Chapter 2

Literature Survey and Analysis of Trends

The main objective of this chapter is to provide a systematic literature survey on different preprocessing strategies evaluated in the text analysis domain. The literature survey helps practitioners, researchers, and project managers explore appropriate evaluation techniques for a particular domain. Moreover, state-of-the-art research facilitates a better understanding of the overall picture for the researchers and practitioners, like what has been studied by the community, what is still missing, and the advantages and pitfalls. We aim to distil the key concepts and analyze their applicability, scope and challenges. This chapter aims to systematically map the existing approaches, frameworks, and validation methods to intersect the place of preprocessing methods in the text analysis domain. We categorize the literature review into three broad categories: stopword removal, stemming and decompounding applied in different text analysis tasks.

2.1 Stopword Removal

In the early days of IR systems, Luhn [80] demonstrated that natural language texts are categorized into keyword and non-keyword terms. The keyword terms were essential and occurred very few times in a text document, whereas the non-keyword terms like 'of', 'or', 'the', and 'to' occur frequently in a collection. The non-keyword terms are also called stopwords in the text analysis domain. In the Brown corpus, Fox [41] observed that 42% of words belong to a set of hundred frequent words. Francis et al. [42] shows

that the ten most frequently used words belong to 20% - 30% of the tokens in an English document. These stopwords are known as classical or standard stopwords in the English language. The classical stopwords had two major drawbacks. Primarily, they are old and outdated and need to contain new stopwords on the Web. Secondly, stopwords are domain-independent and too generic.

Several stopword lists have been proposed and evaluated to overcome the limitations of classical stopword lists in the past few years. Generally, stopwords are categorized into two groups. The first group is information gain methods ([78], [13], [131]), and the second group is based on Zipf's law ([23], [83]). The information gain methods are based on the idea that the amount of information a particular term holds means a stopword possesses low information values. Few methods like Kullback-Leibler (KL) divergence measure [78], entropy measure [12], and maximum likelihood measure [13] are investigated to measure the informativeness of a term. Zipf's law [145] states that the frequency of a word is inversely proportional to its rank. In this method, the top few words are considered stopwords in the collection [78]. Many stopword lists have been proposed and evaluated in different languages and domains by applying the above mentioned methods. We outline the few stopword lists below.

The effect of stopword removal is investigated in different computational tasks like word segmentation, novelty detection, text classification, text categorization and text retrieval. Zou et al. [146] proposed a Chinese stopword list based on the aggregation approach. In this approach, different stopword generation approaches are tried, and a stopword list is generated taking a few top-ranked words extracted from each of them. The aggregation is done based on the word statistics and entropy. The authors found that an effective stopword list improves the accuracy of Chinese word segmentation. Davarpanah et al. [30] proposed a generic stopword list in the Farsi language by an aggregation method. The stopword list is extracted from syntactic classes, domain-dependent, corpus statistics and expert judgments. They observed that stopword removal improves the efficiency of the Farsi IR system and plays a vital role in text segmentation. Sadeghi and Vegas [112] proposed a light stopword list in the Farsi language based on the aggregation method. They aggregated on term frequency, normalized inverse document frequency and word entropy. They show that the first 32 Farsi stopwords significantly reduce the index size.

Yaghoub-Zadeh-Fard et al. [144] proposed an aggregation-based stopword list in the

Farsi IR system. The author presented a stopword list by aggregating the parts of the speech and statistical features of terms. The experimental results show that the stopword list enhances retrieval effectiveness, decreases index size and improves response time. Alajmi et al. [7] presented an aggregation-based stopword list for the Arabic language. They show that the proposed stopword list outperforms the general stopword list in the text categorization. Sarica and Luo [118] proposed an aggregation-based stopword list in technical language processing applications. They aggregated different statistical features of terms such as term frequency (tf), inverse document frequency (idf), tf-idf and entropy. The stopword list can be used as a complementary to NLTK and USPTO [1] list in text analysis tasks related to engineering disciplines.

Sinka and Corne [131] proposed a web-specific stopword list by calculating the entropy of a term. They show that the proposed stopword list outperforms the other lists in hard text classification. Jasleen and Jatinderkumar [57] created a manual stopword list in the Punjabi language. They also evaluated the effect of the stopword in poetry classification. Rakholia and Saini [104] presented a stopword list in the Gujarati language based on manual inspection and linguistic experts. They designed the stopword list for the machine translation application. Raulji and Saini [107] investigated a hybrid approach stopword list in Sanskrit. They extracted a list of seventy-five generic stopwords from seventy-six thousand words. El-Khair [2] evaluated the effect of general, corpus-based, combined stopword lists in Arabic IR. They show that the general stopword list provides better retrieval effectiveness than the other two.

