

Supervised Automatic Text Summarization of Konkani Texts Using Linear Regression-Based Feature Weighing and Language-Independent Features



Jovi D'Silva and Uzzal Sharma

Abstract Automatic summarization of text documents is a widely researched domain in natural language processing. A lot of research is carried out on the most commonly spoken languages in the world. Automatic text summarization needs to be explored to include some of the less popular languages in the world to help sustain such languages and promote their use. A language-independent summarization system that can be effortlessly extended to other such languages, which could have a limited number of resources to carry out such research is required. In this paper, we examine the efficiency of supervised linear regression models for the performing single document extractive automatic text summarization on Konkani language folktales dataset. We use 13 language-independent features and linear regression models to learn feature weights. These weights are then used to calculate a sentence's score; top ranking sentences are then chosen for summary generation. We employ a k-fold evaluation strategy to evaluate the system-generated summary against a human-generated summary using ROUGE evaluation toolkit. Additionally, we also evaluate the use of L1 and L2 regularization on the summarization task. The work represents early attempts in automatic text summarization pertaining to Konkani language, and the dataset employed in these experiments is unique and devised particularly to facilitate research in this domain. The language-independent features used can be readily extended to other low-resource languages. The systems implemented in this work performed better as compared to an unsupervised system based on k-means approach and also beat the baseline systems.

Keywords Supervised machine learning · Text summarization · Konkani · Regression · Extractive · Natural language processing · Low resource · Language-independent features

J. D'Silva (✉) · U. Sharma
Assam Don Bosco University, Guwahati, Assam 782402, India
e-mail: jovidsilva@gmail.com

U. Sharma
e-mail: uzzal.sharma@dbuniversity.ac.in

1 Introduction

Automatic summary generation has gained tremendous significance over a period of years, mainly because of the enormous amount of information available on the Web today [1]. It is crucial to provide users with a gist of the document so that one can determine if they need to invest their time reading an entire article or not. Text summarization condenses the contents of a long text document into a concise outline that preserves the meaning of the original document [2, 3]. The field of text summarization has gained immense popularity in Natural Language Processing (NLP). Hence, when the summarization of texts is done automatically by a computer system, it is termed as 'Automatic Text Summarization' (ATS). The summaries produced by ATS systems can be categorized as abstractive summaries or extractive summaries. Abstractive summaries are those, wherein the summary is generated by capturing the meaning of the information conveyed in the text document and then generating a summary that may be reworded by the summarizer. Extractive summaries are the ones that are generated by picking up relevant sentences or phrases in their original form from the source document to generate the summary [4, 5].

There is a growing need to explore the field of ATS to include the less-popular languages in ATS. Languages that are spoken by most of the global population, like English, have been the primary candidates for such research. However, the Internet houses articles, books, and other work in many other languages too. Therefore, there is a need for automatic text summarization tools that can effortlessly summarize documents written in other, less popular languages, as well. The methods that are used to implement such text summarization tools can be classified as language dependent or language-independent methods [6]. Language-dependent methods rely on tools related to the language, like lexicons of words for stop-word elimination, language-specific dictionary for lemmatization, and so on [7]. A language-independent system does not depend on any such language-specific tools or knowledge [7].

In this paper, we propose a language-independent method using supervised machine learning approach for generating extractive text summaries by using linear regression and language-independent features. The method was used to experiment on a dataset of Konkani language documents, specifically designed for ATS research [8]. An attempt was made to extend the domain of ATS to a low-resource language, Konkani, which is a language spoken by a minority population of India. A language with a limited data, tools, language experts, and/or speakers is considered as a low-resource language. This was the primary motivation when using language-independent features for the experiment. Konkani fits this criterion since it has not been a preferred language for research in NLP.

Konkani language is spoken along the Konkan coast, that lies on the west coast of India and is spoken by only 2.2 million people in India as per the census of India data of 2011 [9], which amounts to just about 0.19% of the total population of the country. These numbers have rapidly declined over the previous decade and continue to decline. Hence, it is very crucial, given the current status of this language, to help its digital growth by providing readers with digital versions of Konkani texts

and encouraging more people to read and contribute their literary knowledge of the language. Konkani ATS tool could help in boosting the digital growth of the language by summarizing text documents to its readers. Konkani is one of the oldest languages of the country, with the earliest scriptures dating back to 1187 A.D. [10]. This paper illustrates some of the initial attempts at including a low-resource language Konkani into the field of ATS.

The proposed technique uses linear regression algorithms, ordinary least squares, ridge regression, and Least Absolute Shrinkage and Selection Operator (LASSO) regression, for determining a sentence score and producing a summary of top ranking sentences. We use 13 language-independent features to represent the sentences of a text document. Traditionally, linear regression algorithms function by evaluating the values of multiple independent variables and then predicting the value of a dependent variable. The linear regression algorithms learn a set of coefficients to make a prediction as a weighted sum. In this experiment, we use these coefficients as feature weights. These feature weights are then used for calculation of ranking scores of sentences. The sentences are then selected to form a summary based on these ranks.

The major contributions of this study are as follows:

- It is the first attempt at studying the use of supervised linear regression algorithms for sentence scoring-based extractive ATS of Konkani language folktales.
- To examine the use of language-independent features for a low-resource language, Konkani.
- To examine the effects of L1 and L2 regularization on the text summarization task using regression.
- To compare and evaluate, ordinary least squares, ridge regression, and Least Absolute Shrinkage, and Selection Operator (LASSO) regression for text summarization.
- To evaluate the summaries generated by the system against the lead baseline, modified Luhn's heuristic baseline, and human-generated summaries using ROUGE toolkit.

