stemmer to improve the performance of Arabic plagiarism detection system. Many researchers compared the performance of various stemming algorithms for IR system such as Flores and Moreira (2016) to evaluate on Portuguese, Spanish, French, and English language stemmers in IR experiments and concluded 70% of the query topics improvement in AP (average precision). Karaa (2013) modified the Porter (1980) stemmer and claimed that new porter stemmer improves the IR system by 0.852 precision and 0.884 recall. In contrast without stemming precision is 0.661 and recall is

0.671 and with orginal porter (1980) stemmers precision 0.732 and recal 0.775 shown in Table 9.

The recall, precision, and F1-measure (Jabbar et al. 2018a, b) are standard measures to assess the performance of stemmer in an IR system.

Recall Recall indicates the ratio between stem words extracted by stemmer and total possible stem words as mentioned in Eq. (22).

$$Recall = \frac{Total\ Correct\ Stem\ Produce\ By\ System}{Total\ Possible\ Correct\ Stem} \tag{22}$$

Precision Precision is a ratio of total correct stems and total produced stems. It is calculated by Eq. (23)

$$Precision = \frac{Total\ Correct\ Stem\ Got\ By\ System}{Total\ Stem\ Produce\ By\ System} \tag{23}$$

Weighted F1-measure A variant of F1-measure allows weighting emphasis on precision over recall. It is calculated by (Eq. 24).

$$F1_{weighted} = \frac{(\beta^2+1)(precision \times recall)}{\beta^2(recall)+precision} \tag{24}$$

where B = Weighting between precision and recall typically $\beta = 1$.

A weighted combination of recall and precision In addition to the standard precision/recall measures, several other methods are also adopted by the researchers such as Lennon et al. (1998) who have used a weighted combination of recall and precision as given in Eq. (25).

$$E = 1 - \frac{(1+b^2)PR}{b^2P+R} \tag{25}$$

where E = an effective function, lower value of E indicates better performance, P = Precision, R = Recall, b = measures the relative importance attached to PR

AP and Mean Average Precision (MAP) are also used to evaluate the impact of stemming in an IR system. AP refers to the mean of precision and recall; whereas, the MAP represents the average of AvPs when more than one query is used (Flores and Moreira 2016).

TERRIER (Qunis et al. 2006) is an open source IR system that is a highly flexible, efficient and comprehensive platform for carrying out stemming experiments. TERRIER (Qunis et al. 2006) is developed by the School of Computing Science, the University of Glasgow in JAVA programming language and is available at https://terrier.org/. It supports UTF (Unicode Transformation Format) text hance corpora of many languages can be used. It uses Porter stemmer (1980) by default. Many other IR systems are also available such as Lemur/Indri ("Lemur" 2016) Lucene/Solr ("Lucene" 2018), Xapian ("Xapian" 2018). All IR systems can be used to perform basic IR tasks. However, TERRIER has some deficiencies that may include:

- It is difficult to check how many words are relevant in the corpus.
- It is hard to choose a stemmer for search engine because every search engine has a different database.
- The results are not reliable because every stemmer is evaluated on different datasets.

Table 10 Comparison of Paice method with the manual method

Stemmer	Paice's evaluation			Gold standard method
	UI	OI	SW	Accuracy (%)
ETS stemmer (Al-Shammari and Lin 2008)	0	0	**0**	**100**
Khoja stemmer (Khoja and Garside 1999)	0	0.0755	**0**	70

Table 11 Results of the test using the sample I

Stemmers	References	Accuracy (%)	SW
STEMBERS	Alvares et al. (2005)	**62.20**	1.60×10–3
STEMP	Orengo and Huyck (2001)	55.30	1.44×10–3
PORTER	Porter (1980)	43.80	0.67×10–3

Table 12 Results of the test using sample II

Stemmers	References	Accuracy (%)	SW
STEMBERS	Alvares et al. (2005)	**69.02**	3.50×10–4
STEMP	Orengo and Huyck (2001)	67.60	3.30×10–4
PORTER	Porter (1980)	57.86	1.25×10–4

There is no mechanism defined to find number of produced stem words and actual stem words because the platform only tells whether a word is stemmed or not. Moreover, it does not tell the degree of the correctness of the stem.

