SN Computer Science (2025) 6:527
https://doi.org/10.1007/s42979-025-04045-7

ORIGINAL RESEARCH

Leveraging Sequence-to-Sequence Models for Kannada Abstractive Summarization

Dakshayani Ijeri[1] · **Pushpa B. Patil**[2]

Received: 30 January 2025 / Accepted: 13 May 2025

Abstract

The current scenario in the digital world is overwhelming with the digital data every day in various sectors such as business, healthcare, education, entertainment and many others. The data is available even in Indian regional languages such as Kannada, a Dravidian language spoken by over 50 million people. Text summarization plays a vital role in facilitating efficient information retrieval which saves time and strengthens accessibility by refining complex content into concise, meaningful insights. Text summarization had advanced significantly in global languages, whereas other Indian regional language like Kannada is in limited progress. Minimal development is carried out in this language using extractive method which fails to generate coherent, human-like summaries. The proposed work focuses on Kannada language which is one of the significant languages in India as 20% of Indian population communicate in Kannada as their birth language and it holds 27th rank among top 30 languages across the world. Text summarization can be implemented in two approaches such as extractive method and abstractive method. Extractive summarization method gives the summary by taking out the main sentences from the document and abstractive summarization method gives brief and concise information with respect to the context of the paragraph using new words, phrases, or sentences that may not appear in source text. Proposed work is based on abstractive approach using sequence to sequence model based on Long Short-Term Memory (LSTM). This work presents a novel approach for text summarization in Kannada, making it the first known study to address this problem in this language using abstractive method. The model achieved the accuracy of F1-score of 0.7046 for Rouge-1, 0.5499 for Rouge-2 and 0.7046 for Rouge-L on the dataset of 10,000 documents.

Keywords Abstractive text summarization · Sequence to sequence model · NLP

Introduction

The online universe is bombarded with digital data everyday due to the usage of internet applications by people for wide range of purposes. Hence it has become necessity to process the digital data to enable efficient management, analysis and dissemination of huge data which is beneficial in simplifying everyday task to complex decision making. Both written and spoken forms exhibit variation, from spelling variations and slang to accents and borrowing from other languages. Natural language processing (NLP) addresses this complexity by adding numerical structure to language data, facilitating tasks like speech recognition and text analytics by disambiguating language. Some of the applications of NLP are Text Summarization, Sentiment Analysis, Chabot, Language Translator etc. The existing tools of summarization had evolved periodically to perform on global languages, but there is a scarcity of progress in the Indian regional language

✉ Dakshayani Ijeri
ijeridakshayani@gmail.com

Pushpa B. Patil
pushpapatil2008@gmail.com

[1] Department of Computer Science and Engineering, BLDEA's V. P. Dr. P. G. Halakatti College of Engineering and Technology, (Affiliated to Visvesvaraya Technological University, Belagavi-590018), Vijayapura, Karanataka 586103, India

[2] Department of Computer Science and Engineering (Data Science), BLDEA's V. P. Dr. P. G. Halakatti College of Engineering and Technology, (Affiliated to Visvesvaraya Technological University, Belagavi-590018), Vijayapura, Karanataka 586103, India

Published online: 06 June 2025

like Kannada. Generating a coherent human like summaries is a challenging a task for Kannada language due its complex morphological structure and limited annotated datasets.

Text summarization can be implemented in two ways namely extractive and abstractive methods. Extractive text summarization generates the concise information from vast data by extracting the prime sentences from original data whereas the abstractive summarization generates the brief information from vast data by introducing new relevant sentences in summary exactly resembling the human generated summary. Since English is popular language across the world, NLP has evolved with advanced approaches in various applications. NLP in Indian languages lags due the scarcity of large, annotated dataset and vast diversity across the country. It is complex to process the Indian regional languages due to the presence of many dialects, each having unique grammar, syntax, and morphology. Limited work is being carried out on Kannada language as compared to other Indian languages, as it lacks with computational resources and tools and presence of different variations of same word based on tense, mood, number, and case makes it more complex. It includes a variety of dialects influenced by regions and communities, adding another layer of complexity for both speakers and NLP models. The proposed approach is focused on automatic abstractive text summarization in Kannada language using sequence to sequence model, to the best of our knowledge this abstractive text summarization is the first attempt in Kannada language. So far, minimal research has been carried on Kannada language for text summarization and existing work is based on extractive text summarization. The existing work for text summarization in Kannada language has been experimented with limited dataset and achieved less accuracy.