The paper is organized as follows; the related work is given in Sect. 2. Section 3 highlights the dataset details. Section 4 gives the idea of the overall approach followed for summarization. Section 5 gives the details of the algorithms. Section 6 describes the choice of regressands. Section 7 outlines the features used for feature extraction. Section 8 recounts the methodology. Section 9 presents the results and discussion followed by the conclusion in Sect. 10.

2 Related Work

In the year 2009, Ulrich et al. proposed a Machine Learning (ML) approach based on regression for the summarization of emails. They showed that classifiers, based on regression, yielded better output compared to binary classifier [11]. Esther Hannah et al. presented an automatic text summarizer designed to produce summaries

like humans. They generated a model that approximated a relation by utilizing multivariate statistical technique [12].

Xie and Liu illustrated an innovative approach to provide extractive summaries of lengthy meetings with supervised learning technique using regression [13]. Gupta and Lehal proposed an approach for the summarization of Punjabi language news articles by applying mathematical regression. It aided in the estimation of "text feature weights." They considered statistical and linguistic features while determining the relevance of a sentence [14].

Aristoteles et al. proposed using binary regression techniques to perform text feature weighting on Indonesian text [15]. Kumar et al. presented a technique for extractive summary generation using regression. They used ten distinct features, and weights were assigned to features using "weight learning method" [16]. Dlikman and Last explored and evaluated the performances of several ML algorithms when applied to perform extractive text summarization. They introduced an innovative technique based on a "similarity score" between a prospective sentence and the "benchmark summaries" [17, 18].

Oliveira et al. illustrated a novel regression-based methodology using "Integer linear programming (ILP)" for the summarization of single document. This proposal is based on the premise that "no single summarization technique can achieve high performance for all input articles." This new regression-based technique uses ILP for the generation of multiple probable summaries for every document. Thereafter, a regression method was applied to choose the summary that was the most "informative" of the lot [19].

Zopf et al. underlined that sentence regression technique determines the significance of every sentence with the help of "learned utility scores." Thereafter, the scores were used to pick sentences from the original document under a greedy selection strategy. They proposed that learning the prediction of "ROUGE precision scores" led to getting enhanced output [20]. Malallah and Ali built a mathematical model for weight estimation between independent and dependent variables by utilizing linear regression technique for multi-document summarization task. They used TAC 2011 dataset comprising English language documents and the results were evaluated using ROUGE [21].

Rezaei et al. applied supervised machine learning algorithms, like regression and classification, to extractive summarization of Persian news documents. They depicted the sentences of a document as feature vectors and assigned ranks to the sentences by considering the overlap with the "golden summaries" written by humans. The algorithms then learned the prediction of these ranks that were to be assigned to sentences of any document provided as input [22]. Cagliero and Quatra used regression for extracting highlights from scientific papers. The regression-based model trained on various features that described similarity between highlights and candidate sentences. The model predicted the similarity between the highlights and the sentences in the paper to be annotated [23].

El-Kassas et al. highlighted and compared various ATS methods. They noted that simple regression-based machine learning models achieve better results compared to other models used for summarization [24, 25].

To summarize, extractive summarization has been effectively used for text summarization. Linear regression-based methods have also been used for text summarization based on feature weighing [12]. Most research has been done on popular languages such as English. We use folktales written in Konkani for our experiment, and Konkani being a low-resource language, tools and datasets were not easily available. A ML system based on unsupervised k-means clustering algorithm using the same Konkani dataset has been presented in [26]. Another approach using supervised ML technique for extractive ATS using linear and nonlinear classification algorithms is presented in [27]. There is a need to expand the research to other such low-resource languages. Thus, we propose the use of language-independent features, and we employ linear regression algorithms for feature weighing-based sentence scoring. Further, we examine the impact of L1 and L2 regularization on the text summarization task. The summaries thus generated are evaluated against human-generated summaries.

3 Dataset

The dataset employed for this research using regression methods was specifically compiled to carry out research pertaining to ATS in Konkani literature [8]. The dataset comprises Konkani language folktales compiled from five unique books written by various authors. Folktales are short stories that have cultural significance with a lesson or a moral and were traditionally passed on verbally from a generation to another. “Konkani” is the primary language spoken along the Konkan Coast of India and has not been a popular language of choice for research in ATS. The motivation for producing a dataset from scratch was that there was no existing dataset in Konkani language literature at the time this study was undertaken. Hence, the dataset is unique, and the stories in this dataset are compiled from five rare books that were also difficult to procure. It includes 71 stories that are written in Devanagari script [8]. The dataset comprises 9849 sentences, and every story has an average of 138 sentences and 1155 words. The longest story in the dataset comprises 2852 words, and the shortest story comprises 520 words.

Two Konkani language experts manually generated “gold-standard” summaries for each of the 71 stories in the dataset, which were used for evaluation of the quality of the system-generated summaries. Apart from this, a new benchmark was presented comprising extractive summaries of 300 words composed using the sentences manually marked as relevant to include in an extractive summary by the same language experts. Every sentence in the documents was annotated by the language experts. This labeled data were used for training the ML algorithms. The summaries produced by the system and the human-annotated benchmark are extractive; these were then evaluated against the pair of “gold-standard” summaries to verify how closely they matched. The word limit of 300 words for summary length was chosen in consultation with the language experts, as it would fill up a page. Also, the length of the

shortest story is 520 words; keeping this mind, 300 words limit seems apt for the summary length. The details of the dataset construction can be found in [8].