5 Analysis of evaluation methods

Every evaluation method measures the specific features of the stemmer and ignores the rest. For example, Al-Shammari and Lin (2008) assessed the performance of proposed Arabic stemmer named Educated Text Stemmer (ETS) using Paice (1994) evaluation method and claimed that ETS stemmer (Al-Shammari and Lin 2008) performed better than Khoja and Garside (1999)'s stemmer. However, they obtained "0" value of stemming weight (SW) for both above-mentioned stemmers which show that both stemmers are equal in strength and performance. Whereas, according to the gold standard evaluation method ETS stemmer's accuracy is 100% that is better than Khoja and Garside (1999) which achieve 70% stemmer's accuracy, as shown in Table 10. The authors randomly choose two samples of Arabic documents. First sample consists of 47 medical documents that contained 9,435 Arabic words, and the second sample comprises 10 long, Arabic sports articles from CNN.com with total 7,071 words. (Al-Shammari and Lin 2008)

Paice (1994) evaluation method, in some experiments, shows the contrary results from the gold standard evaluation method as shown in Table 9. This controversy in results can also be seen in Brazilian Portuguese languages (Alvares et al. 2005), in

Table 13 Experiment's result of Paice evaluation method

Author	Stemmer	Accuracy (%)	OI	UI	SW
AlSerhan et al. (2008)	Stem1	100	0	0	**0**
	Stem2	100	0	0	**0**
	Stem1	0	0	0	**0**
	Stem2	0	0	0	**0**
	Stem1	28.57	1	0	**0**
	Stem2	85.71	0.2	0.36	**0.56**

Table 14 Comparison of manual and Sirsat's method

Stemmer	Sirsat's method			Manual
	WSF	CSWF	AWCF	Accuracy
HStemV1	**69.47**	81.12	52.83	56.35
HStemV2	65.7	**87.37**	**59.45**	**57.39**

Table 15 Comparison of Gold standard, Frakes and Sirsat's evaluation parameters

Stemmer	Frakes evaluation metrics				Sirsat's evaluation method		Gold standard method
	ICF	MWC	WCF	MCR	WSF	CSWF	Accuracy (%)
Light10	51.08	2.04	80.86	1.57	80.84	32.2	14.96
Motaz	36.04	1.56	54.31	0.64	54.31	57.75	18.59
Tashaphyne	**69.94**	**3.32**	**87.23**	**2.02**	87.23	22.63	10.95
SAFAR-stemmer	48.22	1.93	62.55	1.11	**62.55**	**80.61**	**33.70**

which results produced by Paice (1994) are totally opposite to the accuracy measured by Gold standard/manual methods. Similarly, STEMBERS stemmer's performance is 62.20% that is better than the counter stemmer STEMP [55.30%] and PORTER [43.80%], but the SW value 1.60×10^{-3} of STEMBERS shows the lowest performance among evaluated stemmers as mentioned in Table 11. By observing Table 12, similar performance is reflected in sample II experimental data, in which STEMBERS stemmer achieved an accuracy of 69.02% that is higher than its counterpart stemmer i.e. STEMP, PORTER. However, SW is 3.50×10^{-4}, which is equal to STEMP stemmer but higher than the PORTER [1.25×10^{-4}] stemmer. Sample I and sample II comprosis 102 and 2,696 semantic groups constructed manually.

This trend in Paice evaluation method has also been proved by AlSerhan and Alqrainy (2008), where, they compared the results obtained through manual method and Paice evaluation method for Arabic language using two virtual stemmer's results (AlSerhan and Alqrainy 2008). It is depicted from the obtained results that Paice evaluation method denies the gold standard results. As stated before, the 0 value of SW shows the strongest stemmer, and 0 value is obtained when OI, or UI or both are zeros as mentioned in Table 12. When gold standard accuracy is 100% or 0% as shown in Table 13(2nd column), in both cases, the value of SW is zero that is far from reality.

Table 16 Experimental data for Paice evaluation

Groups	Input words	Actual stem	VS1	VS2
G1	بدنساز [Bdnsaz/Bodybuilder]	بدن [Bdan/body]	بدن [bdan/body]	بد *[Bd/*bad*]*
	بدنسازی [Bdnsazi/Bodybuilding]	بدن [Bdan/body]	بدن [bdan /body]	بد *[Bd/*bad*]*
	بدنی [Bdni/bodily]	بدن [Bdan/body]	بدن [bdan /body]	بد *[Bd/*bad]
	ابدان [Abdan/bodies]	بدن [Bdan/body]	بدن [bdan /body]	بد*[Bd/*bad*]*
	بدنوں [Abdano/bodies]	بدن [Bdan/body]	بد [bd/bad]	بد*[Bd/*bad*]*
G2	بدپن [Bdpan/badness]	بد [Bd/bad]	بدن [bdan/body]	بد [Bd/bad]
	بدنیہ [Bdniah/bad luck]	بد [Bd/bad]	بد [Bd/bad]	بد [Bd/bad]
G3	بد روح[Bad roh/Evil spirit]	روح[Roh/soul]	روح [Roh/soul]	بد [Bd/bad]
	بد روحیں[Bad rohain/Evil spirits]	روح[Roh/soul]	روح [Roh/soul]	بد [Bd/bad]

But when accuracy is 85.71% obtained by gold standard method, its corresponding SW value is 0.56, and the accuracy of its competitors is 28.57 that is lower, but SW value is 0 that means Stem2 is a stronger stemmer than Stem1 as reflected from Table 13.