The proposed model is developed using sequence to sequence model. The model consists of encoder-decoder architecture with Long Short-Term Memory (LSTM) layers to identify relevancy between the sentences, this architecture generates a context vector with a fixed length based on which the decoder generates the output. Root Mean Square Propagation (RMSprop) optimizer is used to overcome the problem of slow convergence and poor performance in training data due to the presence of sparse gradient in data. The model performed with a good accuracy as compared to the existing works in Kannada language.

The contributions of proposed approach are:

- The proposed work presents a novel approach for abstractive text summarization in Kannada language using Sequence to Sequence model based on encoder-decoder LSTM architecture to identify semantic relationship between sentences. The existing models are based on extractive text summarization with limited dataset.
- The proposed work implements RMSprop optimizer is used to overcome the problem of slow convergence and poor performance in training data.
- The proposed work is capable to generate efficient summary for input data of varying length. Existing models works for minimum length of input.

The further paper is organized as follows. Sect. "Related Work" covers the literature survey; Sect. "Proposed Methodology" discusses in detail regarding methodology used for Kannada abstractive summarization and also covers the discussion on algorithm adopted for summarization. Sect. "Results and Discussions" demonstrates the analysis of the performance of a model. Sect. "Conclusion" includes the conclusion of work.

Related Work

In 2025, Sharma and singh [1] proposed a model for extractive text summarization of Indian legal documents by integrating InLegalBERT with K-Means clustering. The input text is converted to high vector embedding utilizing the InLegalBERT method. Similar semantic sentences are grouped and assigned with ranks-based K-Means cluster method. The legal dataset used for this work consists of 7030 legal documents with average 5389 tokens for each document. This model achieved accuracy of 0.3858 with Rouge-L. The model is trained with only Indian legal cases which may not produce promising result on other domain data. The number of clusters are pre-decided during implementation which is not suitable with the varying number of input sentences.

In 2023, Aoteet et. al [2] proposed a model for extractive text summarization in Hindi language using Binary Particle Swarm Optimization with genetic algorithm. The model performs summarization on MDS (Multi-Document Summarization). The dataset consists of Hindi news articles supplied by the Forum of Information Retrieval Evaluation (FIRE). The suggested model combines features, including Title, Sentence Length, Thematic Words, Sentence Position, Proper Nouns, Numerical Data, Term Frequency, and Inverse Sentence Frequency, to provide the desired results. The highest accuracy of 74% was attained in the best results. The model compromises with speed of algorithm to achieve accuracy and hence may not be applicable to real time applications.

In 2023, Anand and Srinivasu [3] had introduced an approach for extractive text summarization on Telugu language using graph-based approach. The count of nouns and verbs dominate the sentence rank. The model is

evaluated with 3000 text documents with average F-score of 0.67.

In 2023, Babu et.al [4] introduced a model for abstractive summarization approach in Telugu language using the Diverse Beam Search algorithm. Telugu newspapers like Andhra bhoomi, Eenadu, and Sakshi and Vaartha are used to create the dataset. This work presents a sequence-to-sequence (seq2seq) encoder-decoder architecture for abstractive Telugu text summarization. Two components make up the seq2seq model an LSTM decoder and a Bi-LSTM encoder. The Diverse Beam Search approach increases the diversity of the summary using pointer generator and attention mechanisms, the work is inadequate when producing summaries with longer sentences. This model achieves an accuracy of 37%. This work is experimented with input data of average 10 sentences and generates results with 2 sentences.

In 2023, Pawar et.al [5] introduced a model for extractive text summarization in English language using sentence clustering and word graph approach. The clustering method involves lemmatization, Jaccard similarity and Euclidean distance for sentence clustering. Later word graph technique is used to synthesize the summary based on various clusters. The proposed efficiency and significant reduction in redundancy of the findings are outstanding in word graph technique. The model achieved an average accuracy of 55% with a sentence clustering approach. The accuracy could have been justified if dataset size was discussed.

In 2022, Pallavi et.al [6] introduced an approach for extractive text summarization in Kannada language using sentence ranking algorithm. This model is experimented on input with 20 sentences and 256 words and generates results with 90 words. The discussion would have been more helpful if provided with dataset size and accuracy.

In 2022, Rao et.al [7] proposed a model extractive text summarization in Telugu language. This approach uses three methods Genetic algorithm (GA), Particle swarm optimization and Heuristic search (HS). The preferred methodology demonstrates a significant F-measurement outcome over the generative GA and HS. This model achieves the consistency and continuity of the summary. The dataset is built using Telugu newspaper. The Particle Swarm Optimization (PSO) method uses the position of the particles and converges rapidly to the optimal solution, in contrast Harmony Search (HS) or Genetic Algorithm (GA), which don't operate this way. The proposed technique is evaluated using Telugu documents and the ROUGE tool. Overall, the PSO approach demonstrated higher efficiency. This model achieves an accuracy rate of 58%.