Ul Haque et al. [141] proposed a corpus-based and finite state-based stopword list in the Bengali language. They show that the stopword removal improves the accuracy of the system by 90% and 80%, respectively. Jha et al. [60] presented a deterministic finite automata (DFA) based stopword removal algorithm for the Hindi language. They demonstrate that the proposed stopword list achieves an accuracy of 99%. Fox [41] proposed a stopword length 421 in English by considering the statistical feature of terms and DFA. They observed that the proposed stopword list improves the effectiveness of the text analysis task. Based on Fox [41] guidelines (Dolamic and Savoy [37] and Savoy [119]) proposed a stopword list for Bengali, Marathi, Hindi and French. They show that stopword removal improves retrieval effectiveness in different Indian and European languages

[1] https://www.uspto.gov/

IR.

Silva and Ribeiro [127] investigated the impact of different preprocessing strategies in text categorization. They used SVM for text categorization and found that the different preprocessing strategies improve precision, recall, f-measure and accuracy. Ayral and Yavuz [13] proposed a domain-specific stopword list and evaluated their effectiveness on text classification. They used a Bayesian classifier for text classification. They noticed that the stopword list improved the effectiveness of the classification of web pages. AlShargabi et al. [6] evaluated the effect of stopword removal in Arabic text classification. They used different classifiers such as Support Vector Machine (SVM), Sequential Minimal Optimization (SMO), Naive Bayes (NB) and J48. They evaluated the classifier accuracy by K-fold cross-validation, the time needed for classification and the percentage split method. ISIK and DAG [56] evaluated the effect of different preprocessing methods on text classification. They observed that the NLTK-based stopword elimination method improves the classifier effectiveness significantly. Kwee et al. [72] evaluated the effect of the preprocessing step in multilingual sentence-level novelty detection in English and Malay languages. They show that the preprocessing step improves novel sentence detection. Makrehchi and Kamel [83] proposed an automatic domain-specific stopword extraction method based on backward filter level performance and sparsity measure of training data. The proposed method provides more promising results than the idf and information gain approach.

Ladani and Desai [73] presented an overview of different stopword generation approaches in Indian and non-Indian languages. They outline the effect of stopword lists in text classification, IR and NLP applications. Saini and Rakholia [116] investigated a comprehensive analysis of stopword length in 42 different international languages comprising three continents: South America, Europe and Asia. They observed that most of the Asian language's stopwords are native-scripted, while the European languages are Roman scripted. The average number of stopwords for any given language could be 200. Many modern language applications like NLTK [2], CLTK [3] and Scikit-learn [4] provide different stopword lists in different languages. The NLTK supports a stopword list for 21 languages. The CLTK supports stopword lists for ancient languages. By default,

[2]https://www.nltk.org/

[3]https://github.com/cltk/cltk

[4]https://scikit-learn.org/

Scikit-learn supports an English stopword list. Different machine-learning algorithms are implemented through the Scikit-learn library.

The above study concludes that stopword removal improves effectiveness in different computational tasks like text retrieval, classification, segmentation and novelty detection. We also observed that different corpus-based stopword lists were proposed in low-resource Indian languages. These observations came from exploring various European and a few Asian languages. We summarize different stopword removal techniques, datasets, observations, and their application in Table 2.1.