Some of Sirsat's parameter's values (Sirsat et al. 2013) showed a tendency in gold standard's results as shown in Table 14. Where, HStemV2 showed better performance in terms of accuracy as shown in Table 14. On the other hand, with respect to Sirsat's parameter (Sirsat et al. 2013) WSF, HStemV1 performs better as shown in Table 14.

Frakes evaluation method also talks about how many words are changed when stemmer ignored the correctness. Jabbar et al. (2016) used the Quranic Arabic Corpus (Dukes and Habash 2010) which contains 18,350 unique Arabic words to compare the results with counterpart stemmers Light10 (Larkey et al. 2007), Motaz (Saad and Ashour 2010), and Tashaphyne (Zerrouki 2016). SAFAR-Stemmer (Jaafar et al. 2016) provided results statistics about their stemmer. Form these calculations, we compute the Frakes evaluation (Frakes and Fox 2003) parameters (as shown in Table 14). Sirsat's parameters (Sirsat et al. 2013) and manual accuracy are also calculated and mentioned in Table 15. SAFAR-Stemmer (Jaafar et al. 2017) achieved 33.70 accuracy that is higher than its competitor Light10 (Larkey et al. 2007) [14.96], Motaz (Saad and Ashour 2010) [18.59] and Tashaphyne (Zerrouki 2016) [10.95]. Sirsat's evaluation parameter WSF is 62.55 and CSWF is 80.61 which is highest among the stemmers. Sirsat's evaluation and gold standard method show that the SAFAR-Stemmer is better than Light10 (Larkey et al. 2007), Motaz (Saad and Ashour 2010), and Tashaphyne (Zerrouki 2016) stemmers. However, Frakes evaluation metrics (Frakes and Fox 2003) deny this result as Tashaphyne (Zerrouki 2016) performed better with respect to the higher values of ICF [69.94], MWC [3.32], WCF [87.23], and MCR [2.02] as shown in Table 14.

Considering the observation mentioned in Table 15 the performance graph of mentioned stemmers varied with respect to evaluation methods. So, to be more accurate, we develop a virtual scenario for the scarce resourced Urdu language stemming (see Table 16) and experimented two virtual stemmers i.e. VS1 and VS2.

Virtual stemmer1 Paice evaluation

Unachieved Merge Total (UMT) derived using (Eq. 5).

Table 17 Calculation of UMT, DMT, WMT, DNT

Stemmer	G	UMT	DMT	WMT	DNT
VS1	G1	4	10	4	10
	G2	1	1`	1	7
	G3	0	1	0	7
VS2	G1	0	10	0	10
	G2	0	1	24	7
	G3	0	1	0	7

Table 18 Calculation of GUMT, GDMT, GWMT, GDNT

Stemmer	GUMT	GDMT	GWMT	GDNT
VS1	5	12	5	24
VS2	0	12	24	24

$$UMT_g = \frac{1}{2}\sum_{i=1}^{s} u_i\left(n_g - u_i\right) \qquad (5)$$

Then for a particular group $g1$,

s = Number of distinct stems produced by stemmer (a stemmer may produce multiple stems for a particular group of words).

n_{g1} = possible number of morphological forms in particular group having same stem. 5 The total number of the stems بدن [Bdan/body] and بد [Bd/bad] in group 1.

u_i = ith stem produced by stemmer. For $u_1, 4$ stems of word بدن [Bdan/body] in group 1 and for u_2, 1 stem بد [Bd/bad] in group 1 is produced.

$$UMT_{g1} = \frac{1}{2}[4 \times (5-4) + 1 \times (5-1)]$$
$$UMT_{G1} = 4$$

We calculate the UMT [using Eq. (5)] for both stemmers and for each group that is shown in Table 16. The UMT is the only one that valeted the conflation regardless of correctness (as mentioned in Table 16). The value of UMT is 0 if all words in the group are conflated (correctly or incorrectly). This causes the inverse result of the gold strand evaluation results. UsingEq. (3), we can calculate:

$$DMT_g = \frac{1}{2}n_g\left(n_g - 1\right) \ (3)$$
$$DMT_{g1} = \frac{1}{2}[5 \times (5-1)]$$
$$DMT_{g1} = 10$$