4 Approach

We use a supervised ML approach, where we use linear regression algorithms to learn feature weights given the feature vectors and regressands in the training instances. The algorithms used, especially with regularized algorithms, are affected by the scale of the features. Thus, before the training, all the features are scaled between 0 and 1. The regressands used are “1.0” and “0.0,” which indicate if a sentence is to be included in the system-generated summary by marking each sentence with “1.0” and “0.0” if it is not to be included in the summary.

We make use of three linear regression algorithms: ordinary least squares, ridge regression, and LASSO regression. Linear regression algorithms use feature weights, also known as coefficients, to make a prediction as a weighted sum. Regularization is a means of placing a penalty on complex models. In terms of linear regression, it shrinks the coefficients or feature weights. Ordinary least squares algorithm does not perform any form regularization, but ridge regression performs L2 regularization which only shrinks feature weights and prevents over-fitting. LASSO performs L1 regularization which not only shrinks the feature weights but also can perform feature selection by setting some feature weights to exact zero, effectively negating their presence. The feature weights learnt by each of the three algorithms are then used to calculate a sentence score; the top scoring sentences are then selected to form the final summary.

Linear regression algorithms based on ML have shown to achieve better results compared to other models for summarization tasks [24]. Also, linear regression algorithms are easier to train, show better performance when generalizing new and unseen data and are not prone to over-fitting.

5 Regression Models for Text Summarization

Scikit Learn, a ML library, was used for implementing the algorithms [28]. Traditionally, in linear regression technique, the values of multiple independent variables are used for the estimation of the value of a dependent variable. These regression algorithms learn a set of coefficients to make predictions as a weighted sum. In this experiment, we use these coefficients as feature weights.

Our dataset contains 13 language-independent features. The model learns the weights of these 13 features. The weights indicate the relevance of each feature. The weighted sum of features is used as a sentence score, with the top sentences being selected to form the summary.

The three linear regression algorithms used for the experiments are ordinary least squares, ridge regression, and least absolute shrinkage and selection operator (LASSO) regression.

5.1 Ordinary Least Squares

This method constructs a model for the prediction of unknown variables that minimizes the sum of the squared errors between the predicted value and the observed values. Ordinary least squares algorithm does not perform any form of regularization. Mathematically, it is illustrated by Eq. (1).

$$L = \min_w \|X_w - y\|_2^2 \quad (1)$$

where X is training data, y is regressands, $w = (w_1, \dots, w_p)$ are the coefficients, and L is the objective function to minimize.

5.2 Ridge Regression

Ridge regression solves some of ordinary least squares' issues by placing a penalty on the coefficients' size [28]. The equation is given by Eq. (2).

$$L = \min_w \|X_w - y\|_2^2 + \alpha \|w\|_2^2 \quad (2)$$

where $\alpha \geq 0$ is the complexity parameter that regulates the degree of "shrinkage," the larger the " α " value, greater is the shrinkage. Ridge regression uses L2 regularization which is given by $\alpha \|w\|_2^2$. In Scikit Learn, we set the " α " hyperparameter to 0.08 [28].

5.3 LASSO Regression

LASSO regression, unlike ridge regression, does not take the square of the coefficients. It uses L1 regularization which can result in zero coefficients. So, LASSO not only puts a penalty on the coefficients but also aids in selecting important features. Mathematically, it can be represented by Eq. (3).

$$L = \min_w \|X_w - y\|_2^2 + \alpha \|w\|_1 \quad (3)$$

where “ α ” is a constant, and $\alpha \|w\|_1$ represents L1 regularization. In Scikit Learn, this is set as a hyperparameter with the value 0.001 [28].

For text summarization, the linear regression model helps in selecting important sentences by assigning each sentence a score. These scores are calculated by a weighted sum using each sentence’s feature vector and the feature weights which are learnt by the linear regression algorithm. They can perform feature weighing or finding the importance of features, where the importance of a particular feature increases with its weight.

6 Choice of Regressands

These are the target values to be predicted by the system. They are continuous values in case of regression models although in our case we do not use the regressand. It is, however, crucial for building regression models, and it is the choice of the regressands which has to be predicted by the models. Most of the recent work attempted to predict ROUGE recall scores of individual sentences and some others suggested the use of “Cosine scores” that range between 0.0 and 1.0 [13, 17, 20, 21]. The regressand value is set to “1” or “0” depending on whether the sentence is present in the manual summary or not [12]. In our case, we use the values 1.0 or 0.0, which are assigned by subject experts based on if the sentences are significant or not with respect to inclusion in the system summary [12, 16].

7 Feature Extraction

We used of 13 features to represent the sentences. These features were also used in the supervised ML learning approach presented in [27]. The features chosen are language independent and are structure based or vector based. The features are language independent since they do not use any language-dependent resources nor do they use any language-specific knowledge but use statistics [29]. These statistics are calculated by considering characters, words, sentences in a document and can be calculated for a document in any language, i.e., counting the number of sentences or words or characters in a document for any language.

The following are some notations used in the equations stated below; “ T ” represents a source document. “ P ” is a sentence from a document. The position of sentence “ P ” in document “ T ” is denoted by “ n .” The number of times a term “ m ” appears in a document is given by “ $tf(m)$.” The number of sentences in a document “ T ” is depicted by “ s ,” and “ w ” represents the total number of words in a sentence, and “ c ” is the number of characters in a sentence.