Table 2.1: Summary of effect of stopword lists in text processing task

Year and Ref	**Approaches**	**Datasets**	**Observations**	**Applications**
2006 Zou et al. [146]	aggregation method	Chinese [5]	They aggregate the statistic and information of a term to generate a generic Chinese stopword list	Chinese language processing and NLP domain
2009 Davarpanah et al. [30]	aggregation method	Farsi	The author created a generic stopword length of 927 words by aggregation method. Stopword removal reduces space by about 39% and speeds up the efficiency of the search system	Farsi IR, text segmentation
2014 Sadeghi and Vegas [112]	aggregation based	Farsi	They observe that the first 32 Farsi stopwords significantly reduce the index size	Frasi text processing
2015 Yaghoub-Zadeh-Fard et al. [144]	aggregation method	Farsi	The author generated a stopword list by aggregating part of speech and statistical features of terms. The stopword list enhances retrieval effectiveness, decreases index size and improves response time.	Farsi IR
2012 Alajmi et al. [7]	aggregation method	Arabic	The author created a stopword list by aggregating different statistical features of terms. They notice that the aggregation-based stopword list outperforms the general list in text classification.	Text classification
2020 Sarica and Luo [118]	aggregation based	Patent text	The author proposed 87 stopwords for technical language analysis. The proposed stopword list outperforms the other stopword lists in multi-class text classification	Engineering domain, Text classification
2003 Sinka and Corne [131]	entropy-based	Random web pages, Bank search	The entropy-based stopword list outperform the other stopword lists in hard classification	Text classification
2016 Jasleen and Jatinderkumar [57]	manually collected	Punjabi	The author proposed 184 stopwords for Punjabi NLP task	Poetry classification
2016 Rakholia and Saini [104]	manual inspection and linguistic experts	Gujarati	The author proposed a list of 1125 unique stopwords	Machine Translation and NLP applications
2017 Raulji and Saini [107]	hybrid-based	Sanskrit	They proposed a stopword list that comprises 75 words	Sanskrit text processing

Table 2.1: Summary of effect of stopword lists in text processing task

Year and Ref	**Approaches**	**Datasets**	**Observations**	**Applications**
2017 El-Khair [2]	combination of statistical and linguistic approach	Arabic	They found that the general stopword list outperforms the corpus-based and combined lists in IR domain	Arabic IR
2019 Ul Haque et al. [141]	corpus-based	Bengali	The corpus-based stopword elimination provide an accuracy of 70-75% and precision of 100%	IR, Text Summarization, Text Mining
2016 Jha et al. [60]	Deterministic Finite Automata (DFA)	Hindi	The DFA based stopword list provide an accuracy of 99% and time-efficient	Hindi text processing
1989 Fox [41]	term frequency and DFA	English	Fox proposed a stopword length of 421 in English by considering the term frequency of words and few stopwords are added using minimal DFA	English language processing and IR
2010 Dolamic and Savoy [37], 1999 Savoy [119]	Fox [41] guideline	Bengali, Hindi, Marathi and French	Based on the Fox guidelines, the author proposed a stopword length of 165 for Hindi, 114 for Bengali, 99 for Marathi and 215 for French	IR
2003 Silva and Ribeiro [127]	Support Vector Machine	Reuters-21578 [6]	The stopword removal and stemming techniques improves the effectiveness of text categorization	Text categorization
2011 Ayral and Yavuz [13]	Bayesian classifier	English web page	They observed that the document coverage rank and topic coverage rank of words belonging to natural language corpora follow Zipf's law	Text classification
2011 AlShargabi et al. [6]	Support Vector Machine (SVM), Naive Bayesian, J48	Arabic	SVM outperforms the other classification techniques in terms of K-fold cross-validation, time needed for classification and percentage split method	Text classification
2020 ISIK and DAG [56]	NLTK [7] based	English	The NLTK based stopword removal technique improve the effectiveness of text classification	Text classification
2009 Kwee et al. [72]	novelty detection algorithm	TREC 2003 and 2004, English and Malay collection	They observe that different preprocessing strategies have a significant impact on the effectiveness of novelty sentence detection	Sentence-level novelty detection
2016 Saini and Rakholia [116]	comprehensive analysis	South America, Europe and Asian	They analyze different stopword lists and observed that the average number of stopwords for any language could be 200	NLP domain

2.2 Stemming

The stemming method enhances retrieval effectiveness by conflating different word variants into a common root or stem [84]. Traditionally, language-specific stemmers have been explored to reduce index size and improve retrieval effectiveness. Rule-based stemmers like Lovin [79] and Porter et al. [98] improve the retrieval effectiveness of a system. They implemented different rules to generate the root word. However, the stemming algorithm sometimes generates over-stemming (e.g., 'university' becomes 'univers', and 'wander' becomes 'wand') or under-stemming (e.g., 'India' and 'Indian' do not conflate to the same root), which reduce retrieval effectiveness. A better word conflation can be achieved using an online dictionary, as suggested by Krovetz [69]. Harman [46] observed that Lovins or Porter stemmers do not provide statistically significant results. A query-by-query analysis unveiled that the stemming method improves retrieval effectiveness in a set of queries, whereas it reduces effectiveness in another set of queries. Hajeer et al. [45] proposed an enhanced Porter stemming algorithm (EPSA) to overcome the drawbacks of the Porter stemming algorithm. They observed that the proposed algorithm gives fewer errors than the original Porter algorithm in both over-stemming and under-stemming measures. The enhanced Porter stemming algorithm improves precision by 2.3% and realizes the same recall percentage. In English, Fautsch and Savoy [40] show that morphological analysis does not provide better results than the stemming approach.