The DMT values of both stemmers against each group are mentioned in Table 16.

$$WMT_G = \frac{1}{2}\sum_{i=1}^{t} v_i\left(n_s - v_i\right)$$

Table 19 Experimental results to check the viability of various stemming evaluation methods

Stemmer	Paice (1994, 1996)'s evaluation method				Frakes and Fox (2003)'s Evaluation Metrics				Sirsat et al. (2013)'s evaluation method			Gold standard evaluation
	UI	OI		SW	MWC	ICF	WCF	MCR	WSF	CSWF	AWCF	Accuracy (%)
VS1	0.42		0.21	0.5	3	66.67	**1**	2.3	**100**	**77.8**	**57**	**77.8**
VS2	0		1	**0**	**9**	**88.89**	**1**	**3.3**	**100**	22.2	50	22.2

$n_s = 5$. The total number of the stems بدن [Bdan/body] in group 1 and group 2, $v_1 = 4$ The number of stem بدن [Bdan/body] in group 1, $v_2 = 1$ The number of stem بدن [Bdan/body] in group 2

$$WMT_{G1} = \frac{1}{2}[4 \times (5-4) + 1 \times (5-1)]$$
$$WMT_{G1} = 4$$

The values of WMT (Eq. 7) for each stemmer against each group are represented in Table 17.

$$DNT_g = \frac{1}{2} n_g (w - n_g)$$

For first group

$$w = 9 \quad n_{g1} = 5$$
$$DNT_{g1} = \frac{1}{2}[5 \times (9-5)]$$
$$DNT_{g1} = 10$$

The remaining values of DNT_g are calculated and are provided in Table 17.

The values of GUMT (Eq. 6), GDMT (Eq. 7), GWMT (Eq. 8), and GDNT (Eq. 10) are calculated from Table 17 and are given in Table 18.

From Table 19, it is observed that the accuracy obtained by VS1 is comparatively higher [77.8] than VS2 stemmer [22.2].

It is clear from Table 18 that the stemming weight SW shows that the VS1 [0.5] is a light stemmer and VS2 [0] is a heavy stemmer. These results indicate that the stemmer SV2 is a perfect stemmer that is quite opposite of manual evaluation results.

The MWC value is 3 for VS1 and VS2 is 9 that is higher. It is observed from Table 17 that index ICF obtained by VS2 [88.89] is higher than the VS1 [66.67]. The WCF is 1 for both stemmers, but, MCR is higher for VS2 [3.3] than VS1 [2.3]. Frakes proposed parameters to assess the performance of the stemmer which gives conflicting results as that of the gold standard method.

The WSF obtained by both stemmers is 100% that is above a threshold value [50%]. This indicates that the strength of both the stemmers is better and aggressive in nature. The AWCF value of stemmer VS1 is 57 and 50 for VS2 that shows VS1 is stronger and more aggressive than the stemmer VS2. However, there is a comparatively large difference between both the stemmers with respect to CSWF, VS1 is 77.8 and VS2 is 22.28 that show the same tendency as a gold standard evaluation method reflects. But two other parameters give contrast results from the manual assessment.

6 Challenges and future directions

The text stemming is well studied and still open area of work especially for Arabic script based langauges. Text Stemming has been recognized as an excellent pre-processing tool in many NLP applications. Several approaches are proposed by the researchers facing various issues and challenges. However, how to evaluate a stemmer is an open question. In this

section, we present different issues and challenges to evaluate a stemmer that must be considered by the researchers in this domain.

Language type Concatenative and non-concatenative are two categories of languages with respect to the morphological process. For a concatenative language, the affix and stem are traced in a linear fashion but in non-concatenative language, the stem and affix are intervening (Kastner 2019). As the stemming approaches are language dependent, the stemmer evaluation methods are also changed accordingly. For instance, in the case of Semitic language, Paice (1994, 1996) evaluation method is not suitable (AlSerhan and Alqrainy 2008).

Types of affixes A variety of affixes is used in various languages such as the Indonesian language possess prefix, suffix, confixes, and suffix (Setiawan et al. 2016). State of the art stemming evaluation methods do not tell us about the types of affixes that are handled and the types of affixes that are not considered.

Text stemming approaches Most of the stemmers presented in the literature are Linguistic knowledge-based, that handle more than one affix type. On the other hand, statistical stemmers only remove the suffix. With the development of high end AI methods like neural networks, development of stemmers may produce superior results.

Domain-specific Some researchers evaluate the stemmer in a particular NLP application. Every application only considers the specific feature of the language. For instance, English IR system, Porter (1980) claimed that only suffix removal is enough; however, it is not suitable for Semitic language like Arabic.