- **LOC_S**: Given by Eq. (4), it measures the position of a sentence to the start of a document [30].

$$\text{LOC}_S = \frac{1}{n} \quad (4)$$

- **LOC_E**: The position of a sentence with respect to the end of a document, illustrated as “ n ” [31], given by Eq. (5)

$$\text{LOC}_E = n \quad (5)$$

- **LOC_B**: This measure, illustrated by Eq. (6), gives the position of a sentence with respect to the borders of a source document [32].

$$\text{LOC}_B = \max\left(\frac{1}{n}, \frac{1}{s - n + 1}\right) \quad (6)$$

- **COUNT_W**: It is the count of all the words in a sentence [33], as depicted in Eq. (7).

$$\text{COUNT}_W = w \quad (7)$$

- **COUNT_CH**: The count of the characters that appear in a sentence [33] is given by Eq. (8).

$$\text{COUNT}_CH = c \quad (8)$$

- **TF**: This measure, denoted by Eq. (9), is obtained by averaging all the “Term frequencies” of every term in a sentence [34].

$$\text{TF} = \frac{\sum_{m \in P} tf(m)}{w} \quad (9)$$

- **TFISF**: It is illustrated by Eq. (10) [35],

$$\text{TFISF} = \sum_{m \in P} tf(m) \times \text{isf}(m) \quad (10)$$

where inverse sentence frequency (ISF), $\text{isf}(m)$ is depicted by Eq. (11),

$$\text{isf}(m) = 1 - \frac{\log(s(m))}{\log(s)} \quad (11)$$

where $s(m)$ is the number of sentences that have the term “ m ” appearing in them.

- **HEADING_O**: It represents the similarity of sentence “ P ” with the document heading “ H ” given by Eq. (12), [30]

$$\text{sim}(P, H) = \frac{|P \cap H|}{\min\{|P|, |H|\}} \quad (12)$$

- **HEADING_J**: It is called as “Jaccard Similarity” calculated for a sentence with respect to the document heading, given by Eq. (13) [30].

$$\text{sim}(P, H) = \frac{|P \cap H|}{|P \cup H|} \quad (13)$$

- **HEADING_C**: Given by Eq. (14) is the “Cosine similarity” of sentence “ P ” with reference to the heading of the document [30].

$$\text{sim}(P, H) = \frac{P \cdot H}{|P||H|} \quad (14)$$

- **T_COMP_O**: Eq. (15) denotes the overlap of the similarity of a sentence “ P ” with the source document’s complement [36].

$$\text{sim}(P, T - P) = \frac{|P \cap (T - P)|}{\min\{|P|, |T - P|\}} \quad (15)$$

- **T_COMP_J**: Eq. (16) represents the “Jaccard Similarity” of a sentence “ P ” with the complement of the source document. [36]

$$\text{sim}(P, T - P) = \frac{|P \cap (T - P)|}{|P \cup (T - P)|} \quad (16)$$

- **T_COMP_C**: Illustrated by Eq. (17) is the “Cosine Similarity” of sentence “ P ” computed with reference to the source document’s complement [36].

$$\text{sim}(P, T - P) = \frac{P \cdot (T - P)}{|P||T - P|} \quad (17)$$

8 Proposed Methodology

This section describes the supervised ATS methodology used for the summarization of Konkani texts using regression in a step-by-step manner. The steps involved in the process are illustrated using a block diagram, shown in Fig. 1.

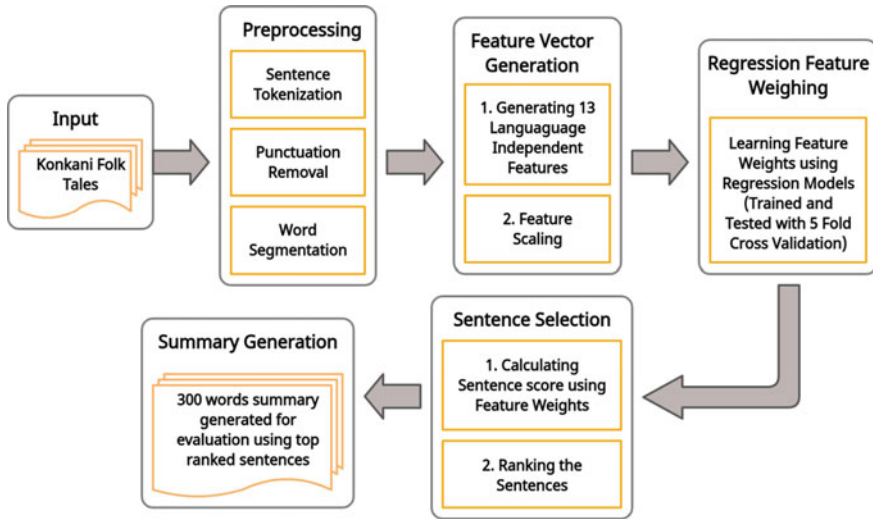


Fig. 1 Supervised ATS methodology block diagram

8.1 Preprocessing

Each sentence is put on a new line. The punctuations are removed from every single sentence from all the documents. These would, however, be preserved in the output summary. The words are separated from the sentences, and then, the cleaned sentences, along with their words, are used for generating feature vectors.

8.2 Feature Vector Generation

Each sentence is translated into a “feature vector.” The sentences in each story are depicted as feature vectors together with their corresponding “regressand” that is set to “1.0” when a sentence is to be included in the output summary or set to “0.0” when a sentence is not to be incorporated in the final summary. Each of the features are then scaled between 0 and 1 which brings all the features on a common scale.