In recent years, the rapid growth of e-content in non-English languages requires efficient stemming techniques for other popular languages. Braschler and Ripplinger [17] observed that the stemming technique improves MAP score by 23% for short queries and 11% for long queries in German. Hollink et al. [54] evaluated the impact of different stemming methods in eight European languages. They observed that the stemming technique improves retrieval effectiveness and produces statistically significant results in Finnish and German, whereas the stemming method does not provide statistically significant results in English, French, Russian, Dutch and Spanish. In Swedish, algorithmic stemmers give better MAP scores. Depending on the complexity of languages, the effect of stemming strategy varies in the search system. The stemming method improves retrieval effectiveness in Finnish, while ignoring the stemming technique, which provides better retrieval effectiveness in English and Italian. In compounding languages like German, morpho-

logical analysis gives a better retrieval score. Similar observations were made in other morphologically rich languages like French [133], and they require an in-depth analysis (e.g., Finnish ([9], [66])). For a morphologically rich language like Finnish, a noun can have 2,000 different variations, but in real-world corpora, 84%-88% of inflected nouns are generated by only 6 of a possible 14 grammatical cases [62]. A lexical stemmer can handle the morphologically rich language, but these stemmers are not always freely available (E.g., Xelda system at Xerox). Also, the design and implementation are pretty complicated [140]. Mayfield and McNamee [85] proposed an n-gram-based indexing approach in European languages. They show that 4-gram performs best in different European languages. The major drawback of the n-gram approach is the size of the inverted index. This approach expands the index size substantially, which increases query processing time. The 4-gram model takes ten times more processing time than the word-based retrieval.

Recently, researchers have proposed rule-based, unsupervised and hybrid stemming techniques in different Indian languages. They evaluated the effectiveness of stemmer using over-stemming, under-stemminng and accuracy. The accuracy of the stemmer is calculated using the fraction of words stemmed correctly. Ramanathan and Rao [105] proposed a rule-based stemmer in 'Hindi'. The author created 65 suffixes manually and truncated the longest possible suffix at first, followed by the shortest suffix. The stemmer provides an under-stemming and overstemming error of 4.68% and 13.84%, respectively. Prajitha et al. [99] investigated a rule-based stemming technique in the Malayalam language. The stemmer offers an accuracy of 86.6%. Majgaonker and Siddiqui [81] presented a rule-based and unsupervised stemmer in Marathi. The rule-based and unsupervised stemmer offers an accuracy of 81.6% and 82.05%, respectively. Dolamic and Savoy [37] proposed a light (inflectional) and aggressive (derivational) stemmer in Bengali, Hindi and Marathi languages. They removed inflectional and derivational suffixes that frequently occurred from nouns and adjectives. They observed that the stemming method improves retrieval effectiveness in the IR domain. Saharia et al. [113] proposed a rule-based and hybrid stemming technique for the Assamese language. The rule-based and hybrid stemming techniques provide an accuracy of 61% and 82%, respectively. Patel et al. [96] proposed a hybrid approach (unsupervised and rule-based) lightweight stemmer for Gujarati. The stemmer achieved an accuracy of 67.86%. Jiandani and Bhattacharyya [136] investigated a rule-based and hybrid stemming technique in Gujarati. They enhance the effectiveness

of the stemmer using the POS (Part Of Speech) module and a set of substitution rules. The rule-based and hybrid stemmer have an accuracy of 70.5% and 90.7%, respectively. Mishra and Prakash [87] proposed a hybrid stemmer called 'Maulik' in Hindi. The stemmer offers an accuracy of 91.59% and reduces the under-stemming and over-stemming errors. Kumar and Rana [71] proposed a Punjabi stemmer based on a brute force algorithm and suffix stripping approach. They observed that the stemmer provides an average accuracy of 80.73%.