Besides above all, certain issues like computation complexity, space complexity, linguistic correctness and considered types of affixes are helpful to determine the performance of a stemmer. Hence, the metrics to evaluate a stemmer must address all issues mentioned above.

7 Conclusions

The purpose of stemming algorithms is very simple that is to extract the same stem from various conflation forms of words. Various researchers proposed different parameters to measure the performance of text stemming algorithms. However, each criterion only measures specific aspects of stemmer performance. Different text stemming evaluation (TSE) methods proved to be useful in case of specific NLP applications.

In the developed NLP systems that use stemming, there is no standard TSE-method which can provide a landmark to measure the performance of the stemming algorithms. A stemmer performance may increase or decrease if different evaluation methods are used. Therefore, there is a certain need to develop a standard evaluation method. The reason for such result lies in the type of experimental data, training data, size of data and construction of stemming rules (if rule based approach is used).

A varity of features such as affix types, language types, data sets types and size can be used to develop a robust stemmer's evaluation mechanism that consider the conflation ratio as well as linguistically correctness of the stem. We conclude that this article provides a comprehensive review of the state-of-the-art text stemming evaluation methods, their challenges and the avenues for future work.

Funding Funding was provided by Bahauddin Zakariya University (PK) (Grant No: 2019-05).

References

Ababneh M, Al-Shalabi R, Kanaan G, Al-Nobani A (2012) Building an effective rule-based light stemmer for arabic language to improve search effectiveness. Int Arab J Inf Technol 9(4):368–372

Abainia K, Ouamour S, Sayoud H (2017) A novel robust Arabic light stemmer. J Exp Theor Artif Intell 29(3):557–573

Abu-Errub A, Odeh A, Shambour Q, Hassan OAH (2014) Arabic roots extraction using morphological analysis. Int J Comput Sci Issues (IJCSI) 11(2):128

Ali M, Khalid S, Aslam MH (2018) Pattern-based comprehensive Urdu stemmer and short text classification. IEEE Access 6:7374–7389

Ali M, Khalid S, Saleemi M (2019) Comprehensive stemmer for morphologically rich urdu language. Int Arab J Inf Technol 16(1):138–147

Alotaibi FS, Gupta V (2018) A cognitive inspired unsupervised language-independent text stemmer for Information retrieval. Cognit Syst Res 52:291–300

Al-Kabi MN, Kazakzeh SA, Ata BMA, Al-Rababah SA, Alsmadi IM (2015) A novel root based Arabic stemmer. J King Saud Univ-Comput Inf Sci 27(2):94–103

Al-Omari A, Abuata B (2014) Arabic light stemmer (ARS). J Eng Sci Technol 9(6):702–717

AlSerhan HM, Alqrainy S, Ayesh A (2008, November). Is paice method suitable for evaluating Arabic stemming algorithms? In: International conference on computer engineering & systems, 2008 (ICCES 2008). IEEE, pp 131–135

Al-Shammari ET, Lin J. (2008, October). Towards an error-free Arabic stemming. In Proceedings of the 2nd ACM workshop on Improving non English web searching. ACM, pp 9–16

Al-Sughaiyer IA, Al-Kharashi IA (2004) Arabic morphological analysis techniques: A comprehensive survey. J American Soc Inf Sci Tech 55(3):189–213

Alvares RV, Garcia AC, Ferraz I (2005) December) STEMBR: a stemming algorithm for the Brazilian Portuguese language. Portuguese conference on artificial intelligence. Springer, Berlin, pp 693–701

Aronoff M, Fudeman K (2011) What is morphology? vol. 8.Wiley, pp 2–3

Bimba A, Idris N, Khamis N, Noor NF (2016) Stemming Hausa text: using affix-stripping rules and reference look-up. Lang Resour Eval 50(3):687–703

Bölücü, Necva and Burcu Can. (2019). Unsupervised Joint PoS Tagging and Stemming for Agglutinative Languages. ACM Trans. Asian Low-Resour. Lang. Inf. Process. 18, 3, Article 25 (January 2019), 21 pages. https://doi.org/10.1145/3292398

Boudchiche M, Mazroui A (2015, December). Evaluation of the ambiguity caused by the absence of diacritical marks in Arabic texts: statistical study. In: 2015 5th international conference on information and communication technology and accessibility (ICTA). IEEE, pp 1–6

Boukhalfa I, Mostefai S, Chekkai N (2018, March) A study of graph based stemmer in Arabic extrinsic plagiarism detection. In: Proceedings of the 2nd mediterranean conference on pattern recognition and artificial intelligence. ACM, pp 27–32

Brychcín T, Konopík M (2015) HPS: high precision stemmer. Inf Process Manag 51(1):68–91

Buckley C (1985) Implementation of the smart information retrieval system. Technical report 85–686, Cornell University.