In 2022, Durga et al. [8] proposed a model for the extractive summarization for Telugu language using the grasshopper optimization algorithm based on Histo fuzzy c-means median support. A set of internet research datasets is used to assess the proposed system. The documents are grouped using Histo-Fuzzy C-means Clustering, and the rank of sentences are determined by their weights. To provide a concise and lucid summary, the Median Support Based Grasshopper Optimization Algorithm (MSGOA) is utilized. The Grasshopper Optimization Algorithm produces better outcomes as the number of iterations is more but as the number of iterations increase then it takes a more time to generate the result. This model demonstrates impressive performance, attaining an accuracy of 84%.

In 2022, Agarwal et.al [9] introduced a model for abstractive summarization in Hindi language using IndicBart which is specifically fine-tuned for productive text generation. The dataset is created from pairs of articles and headlines sourced from various Hindi newspapers. With a left-to-right autoregressive decoder and a bidirectional encoder to process the distorted text, BART functions as a sequence-to-sequence model. The system achieves 55% accuracy. This approach is limited to generate 75 tokens in summary.

In 2022, Kumar, et.al [10] developed a model based on extractive text summarization for multi-document using firefly algorithm. The suggested model considers the topic relation, readability and cohesion factor. The behavior of fireflies serves as the foundation for the Firefly algorithm. The Firefly-based text summarization (FbTS) technique requires parameter adjustment that includes the number of repetitions. This model attained accuracy of 48%. The dataset considered for this approach is 1900 documents.

In 2022, Almarjeh et.al [11] introduced a model for Arabic abstractive text summarization. This approach was implemented using transformer-based and RNN-based architectures for summarization. The dataset considered here is 75,702 examples with 33 average number of words in each sample. This approach results in single sentence unreliable summary and model works on modern standard Arabic language and does not perform well on dialectic Arabic. The accuracy of system is 49%.

In 2021, Jayashree et.al [12] introduced an extractive text summarization for Kannada language based on POS tagging. The sentence ranking algorithm is utilized and achieved average accuracy of 0.607. The dataset considered here is of 8 articles with 10 lines each. The POS tagging mechanism sometimes miss classifies the POS words as stop words.

In 2021, Jayan and Govindaru [13] developed a model of text summarization on Malayalam language. This model is based on Latent Semantic Analysis (LSA) and Text Rank algorithms and achieved F-score of 66.99 for LSA and 62.32 for Text Rank, these scores are based the unigrams matched between machine generated summary and reference summary. F-score calculation would have been more justifiable if data size information is available.

In 2021, Karmakar et.al. [14] presented two deep learning architectures—a Stacked LSTM-based Sequence-to-Sequence (Seq2Seq) Neural Network and an Attention-based strategy— for abstractive text summarization in Hindi and Marathi languages. Lists of uncommon terms and stop-words in Marathi and Hindi that were prepared for preprocessing are used to assist these models. Due to the scarcity of regional language datasets, the information was collected from various sources, such as literature, newspapers, and carefully selected datasets. Hindi consists of approximately 2lakh samples with 300 length and Marathi dataset consists of 100 news articles. The LSTM-attention model proved to be more effective than the LSTM model. The proposed Attention LSTM model showed convergence with very low training and test loss, although it required longer training time. The accuracy of system in 63.8%.

In 2020, Jayashree and Vinay [15] introduced an approach for Kannada extractive text summarization using Jaccard's similarity score with 40 documents in different domains. The flow of information in generated summary may have sentences out of chronological order. It would have been helpful if the article consists detailed information regarding data size and accuracy.

In 2019, Zakiet.al. [16] proposed a novel approach by integrating reinforcement learning with seq2seq models. This model was applied across multiple datasets in different languages, including English and Arabic. This preprocessing technique, named advanced cleaning, enhances the relevance of vocabulary in the dataset, thereby improving text summarization efficiency without altering the models themselves. This technique was specifically applied to the Arabic dataset with 20 K samples, but can be adapted for other agglutinative languages too. The model generates correct summary with short text and also requires the length of input and output to be fixed.

In 2015, Dave et.al. [17] Presented a novel approach utilizing Word Net ontology to generate abstractive summaries from extractive summaries. The outcomes of the experiments show that the summaries that are produced are clear, brief, and grammatically correct. The English dataset used was sourced from the Kaggle website. The model is experimented with 5 multiple documents. The model generated summary is compared with human generated summary. However, as the document size increases, the system's processing time also increases. It would have been more helpful if work has been discussed with accuracy and performance evaluation parameters.