8.3 Regression Feature Weighing

The dataset is composed of 71 folk tales and was divided into five folds. In every fold created, 80% of the documents in each fold were used for training and the rest 20% were used for testing. It was not possible to get 5 equal folds as the dataset has 71 stories. Therefore, the first four folds comprised 14 stories and the fifth fold

comprised 15 stories for testing. Hence, every document in the dataset gets to be a part of the training set as well as the testing set. During the processing of each fold, the linear regression algorithms were trained to learn the feature weights with the help of the linear regression model. This model estimates the value of a dependent variable when a set of independent variables are supplied to it [12, 21]. The linear regression system receives the dependent and the independent vectors as input; thereafter, a result of a constant weighted value is obtained [12]. Mathematically, the model can be depicted as shown in Eq. (18). The coefficient values learned by the algorithms are used to predict a dependent variable “y.” But, we used these coefficient values learnt in the training phase as feature weights to be used in the sentence ranking phase to generate sentence ranking scores.

$$y = W_0 + W_1x_1 + W_2x_2 + \dots + W_nx_n + \varepsilon \tag{18}$$

where

“y” is the dependent variable.

“W₀” is the intercept value.

W₁, W₂, . . . , W_n are the feature weights or coefficients values.

x₁, x₂, . . . , x_n are the independent variables.

“ε” is the error term.

8.4 Sentence Selection

During the testing phase of the folds, the sentence scores for each of the sentences are calculated, and then, the sentences are ranked in decreasing order of their scores. Using the feature weights learnt, the sentence scores are computed using Eq. (19).

$$\begin{pmatrix} R_1 \\ R_2 \\ \vdots \\ R_m \end{pmatrix} = \begin{pmatrix} F_{01} & F_{02} & F_{03} & \dots & F_n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ F_{m1} & F_{m2} & F_{m3} & \dots & F_{mn} \end{pmatrix} \cdot \begin{pmatrix} W_1 \\ W_2 \\ \vdots \\ W_n \end{pmatrix} \tag{19}$$

In Eq. (19),

“R” represents ranking score of each of the m sentences.

“F” is the feature matrix, where each matrix row represents the features of each sentence in a document present in the testing set.

“W” represents the values of weights learnt from the previous step.

8.5 Summary Generation

The sentences of a document are ranked according to their score, and only, the top-ranked sentences are included in the output summary. An upper limit of 300 words is applied to restrict the output summary length. This threshold value is set to 300 words for the summary evaluations since the human-generated summaries are also of 300 words length. Every story in the dataset is summarized by two Konkani language experts. These human-generated summaries are compared using ROUGE toolkit with the system-generated summaries to check for overlap of content.

9 Results and Discussions

We have implemented a system for generating extractive summaries of Konkani folktales using supervised ML approaches using linear regression. The dataset comprises 71 Konkani language texts written in Devanagari script and the summarization system produced extractive summaries for each of the documents. The evaluation of these system-generated summaries was accomplished with the help of ROUGE toolkit [37].

ROUGE toolkit functions very well with human estimates and uses N-gram statistics. The system-generated extractive summaries of 300 words and the corresponding pair of abstractive summaries created by human summarizers are compared with one another to gauge the quality of the system's output [33]. The reason is that the output produced by ATS system needs to closely match the "gold-standard" summaries that humans can generate. If the composition of the words in the system-generated summary is comparable to the human-generated "gold-standard," then it signifies a rather fluent output.

ROUGE metric helps in estimating the overlap of "uni-grams," given by ROUGE-1 scores, "bi-grams" given by ROUGE-2, and "Longest Common Subsequence (LCS)" given by ROUGE-L, between the system-generated and human-generated summaries. The specifics of these metrics are described by Lin [37]. ROUGE-1 "uni-gram" scores are given in Table 1. ROUGE-2 "bi-gram" values are shown in Table 2, and ROUGE-L (LCS) points are illustrated in Table 3. The idea of using these metrics is to measure the fluency of the system-generated summaries across different granularities of text. It can be observed from Tables 1 and 2 that ROUGE-1 scores are greater than ROUGE-2 scores; this is because that there could be an identical term in both these ROUGE metrics; however, the converse may not be true. But, ROUGE-L scores are higher than ROUGE-2 scores since ROUGE-L tracks terms appearing in the sentence in order but not necessarily consecutive. Therefore, ROUGE-L integrates longest common n-grams occurring in a sequence.

The performance of the systems implemented based on linear regression in this paper was evaluated against the performance of a k-means clustering-based system with 3 clusters using the same Konkani dataset [26]. It was also compared with

Table 1 ROUGE-1 uni-gram scores

System	ROUGE-1 (uni-gram)		
	Precision	Recall	<i>F</i> -Score
Ordinary least squares regression	0.32652	0.32594	0.32621
Ridge regression	0.32598	0.32657	0.32625
LASSO regression	0.32769	0.32774	0.32769
K-means clustering with three clusters	0.31408	0.31373	0.31388
Lead baseline	0.30147	0.30165	0.30154
Modified Luhn's heuristic baseline	0.31421	0.31349	0.31383
Human-annotated benchmark	0.35844	0.35460	0.35608

Table 2 ROUGE-2 bi-gram scores

System	ROUGE-2 (bi-gram)		
	Precision	Recall	<i>F</i> -Score
Ordinary least squares regression	0.08450	0.08463	0.08455
Ridge regression	0.08474	0.08489	0.08481
LASSO regression	0.08737	0.08737	0.08736
K-means clustering with three clusters	0.07942	0.07927	0.07934
Lead baseline	0.08097	0.08103	0.08099
Modified Luhn's heuristic baseline	0.07889	0.07876	0.07882
Human-annotated benchmark	0.11088	0.10908	0.10977