Cambria E, White B (2014) Jumping NLP curves: a review of natural language processing research. IEEE Comput Intell Mag 9(2):48–57

Chintala DR, Reddy EM (2013) An approach to enhance the CPI using Porter stemming algorithm. Int J Adv Res Comput Sci Softw Eng 3(7):1148–1156

CISI Collection https://ir.dcs.gla.ac.uk/resources/test_collections/cisi/. Accessed 30 Dec 2019. Developed by University of Glasgow

Dahab MY, Ibrahim A, Al-Mutawa R (2015) A comparative study on Arabic stemmers. Int J Comput Appl 125(8):38–47

Dang Q, Zhang J, Lu Y, Zhang K (2013) WordNet-based suffix tree clustering algorithm. In: International conference on information science and computer applications (ISCA 2013)

Dey A, Paul A, Purkayastha BS (2014) Named entity recognition for Nepali language: a semi hybrid approach. Int J Eng Innov Technol (IJEIT) 3:21–25

Dianati MH, Sadreddini MH, Hossein RA, Fakhrahmad SM, Taghi-Zadeh H (2014) Words stemming based on structural and semantic similarity. Comp Eng Appl J 3(2):89–99

de Oliveira RAN, Junior MC (2018) Experimental analysis of stemming on jurisprudential documents retrieval. Information 9(2):28

Dukes K, Habash N (2010) Morphological annotation of Quranic Arabic. In Lrec, pp 2530–2536

El-Defrawy M, El-Sonbaty Y, Belal NA (2016) A rule-based subject-correlated Arabic stemmer. Arab J Sci Eng 41(8):2883–2891

Fattah MA, Ren F, Kuroiwa S (2006) Stemming to improve translation lexicon creation form bitexts. Inf Process Manag 42(4):1003–1016
Flores FN, Moreira VP (2016) Assessing the impact of stemming accuracy on information retrieval–a multilingual perspective. Inf Process Manag 52(5):840–854
Frakes WB, Fox CJ (2003) Strength and similarity of affix removal stemming algorithms. In ACM SIGIR forum, vol 37, no 1. ACM, pp 26–30.
Gaidhane MS, Gondhale MD, Talole MP (2015) A comparative study of stemming algorithms for natural language processing. J Eng Educ Technol (ARDIJEET) 3(2):1–6
Giachanou A, Crestani F (2016) Like it or not: a survey of twitter sentiment analysis methods. ACM Comput Surv (CSUR) 49(2):28
Harman D (1991) How effective is suffixing. J Am Soc Inf Sci 42(1):7–15
Hassani K, Lee WS (2016) Visualizing natural language descriptions: a survey. ACM Comput Surv (CSUR) 49(1):17
Husain MS, Ahamad F, Khalid S (2013) A language independent approach to develop Urdu stemmer. Advances in computing and information technology. Springer, Berlin, pp 45–53
Hull DA (1996) Stemming algorithms—a case study for detailed evaluation. J Am Soc Inf Sci 47:70–84
Hussain Z, Iqbal S, Saba T, Almazyad AS, Rehman A (2017) Design and development of dictionary-based stemmer for the urdu language. J Theor Appl Inf Technol 95(15):3560–3569
Islam Md, Uddin Md, Khan M (2007) A light weight stemmer for Bengali and its use in spelling checker. Retrieved 24 March, 2019, from http://hdl.handle.net/10361/328
Ismailov A, Jalil MA, Abdullah Z, Rahim NA (2016) A comparative study of stemming algorithms for use with the Uzbek language. In: 3rd international conference on computer and information sciences (ICCOINS), 2016. IEEE, pp 7–12
Jaafar Y, Namly D, Bouzoubaa K, Yousfi A (2017) Enhancing Arabic stemming process using resources and benchmarking tools. J King Saud Univ-Comput Inf Sci 29(2):164–170
Jabbar A, Iqbal S, Khan MUG (2016a) Analysis and development of resources for Urdu text stemming. In: Proceedings of the 6th annual international conference on language and technology, KICS-CLE, UET Lahore
Jabbar A, Iqbal S, Akhunzada A, Abbas Q (2018a) An improved Urdu stemming algorithm for text mining based on multi-step hybrid approach. J Exp Theor Artif Intell. https://doi.org/10.1080/0952813X.2018.1467495
Jabbar A, Iqbal S, Khan MUG, Hussain S (2018b) A survey on Urdu and Urdu like language stemmers and stemming techniques. Artif Intell Rev 49(3):339–373
Jabbar A, Iqbal S, Khan MUG, Hussain S (2018b) A survey on Urdu and Urdu like language stemmers and stemming techniques. Artif Intell Rev 49(3):339–373
Jivani AG (2011) A comparative study of stemming algorithms. Int J Comp Tech Appl 2(6):1930–1938
Karaa WBA (2013) A new stemmer to improve information retrieval. Int J Netw Secur Appl 5(4):143
Karimi S, Wang C, Metke-Jimenez A, Gaire R, Paris C (2015) Text and data mining techniques in adverse drug reaction detection. ACM Comput Surv (CSUR) 47(4):56
Kastner I (2019) Templatic morphology as an emergent property. Nat Lang Linguist Theory 37(2):571–619
Khalid A, Hussain Z, Baig MA (2016) Arabic stemmer for search engines information retrieval. Int J Adv Comput Sci Appl 1(7):407–411
Khan S, Waqas A, Usama B, Xuan W (2015) Template based affix stemmer for a morphologically rich language. Int Arab J Inf Tech 12(2):146–154
Khoja S, Garside R (1999) Stemming arabic text. Lancaster University, Lancaster, UK, Computing Department
Krovetz R (2000) Viewing morphology as an inference process. Artif intel 118(1–2):277–294
Larkey LS, Ballesteros L, Connell ME (2007) Light stemming for Arabic information retrieval. Arabic computational morphology. Springer, Dordrecht, pp 221–243
Lemur (2016) https://www.lemurproject.org. Accessed 14 Aug 2018
Lennon M, Peirce DS, Tarry BD, Willett P (1981) An evaluation of some conflation algorithms for information retrieval. Inf Sci 3(4):177–183
Lovins JB (1968) Development of a stemming algorithm. Mech Transl Comput Linguist 11(1–2):22–31
Lucene (2018) https://lucene.apache.org, Accessed 12 Aug 2018
Mateen A, Malik MK, Nawaz Z, Danish HM, Siddiqui MH, Abbas Q (2017) A hybrid stemmer of punjabi shahmukhi script. Int J Comput Sci Netw Secur 17(8):90–97
McCormick C (2016) Word2Vec tutorial—the skip-gram model. https://www.mccormickml.com
Mishra U, Prakash C (2012) MAULIK: an effective stemmer for Hindi language. Int J Comput Sci Eng 4(5):711–717