The statistical stemmer provides comparable effectiveness to rule-based stemmers when evaluated in different languages. Xu and Croft [142] proposed a corpus-based stemming method based on word frequency and co-occurrence statistics. The effectiveness of stemmer is evaluated in different languages, such as English newspapers, legal and Spanish texts. Majumder et al. [82] proposed an unsupervised suffix stripper called 'YASS'. The stemmer improves effectiveness in different Indian and European languages IR. Paik and Parui [94] proposed a corpus-based stemming technique in morphologically rich languages. They strip the suffixes in an unsupervised manner. They notice that the stemming strategy improves retrieval effectiveness in agglutinative languages like Bengali, Marathi and Hungarian. Paik et al. [93] proposed a corpus-based stemming method using co-occurrence statistics. They evaluated the effectiveness of stemmer in different European and Asian languages. They observed that the stemming method improves retrieval effectiveness in different low-resource languages. Paik et al. [92] investigated a graph-based stemmer called 'GRAS'. They show that the stemming procedure improves retrieval effectiveness in highly inflectional languages. However, Porter stemmer offers slightly better results than statistical stemmers for English. Brychcín and Konopík [18] proposed a high-precision stemmer in two stages. In the first stage, they applied clustering techniques to prepare large-scale training data. In the second stage, they use a maximum entropy classifier to decide when and how to stem a particular word. They observed that the proposed stemming technique efficiently handles unseen words.

Based on the above facts, we conclude that the stemming method improves retrieval effectiveness in European, Asian and South Asian languages. We also observe that the stemming technique enhances effectiveness in text analysis domains. However, the impact of stemming strategy is less explored in Sanskrit. Hence, we study the effect of the stemming method in the Sanskrit NLP and IR domains. We summarize different stemming

techniques, datasets, observations, and their application in Table 2.2.

Table 2.2: Summary of effect of stemming techniques in text analysis task

Year and Ref	**Approaches**	**Datasets**	**Observations**	**Applications**
1968 Lovin [79]	rule-based	English	Lovin stemmer removes the suffixes by implementing 35 rules. They looked at 294 suffixes, with the longest suffix eliminated at first. The stemming technique improves the effectiveness of an IR system	English text processing and NLP domain
1980 Porter et al. [98]	rule-based	English	Porter stemmer is a rule-based suffix stripping stemmer, where the suffixes are truncated sequentially until no matching suffix is left out	English text processing and IR
2017 Hajeer et al. [45]	rule-based	English	They show that the enhanced Porter stemmer provides fewer errors and better effectiveness than the original Porter stemmer	IR, Web search engine
2004 Braschler and Ripplinger [17]	language-independent and rule-based	German	The author observed that the stemming method improves MAP by 23% for short queries and 11% for long queries	IR
2003 Mayfield and McNamee [85]	N-gram based	European languages	The major drawback of the n-gram approach is the size of the inverted index. This approach expands the index size substantially, which increases query processing time	IR
2003 Ramanathan and Rao [105]	rule-based	Hindi news magazine	The author created 65 suffixes manually, and suffixes are truncated to generate the root word. The stemmer provides an under-stemming and an over-stemming error of 4.68% and 13.84% respectively	Hindi text processing
2013 Prajitha et al. [99]	rule-based	Malayalam	The stemmer provide an accuracy of 86.6%	NLP domain
2010 Majgaonker and Siddiqui [81]	rule-based and unsupervised	Marathi news corpus	The rule-based stemmer offers an accuracy of 81.6% and unsupervised stemmer achieves a maximum accuracy of 82.05%	Text processing
2010 Dolamic and Savoy [37]	rule-based	Bengali, Hindi and Marathi	The author proposed two rule-based suffix stripping stemmers, i.e., 'light' and 'aggressive' stemmers in different Indian languages. The 'aggressive' stemmer performs best in the IR domain	Text Processing and IR
2012 Saharia et al. [113]	rule-based and hybrid approach	Assamese EMILLE [8] corpus	The rule-based and hybrid stemming method offers an accuracy of 61% and 82%	NLP domain