Table 3 ROUGE-L LCS scores

System	ROUGE-L (LCS)		
	Precision	Recall	<i>F</i> -Score
Ordinary least squares regression	0.31836	0.31892	0.31862
Ridge regression	0.31851	0.31909	0.31878
LASSO regression	0.31943	0.31948	0.31943
K-means clustering with three clusters	0.30680	0.30644	0.30659
Lead baseline	0.29642	0.29659	0.29648
Modified Luhn's heuristic baseline	0.30822	0.30753	0.30786
Human-annotated benchmark	0.35228	0.34847	0.34994

lead baseline and modified Luhn's heuristic baseline. A lead baseline summary is generated with the first 300 words of a text document. It was used in document understanding conference (DUC). It is a simple baseline yet challenging for text summarizers to outperform [38]. Luhn's heuristic is the earliest statistical method for extractive text summarization [39]. The modified Luhn's heuristic baseline used Luhn's

heuristic method, with YAKE selecting 30 relevant keywords [40]. Sentence scores were computed with the square of the number of keywords divided by the window size. "Window size" is the maximum distance between any two keywords [39]. The highest scoring sentences were then selected to generate 300 words summary.

"Precision" and "Recall" recount the overlap between the system-generated output and the human-generated summaries. "Precision" attempts to identify if the content produced by the system summary is relevant. "Recall" helps in identifying if the contents of human-generated summaries have been captured by the system-generated summaries. "F-score" integrates the precision and recall scores into a single score. All the scores are represented as overlap percentage. We also introduce a human-annotated benchmark to compare the systems, which comprises sentence marked by language experts as significant to be a part of a 300-word summary. The system-generated summaries and the human-annotated summaries are compared and evaluated with the "gold-standard" summaries. The values of ROUGE-1, ROUGE-2, and ROUGE-L are between 0 and 1; a value 0 implies no overlap between system-generated summary and human-generated summary, whereas 1 implies a powerful match between the two. The comparative charts showing the precision, recall, and F-score values for ordinary least squares regression, ridge regression, LASSO regression, k-means clustering with three clusters, lead baseline, modified Luhn's heuristic baseline, and human-annotated benchmark are depicted in Figs. 2, 3, and 4, respectively.

We observed that linear regression-based models could beat the baseline systems; however, they could not beat the human-annotated benchmark. The linear regression-based models could select important sentences in the document using language-independent features, making the approach easily extendable to other low-resource languages. We also noted that ridge regression could improve on the performance