Mochizuki M, Aizawa K (2000) An affix acquisition order for EFL learners: an exploratory study. System 28(2):291–304
Moghadam FM, MohammadReza K (2015) Comparative study of various Persian stemmers in the field of information retrieval. J Inf Proc Syst 11(3):450–464
Momenipour F, Keyvanpour MR (2016) PHMM: stemming on Persian texts using statistical stemmer based on hidden Markov Model. Int J Inf Sci Manag 14(2):107–117
Mustafa AM, Rashid TA (2018) Kurdish stemmer pre-processing steps for improving information retrieval. J Inf Sci 44(1):15–27
Nguyen, (2013) Nguyen DT, Leveling J (2013) Exploring domain-sensitive features for extractive summarization in the medical domain. International conference on application of natural language to information systems. Springer, Berlin, pp 90–101
Nwesri AFA, Alyagoubi HAH (2015). Applying arabic stemming using query expansion. In 2015 26th international workshop on database and expert systems applications (DEXA) (pp. 299–303). IEEE
Orengo VM, Huyck C (2001) a stemming algorithm for the portuguese language. In; SPIRE '01: Proceedings of eigth symposium on string processing and information retrieval, pp 186–193.
Paice CD (1990) Another stemmer. SIGIR Forum 24(3):56–61
Paice CD (1996) Method for evaluation of stemming algorithms based on error counting. J Am Soc Inf Sci 47(8):632–649
Paice CD (1994) An evaluation method for stemming algorithms. In: Proceedings of the 17th annual international ACM SIGIR conference on research and development in information retrieval. Springer, New York, pp 42–50
Pande BP, Tamta P, Dhami HS (2018) Generation, implementation and appraisal of an N-gram based stemming algorithm. Digit Scholarsh Humanit. https://doi.org/10.1093/llc/fqy053
Paik JH, Pal D, Parui SK (2011) A novel corpus-based stemming algorithm using co-occurrence statistics. In: Proceedings of the 34th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR'11). ACM, New York, pp 863–872
Patil CG, Patil SS (2013) Use of Porter stemming algorithm and SVM for emotion extraction from news headlines. Int J Electron Commun Soft Comput Sci Eng 2(7):9–13
Porter MF (2006) https://snowball.artarus.org/algorithms/english/ stemmer.html
Porter MF (1980) An algorithm for suffix stripping. Program 14(3):130–137
Qureshi AH, Hassan MU, Akhter S (2018) Towards description of derivation in Urdu: morphological perspective. Al-Qalam 23(2):96–100
Rani SPR, Ramesh B, Anusha M, Rani SJGR (2015) Evaluation of stemming techniques for text classification. Int J Comput Sci Mobile Comput 4(3):165–171
Rashid TA, Mohamad SO (2016) Enhancement of detecting wicked website through intelligent methods. International symposium on security in computing and communication. Springer, Singapore, pp 358–368
Rashidi A, Lighvan MZ (2014) HPS: a hierarchical Persian stemming method. arXiv preprint arXiv:1403.2837.
Rehman Z, Anwar W, Bajwa UI, Xuan W, Chaoying Z (2013) Morpheme matching based text tokenization for a scarce resourced language. PLoS ONE 8(8):e68178
Saad MK, Ashour W (2010) Arabic morphological tools for text mining. Corpora 18:19
Saeed AM, Rashid TA, Mustafa AM, Al-Rashid Agha RA, Shamsaldin AS, Al-Salihi NK (2018a) An evaluation of Reber stemmer with longest match stemmer technique in Kurdish Sorani text classification. Iran J Comput Sci 1(2):99–107
Saeed AM, Rashid TA, Mustafa AM, Fattah P, Ismael B (2018b) Improving Kurdish web mining through tree data structure and Porter's Stemmer algorithms. UKH J Sci Eng 2(1):48–54
Sarma B, Purkayastha BS (2013) An affix based word classification method of assamese text. Int J Adv Res Comput Sci 4(9):213–216
Schofield A, Mimno D (2016) Comparing apples to apple: the effects of stemmers on topic models. Trans Assoc Comput Linguist 4:287–300
Setiawan R, Kurniawan A, Budiharto W, Kartowisastro IH, Prabowo H (2016) Flexible affix classification for stemming Indonesian Language. In: 2016 13th international conference on electrical engineering/electronics, computer, telecommunications and information technology (ECTI-CON). IEEE, pp 1–6
Singh J, Gupta V (2016) Text stemming: approaches, applications, and challenges. ACM Comput Surv (CSUR) 49(3):45
Singh J, Gupta V (2017) An efficient corpus-based stemmer. Cognit Comput 9(5):671–688
Sirsat SR, Chavan V, Mahalle HS (2013) Strength and accuracy analysis of affix removal stemming algorithms. Int J Comput Sci Inf Technol 4(2):265–269