Table 2.2: Summary of effect of stemming techniques in text analysis task

Year and Ref	**Approaches**	**Datasets**	**Observations**	**Applications**
2010 Patel et al. [96]	hybrid approach	Gujarati EMILLE corpus	The stemmer implement an unsupervised and rule-based technique to generate the root word. Stemmer provide an accuracy of 67.86%	Gujarati text processing
2011 Jandani and Bhattacharyya [136]	rule-based and hybrid approach	Gujarati	The rule-based and hybrid stemming technique offers an accuracy of 70.5% and 90.7% respectively	NLP domain
2012 Mishra and Prakash [87]	hybrid approach	Hindi magazine	The stemmer provide an accuracy of 91.59% and reduces the understemming and over-stemming error	Hindi NLP domain
2010 Kumar and Rana [71]	brute force	Punjabi	The stemmer provide an accuracy of 80.73%	NLP domain
1998 Xu and Croft [142]	corpus-based and co-occurrence statistics	English and Spanish	The corpus-based stemming technique can be implemented in any low resource languages and improve the effectiveness of an IR system	IR
2007 Majumder et al. [82]	clustering approach	English, French, Bengali	The cluster-based stemming method improve effectiveness in European and low resource Indian languages IR	IR
2011 Paik and Parui [94]	corpus-based	Bengali, Marathi, Hungarian and English	The stemming strategy enhance retrieval effectiveness in agglutinative languages like Bengali, Marathi and Hungarian	IR
2011 Paik et al. [93]	co-occurrence based	European and Asian languages	The stemming strategy outperform different rule-based stemmer in different languages	IR and NLP domain
2011 Paik et al. [92]	graph-based	European and Asian languages	The stemming technique outperform different rule-based and statistical approaches in European and Asian languages	IR and NLP domain
2015 Brychcín and Konopík [18]	clustering and maximum entropy classifier	Czech, Slovak, Polish, Hungarian, Spanish and English	The author presented a multi-purpose stemming tool. Moreover, the stemming technique provides excellent results in the IR	IR and language model

2.3 Decompounding

In a morphologically rich language, compound words are generated by concatenating two or more words, using sandhi [9] principles, or combining several morphemes using linking elements. More compound words in a lexicon affect the effectiveness of computational tasks like machine translation, information extraction, and creating out-of-vocabulary words in the dictionary. So, the decompounding method is an essential pre-processing step in the text analysis domain. Early literature suggests that the decompounding techniques primarily use basic 'mechanical' segmentation methods based on string matching. The most common method in this approach is dictionary-based. Different machine learning-based decompounding models have been proposed in the last decade. These model uses a feature engineering approach to perform the decompounding operations. In recent years, deep learning-based decompounding models have performed best in morphologically rich European and Asian languages. We formalized the state-of-the-art decompounding models in three subsections: corpus-based, machine-learning, and deep learning-based.

2.3.1 Corpus-based decompounding methods

Koehn and Knight [65] proposed a corpus-based decompounding method in German-English machine translation. They used the geometric mean of word frequency to locate the splitting point. They observed that the decompounding method improves the efficiency of machine translation. Hollink et al. [54] evaluated the effect of stemming, lemmatization and decompounding algorithms in eight European languages. They observed that the corpus-based decompounding model improves retrieval effectiveness in Dutch, Finnish, German, and Swedish languages. The decompounding algorithm improves MAP ranges from 4%-18.7% in different morphologically rich European languages. Braschler and Ripplinger [17] investigated the effect of stemming and decompounding methods in German IR. They found that stemming and careful decompounding methods enhance the effectiveness of an IR system. Leveling et al. [76] evaluated a corpus-based decompounding model in German-English cross-language patent retrieval. They noticed that the decompounding method reduces the out-of-vocabulary problem and improves retrieval effectiveness significantly. Ganguly et al. [43] observed that relaxed and co-occurrence-based constituent

[9] The word Sandhi means placing together

selection techniques outperform the standard frequency-based decompounding technique in Bengali IR. Sahu and Mamgain [114] applied a corpus-based decompounding strategy in Sanskrit and improved the splitting accuracy by different ranking methods. They found that the decompounding model enhances precision, recall, and accuracy.

Deepa et al. [32] presented a decompounding method for Hindi speech synthesis. They used a trie-based dictionary for the splitting of a compound word. They show that the algorithm has a split rate of 92%-96% of the input compound words, and the percentage of splitting accuracy is 83%-87%. Erbs et al. [39] evaluated the effect of the decompounding strategy in the keyphrase extraction in German. They found that the decompounding method enhances the word frequency count and identifies more keyphrase candidates. Alfonseca et al. [8] presented a language-independent decompounding technique in German. They observed that the decompounding method improves the efficiency of precision, recall, and accuracy in morphologically rich European languages. Laureys et al. [75] described a compound module that decreases the lexicon size and improves the word error rate for LVCSR in Dutch. The compound module combines a rule-based approach with statistical pruning. They observed that the decompounding module decreased the 30% lexicon size and improved the 11% word error rate in Dutch broadcast news recognition. Devadath et al. [34] proposed a hybrid sandhi splitting method in Malayalam text. They applied statistical methods and pre-defined character-level sandhi rules. They observed that the hybrid sandhi splitting technique outperformed the rule-based approach.