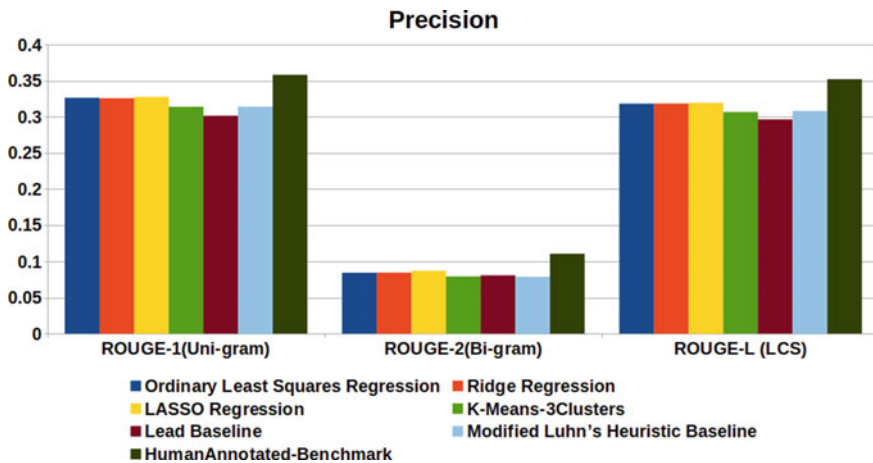


Fig. 2 Comparative chart for precision

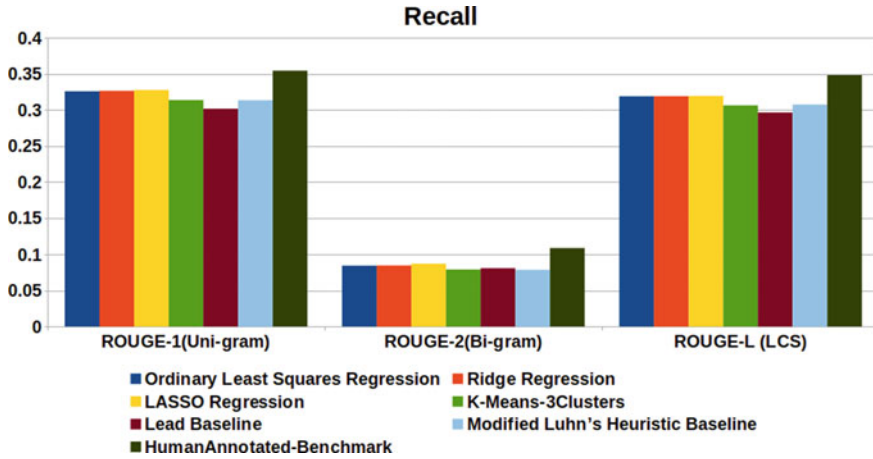


Fig. 3 Comparative chart for recall

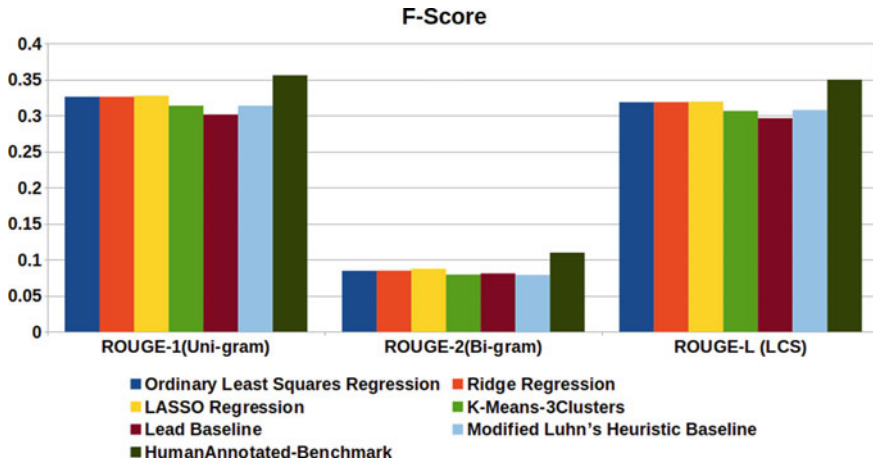


Fig. 4 Comparative chart for F-score

of sentence selection when compared to ordinary least squares. The sentence selection depends on the feature weights learned during the training stage. Unlike ordinary least squares, ridge regression shrinks the values of the feature weights using L2 regularization. This makes the model computationally faster and also reduces the likelihood of over-fitting and allows for a better model fit. LASSO uses L1 regularization which improves on ridge regression as it not only shrinks feature weights but also turns off features that will not be important. This has translated to an even better model fit and, hence, sentence selection. When comparing the systems to the unsupervised ML system, based on k-means method, tested on the same dataset, we see that, being a supervised approach, it can perform better [26].

Using linear regression-based feature weighing along with language-independent features is effective in selecting important sentences from the input document and thus producing an extractive summary.

10 Conclusion

We analyzed the use of supervised linear regression models for the generation of extractive single document summaries in Konkani language. Most of the research in ATS is done on popularly spoken languages in the world, like English. There is a need to extend this research to other less popular languages spoken in the world. Furthermore, the most common genre of the datasets used for such research is news articles, blogs, and reviews. The dataset used in this work belongs to the genre of literature and comprises Konkani language folktales. Folktales have a special significance in the local culture, as the traditions and rituals of a community are passed on to future generations through these stories having a teaching or a moral.

We made use of language-independent features to represent the sentences in the document and used a k-fold evaluation strategy using the ROUGE evaluation toolkit to evaluate the system-generated summaries against the human-generated summaries. The linear regression models are effective at learning feature weights to be used in calculating sentence ranking scores. From our experiments, we observed that our summarization systems, based on linear regression algorithms and language-independent features, could produce acceptable summaries despite not being able to beat our human-annotated extractive benchmark. However, the systems could beat the baselines. In particular, we also observed that using L1 regularization produced better results than L2 regularization and no regularization. Additionally, the language-independent features used can be effortlessly extended to other languages, including low-resource languages, like Konkani, and yet produce promising summaries.

The major limitation of working with a low-resource language, such as Konkani, is the lack of task-specific resources and data. Also, language preprocessing tools such as lemmatizers are not available. Another challenge is the requirement of a human-annotated corpus, for training the ML algorithms, along with human-generated abstracts for evaluation of system-generated summaries.

In future work, the expansion of the size of the dataset can be considered. Other unsupervised approaches, like graph based, semantic based, can be considered for extractive ATS. Also, supervised approaches, including deep learning, can also be explored. Further, we can also consider creating an ensemble of extractive summarizers, which could outperform an individual system.

References

1. Hobson SP, Dorr B, Monz C, Schwartz R (2007) Task-based evaluation of text summarization using relevance prediction. *Inf Process Manag* 43:1482–1499
2. Ling X, Jiang J, He X, Mei Q, Zhai C, Schatz B (2007) Generating gene summaries from biomedical literature: a study of semi-structured summarization. *Inf Process Manage* 43(6):1777–1791
3. Lloret E, Palomar M (2012) Text summarisation in progress: a literature review. *Artif Intell Rev* 37(1):1–41
4. Andhale N, Bewoor LA (2016) An overview of text summarization techniques. In: 2016 International conference on computing communication control and automation (ICCUBEA). IEEE, New York, pp 1–7
5. Moratanch N, Chitrakala S (2017) A survey on extractive text summarization. In: 2017 international conference on computer, communication and signal processing (ICCCSP). IEEE, New York, pp 1–6
6. D'Silva J, Sharma U (2019) Automatic text summarization of Indian languages: a multilingual problem. *J Theor Appl Inf Technol* 97(1). ISSN: 1992-8645
7. Saleh AA, Weigang L (2017) Language independent text summarization of western European languages using shape coding of text elements. In: 2017 13th International conference on natural computation, fuzzy systems and knowledge discovery (ICNC-FSKD). IEEE, New York, pp 2221–2228
8. D'Silva J, Sharma U (2019) Development of a Konkani language dataset for automatic text summarization and its challenges. *Int J Eng Res Technol* 12(10). ISSN: 0974-3154. <https://doi.org/10.5281/zenodo.5531954>
9. Statement—2 (2011) Distribution of population by schedule and other languages India, States and Union Territories—2011. Office of the Registrar General & Census Commissioner, India, Ministry of Home Affairs, Government of India. Available: <https://censusindia.gov.in/2011Census/Language-2011/Statement-2.pdf>. Accessed 15 Nov 2020
10. Kamat K (1997) The origins of the Konkani language. Konkani Heritage Album. Available: <http://www.kamat.com/kalranga/konkani/konkani.htm>
11. Ulrich J, Carenini G, Murray G, Ng R (2009) Regression-based summarization of email conversations. In: Proceedings of the international AAAI conference on web and social media, vol 3(1)
12. Hannah ME, Mukherjee S, Kumar KG (2010) An extractive text summarization based on multivariate approach. In: 2010 3rd International conference on advanced computer theory and engineering (ICACTE), vol 3. IEEE, New York, pp V3–157
13. Xie S, Liu Y (2010) Improving supervised learning for meeting summarization using sampling and regression. *Comput Speech Lang* 24(3):495–514
14. Gupta V, Lehal GS (2011) Features selection and weight learning for Punjabi text summarization. *Int J Eng Trends Technol* 2(2):45–48
15. Aristoteles A, Widarti W, Wibowo ED (2014) Text feature weighting for summarization of documents Bahasa Indonesia by using binary logistic regression algorithm. *Int J Comput Sci Telecommun* 5(7):29–33
16. Kumar A, Yadav J, Rani S (2015) Automatic text summarization using regression model (GA). *Int J Innov Res Comput Commun Eng* 3(5), ISSN (Online): 2320-9801. ISSN (Print): 2320-9798
17. Dlikman A, Last M (2016) Using machine learning methods and linguistic features in single-document extractive summarization. In: Cellier P, Charnois T, Hotho A, Matwin S, Moens M-F, Toussaint Y (eds) Proceedings of DMNLP, workshop at ECML/PKDD, pp 1–8, Riva del Garda, Italy
18. Dlikman A, Litvak M, Last M (2019) Rich feature spaces and regression models in single-document extractive summarization. In: Multilingual text analysis: challenges, models, and approaches, pp 119–154