Sulaiman S, Omar K, Omar N, Murah MZ, Abdul Rahman HD (2014) The effectiveness of a Jawi stemmer for retrieving relevant Malay documents in Jawi characters. ACM Trans Asian Lang Inf Process (TALIP) 13(2):6
Suryani AA, Widyantoro DW, Purwarianti A, Sudaryat Y (2018) The rule-based sundanese stemmer. ACM Trans Asian Low-Resour Lang Inf Process (TALLIP) 17(4):27
Taghi-Zadeh H, Sadreddini MH, Diyanati MH, Rasekh AH (2015) A new hybrid stemming method for persian language. Digit Scholarsh Humanit 32(1):209–221
Thangarasu M, Manavalan R (2013) Design and development of stemmer for Tamil language: cluster analysis. Int J Adv Res Comput Sci Softw Eng 3(7):812–818
The free dictionary (2018) https://www.thefreedictionary.com/. Accessed 03 Aug 2018
Qunis I, Amati G, Plachouras V, He B, Macdonald C, Lioma C (2006) A high performance and scalable information retrieval plateform. In: SIGR workshop on open source information retrieval
Urdu L (2006) https://182.180.102.251:8081/oud/help_3.htm. Accessed 04 Aug 2018
Xapian (2018) https://xapian.org. Accessed 07 Aug 2018
Xer (1994) Xeror linguistic database reference, English version 1.1.4 ed.s
Yadollahi A, Shahraki AG, Zaiane OR (2017) Current state of text sentiment analysis from opinion to emotion mining. ACM Comput Surv (CSUR) 50(2):25
Zerrouki T (2016) Tashaphyne 0.2 (Online). https://pypi.python.org/pypi/Tashaphyne. Accessed 14 Apr 2016
Zhou D, Mark T, Brailsford T, Wade V, Ashman H (2012) Translation techniques in cross-language information retrieval. ACM Comput Surv (CSUR) 45(1):1

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.