2.3.2 Machine learning-based decompounding methods

Pellegrini and Lamel [97] investigated an unsupervised data-driven decompounding algorithm in automatic speech recognition for Amharic text. They observed that the decompounding algorithm reduces the out-of-vocabulary (OOV) words from 35% to 50% and slightly reduces the word error rate (WER). Daiber et al. [27] proposed an analogy-based greedy unsupervised decompounding algorithm for machine translation. They found that the proposed decompounding algorithm is highly effective for ambiguous compounds. They observed that the semantic analogy-based compound splitting algorithm outperformed the frequency-based compound splitting technique in German-to-English machine translation. Haruechaiyasak et al. [47] investigated the effect of word segmentation in the

Thai language. They looked at two different word segmentation methods. One is based on dictionaries, while the other is based on machine learning. In the dictionary approach, they use longest-matching and maximal matching techniques, whereas, in the machine learning approach, they use Naive Bayes (NB), Decision Tree, Support Vector Machine (SVM), and Conditional Random Field (CRF) techniques. They observed that the CRF outperformed all other word segmentation approaches.

Ajees and Graham [4] presented a hybrid decompounding technique in Malayalam. The decompounding method integrates both rule-based and machine-learning-based approaches. The proposed method can be used as an essential pre-processing step in different NLP applications like machine translation, question answering, and extractive summarization. Das et al. [28] investigated the impact of the hybrid-based compound splitting method in Malayalam. They use a machine learning approach to classify the word as being split or not. Subsequently, a rule-based sandhi splitter is used to split the word. Xue [143] proposed a maximum entropy model for Chinese word segmentation. They found that the maximum entropy model can handle unknown words more efficiently than the maximum matching algorithm. They show that the machine learning model provides a precision of 95.01% and a recall of 94.94%. Shree et al. [125] used a CRF tool to split a compound word in the Kannada language. The effectiveness of the CRF tool is verified using five different combinations of training data. They achieved a compound splitting accuracy of 91.19%.

2.3.3 Deep learning-based decompounding methods

Chen et al. [20] proposed a neural network-based LSTM architecture for Chinese word segmentation. They experimented with different LSTM layers and dropout rates. They show that the LSTM architecture provides a precision of 97.5%, recall of 97.3% and F-score of 97.4% in the MSRA dataset. Kitagawa and Komachi [64] presented a deep learning-based LSTM architecture for Japanese word segmentation. They used a character-based n-gram embedding and a gold dictionary created from the test corpus. They show that the proposed model achieves an F1-score of 98.67% in Japanese word segmentation. Premjith et al. [100] proposed different deep learning-based decompounding methods in Malayalam. They achieve an accuracy of 98.08% in recurrent neural networks (RNN), 97.88% in long short-term memory (LSTM), and 98.16% in gated recurrent units (GRU). Hellwig [49]

presented a neural network-based decompounding approach in Sanskrit. The neural network comprises an input layer, hidden forward and backward layer. They used LSTM as a hidden layer to avoid the vanishing gradient problem. They observed that the decompounding method improves precision, recall and accuracy. Reddy et al. [108] introduced a deep sequence to sequence (seq2seq) model that takes the sandhied string as the input and predicts the unsandhied string as output. The model comprises multiple layers of LSTM cells with attention. They show an improvement 16.79% in the F-score to the current state-of-the-art technique.

Hellwig and Nehrdich [50] developed a Sanskrit word segmentation model using character-level convolutional and recurrent neural networks. The proposed model achieves a sentence splitting and string splitting accuracy of 85.2% and 96.7%, respectively, compared to the state-of-the-art technique. Aralikatte et al. [11] presented a deep double decoder (DD-RNN) with an attention model for the Sanskrit sandhi splitting task. They show that the model predicts the splitting location and prediction accuracy of 95% and 79.5%, respectively, which outperforms the state-of-art by 20%. They also demonstrate the model's generalization capability by applying the word segmentation method in Chinese. Dave et al. [31] proposed two-stage deep learning-based decompounding in Sanskrit. In the first stage, they predict the sandhi window using the RNN model. In the second stage, the sandhi window is split into two words using the seq2seq model. They show a location prediction and split prediction accuracy of 92.3% and 86.8%, respectively.