19. Oliveira H, Lins RD, Lima R, Freitas F, Simske SJ (2017) A regression-based approach using integer linear programming for single-document summarization. In: 2017 IEEE 29th International conference on tools with artificial intelligence (ICTAI). IEEE, New York, pp 270–277
20. Zopf M, Mencía EL, Fürnkranz J (2018) Which scores to predict in sentence regression for text summarization? In: Proceedings of the 2018 Conference of the North American chapter of the association for computational linguistics: human language technologies, vol 1 (Long Papers), pp 1782–1791
21. Khadem S, Ali ZH (2018) Multi-document text summarization based on multiple linear regression. AL-Mansour J 30
22. Rezaei H, Moeinzadeh SA, Shahgholian A, Saraee M (2019) Features in extractive supervised single-document summarization: case of Persian news. arXiv preprint [arXiv:1909.02776](https://arxiv.org/abs/1909.02776)
23. Cagliero L, Quatra ML (2020) Extracting highlights of scientific articles: a supervised summarization approach. Expert Syst Appl 113659. <https://doi.org/10.1016/j.eswa.2020.113659>
24. El-Kassas WS, Salama CR, Rafea A, Mohamed HK (2021) Automatic text summarization: a comprehensive survey. Expert Syst Appl 165:113679
25. Gambhir M, Gupta V (2017) Recent automatic text summarization techniques: a survey. Artif Intell Rev 47(1):1–66. <https://doi.org/10.1007/s10462-016-9475-9>
26. D'Silva J, Sharma U (2020) Unsupervised automatic text summarization of Konkani texts using K-means with Elbow method. Int J Eng Res Technology 13(9):2380–2384. <https://dx.doi.org/10.37624/IJERT/13.9.2020.2380-2384>
27. D'Silva J, Sharma U (2021) Automatic text summarization of Konkani Folk tales using supervised machine learning algorithms and language independent features. IETE J Res (in Press). <https://doi.org/10.1080/03772063.2021.1987993>
28. Pedregosa F et al (2011) Scikit-learn: machine learning in Python. J Mach Learn Res 12:2825–2830
29. Litvak M, Last M (2013) Cross-lingual training of summarization systems using annotated corpora in a foreign language. Inf Retrieval 16(5):629–656
30. Edmundson HP (1969) New methods in automatic extracting. J ACM (JACM) 16(2):264–285
31. Baxendale PB (1958) Machine-made index for technical literature—an experiment. IBM J Res Dev 2(4):354–361
32. Lin CY, Hovy E (1997) Identifying topics by position. In: Fifth conference on applied natural language processing, pp 283–290
33. Nobata C, Sekine S, Murata M, Uchimoto K, Utiyama M, Isahara H (2001) Sentence extraction system assembling multiple evidence. In: NTCIR
34. Vanderwende L, Suzuki H, Brockett C, Nenkova A (2007) Beyond SumBasic: task-focused summarization with sentence simplification and lexical expansion. Inf Process Manage 43(6):1606–1618
35. Neto JL, Santos AD, Kaestner CA, Freitas AA (2000) Generating text summaries through the relative importance of topics. In: Advances in artificial intelligence. Springer, Berlin, Heidelberg, pp 300–309
36. Litvak M, Last M, Friedman M (2010) A new approach to improving multilingual summarization using a genetic algorithm. In: Proceedings of the 48th Annual meeting of the association for computational linguistics, pp 927–936
37. Lin CY (2004) Rouge: a package for automatic evaluation of summaries. In: Text summarization branches out, pp 74–81
38. Nenkova A (2005) Automatic text summarization of newswire: lessons learned from the document understanding conference. In: Proceedings of the 20th National conference on artificial intelligence, vol 3, AAAI'05, pp 1436–1441
39. Luhn HP (1958) The automatic creation of literature abstracts. IBM J Res Dev 2(2):159–165
40. Campos R, Mangaravite V, Pasquali A, Jorge A, Nunes C, Jatowt A (2020) YAKE! Keyword extraction from single documents using multiple local features. Inf Sci 509:257–289