Based on the above findings, we conclude that the decompounding technique plays a vital role in text analysis tasks. The effect of the decompounding method is thoroughly investigated in European and a few Asian languages. However, no previous studies exist on decompounding techniques in Indian languages like Marathi, Hindi and Sanskrit from an IR perspective. Hence, we evaluated different decompounding techniques in different Indian languages in the IR domain. We summarize different decompounding techniques, datasets, observations, and their application in Table 2.3.

Table 2.3: Summary of effect of decompounding models in text processing task

Year and Ref	Approaches	Datasets	Observations	Applications
2003 Koehn and Knight [65]	corpus-based	German	The corpus-based decompounding model improve the efficiency of German-English machine translation	Machine Translation
2004 Hollink et al. [54]	corpus-based	European languages	The author show that the corpus-based decompounding model enhances the retrieval effectiveness in morphologically rich European language IR	IR
2004 Braschler and Ripplinger [17]	corpus-based	German	The corpus-based decompounding module improve the effectiveness of German monolingual retrieval	IR
2011 Leveling et al. [76]	corpus-based	German patent	The corpus-based decompounding model reduce the out-of-vocabulary problem and improve the retrieval effectiveness of an IR system	Out-of-Vocabulary problem and IR
2013 Ganguly et al. [43]	corpus-based	Bengali	The corpus-based decompounding model improve the retrieval effectiveness of an IR system	IR
2019 Sahu and Mamgain [114]	corpus-based	Sanskrit	The corpus-based decompounding model improve the splitting accuracy	NLP domain
2004 Deepa et al. [32]	dictionary approach	Hindi	The decompounding module provide a splitting accuracy of 83%-87%	speech synthesis
2015 Erbs et al. [39]	dictionary approach and software tool	German	Decompounding process improve the word frequency count and identifies more keyphrase candidates	Keyphrase Extraction, IR
2008 Alfonseca et al. [8]	language-independent	German	The decompounding module improve effectiveness in morphologically rich European languages	NLP domain
2002 Laureys et al. [75]	hybrid approach	Dutch	The author observed that the decompounding module decreases lexicon size significantly and improves word error rate in broadcast news recognition	large vocabulary continuous speech recognition
2014 Devadath et al. [34]	hybrid based	Malayalam	The hybrid based decompounding module (statistical and rule-based) outperform the other rule-based approach and provide an accuracy of 91.1%	POS tagging, Topic modelling and Document indexing

Table 2.3: Summary of effect of decompounding models in text processing task

Year and Ref	Approaches	Datasets	Observations	Applications
2009 Pellegrini and Lamel [97]	unsupervised method	Amharic	The decompounding algorithm reduces the out-of-vocabulary words and word error rate	automatic speech recognition
2015 Daiber et al. [27]	unsupervised method	German	The decompounding model outperform the frequency-based approach in terms of coverage and accuracy	machine translation
2008 Haruechaiyasak et al. [47]	dictionary and machine learning approach	Thai	The conditional random field (CRF) approach outperform other word segmentation methods	Text Processing
2018 Ajees and Graham [4]	hybrid approach	Malayalam	The hybrid decompounding model perform best in terms of splitting accuracy	machine translation and question answering
2003 Xue [143]	maximum entropy	Chinese	The machine learning model handles the unknown words efficiently and improves effectiveness in text segmentation	NLP domain
2016 Shree et al. [125]	CRF tool	Kannada	The CRF tool improve the splitting accuracy	Text Processing
2015 Chen et al. [20]	long short term memory (LSTM)	Chinese	The neural network based architecture enhance the effectiveness of word segmentation	NLP domain
2018 Premjith et al. [100]	RNN, LSTM, and GRU	Malayalam	The neural network based architecture improve the effectiveness of compound splitting accuracy	NLP domain
2018 Reddy et al. [108]	LSTM with attention	Sanskrit	The decompounding model enhance effectiveness in the field of NLP	Text Processing
2018 Hellwig and Nehrdich [50]	character-level convolutional and RNN	Sanskrit	The decompounding model outperform the other model in sentence and string splitting accuracy	NLP domain
2018 Aralikatte et al. [11]	double decoder RNN with attention	Sanskrit	The generalized decompounding model split the Sanskrit and Chinese compound word efficiently	Text Processing
2018 Dave et al. [31]	RNN and seq2seq model	Sanskrit	The neural network based decompounding model split the compound word efficiently and outperform the dictionary-based approach	Speech Synthesis, Neural machine translation, morphological analysis