In 2015, Khan et.al. [18] proposed a model for abstractive summarization using genetic semantic graph approach. Semantic role labeling (SRL) is used to extract predicate argument structures (PASs) from the document collection, which is then used to produce the semantic graph, where nodes represent PASs. Semantic similarity weights derived from PAS-to-PAS and PAS-to- document set relationships are represented by edges in the graph. A genetic algorithm is used to optimize and weight these associations, represented by various attributes. A modified graph-based ranking technique is used to rank the salient PASs. The top- ranked PASs are used to construct summary sentences using language creation, and maximal marginal relevance (MMR)is used to re-rank the PASs to decrease redundancy. The DUC- 2002 dataset with 59 news articles are used and achieved accuracy of F-Score 0.6094.

InLegalBERT with K-Means clustering model is experimented on Indian legal documents for extractive summarization, this model is trained on Indian legal documents making it domain specific model and has a fixed number of clusters, due to which model may face difficulty in generating summary for other domains and varying input size [1]. A model is proposed for text summarization in Kannada language using extractive approach with a limited dataset [6, 12, 15]. Model sometimes misclassified the POS words as stop words, as POS words are composite of verbs (action words) and nouns (subjects) which are key features for summary [12]. Ignoring these may result in semantic inconsistent summary. The generated summary may not follow the proper order of sentences [15], this leads to summary with sentences which are not arranged with chronological order. The models worked on Telugu language [3, 4, 7, 8]. Noun and verb count dominate the sentence rank [3], here algorithm may assign higher ranks to the sentences with a greater number of nouns and verbs whereas not all the sentences with larger number of nouns and verbs are embedded with significant information. This approach used abstractive approach with limited dataset size and achieved less accuracy [4]. The work utilizes a greater number of iterations to perform better results and increases computation cost and time [8], this also leads model to memorize training data and affects on unseen data. Hindi was the prime language used in [2] using extractive approach, [9, 14] using abstractive approach. The model compromises with speed of algorithm to achieve accuracy and hence may not be applicable to real time applications [2]. This approach is limited to generate 75 tokens in summary [9], may result in summary by omitting supporting information and lead to underspecified summary.

The proposed model is experimented with larger dataset as compared to existing work, this model is not biased towards the count of nouns and verbs. The proposed approach maintains the proper chronological order between the sentences in summary. Model achieved promising results with less epochs and does not put restrictions on number of tokens considered for summary.

Proposed Methodology

The proposed model is implemented using sequence to sequence model based on Long Short-Term Memory (LSTM) encoder-decoder with attention mechanism.

Sequence-to-Sequence Model for Kannada Abstractive Text Summarization

Figure 1 demonstrates the operational process of Kannada Abstractive Text Summarization Model. The model is composed of two prime components namely encoder and decoder. The sequence of input is executed by the encoder and generates the context vector with compression, this context vector is comprised of significant details from the input sequence. A particular kind of recurrent neural network which is suitable for processing of sequential data known as LSTM is used in encoder. The result of decoder is dependent on context vector generated by the encoder. The attention mechanism is used along with LSTM in to guide the decoder in considering the relevant segments of the input to have semantic relation while generating output.

Preprocessing

The following section discusses the procedure of text preprocessing that will be applied on the input data.

Removal of punctuation and special characters

The input text data has to undergo the cleaning process in order to exclude the irrelevant content which converts the data into suitable form required for language processing to generate summary. This process involves removal of punctuation marks, special symbols, and digits that are not relevant to the task. Some of the punctuation marks in Kannada include full stops (.), commas (,), question marks (?), exclamation marks (!), colons (:), and semicolons (;). Removal of these punctuation marks will refine the text with leaving words intact, for example ವಾಹ್! ಎಷ್ಟು ಚೆಲುವು! (Wow! How beautiful!) is converted to ವಾಹ್ ಎಷ್ಟು ಚೆಲುವು (Wow How beautiful) after removing punctuation marks.

Tokenization

The process of dividing text data into smaller pieces known as tokens is known as tokenization. Depending on the application, these tokens can be words, characters, or even sentences. The goal of tokenization is to prepare the text data for further processing.

Stop words removal

In any language, stop words are words that don't add much to a sentence and can be ignored without changing its meaning. Removing stop words reduces the size of the dataset and the amount of time it takes to train the model without significantly affecting the model's accuracy. E.g., "ನಾನು"(me), "ನೋನು" (you)", "ಇದು(this)", "ಆದರೆ(but)".

Stemming

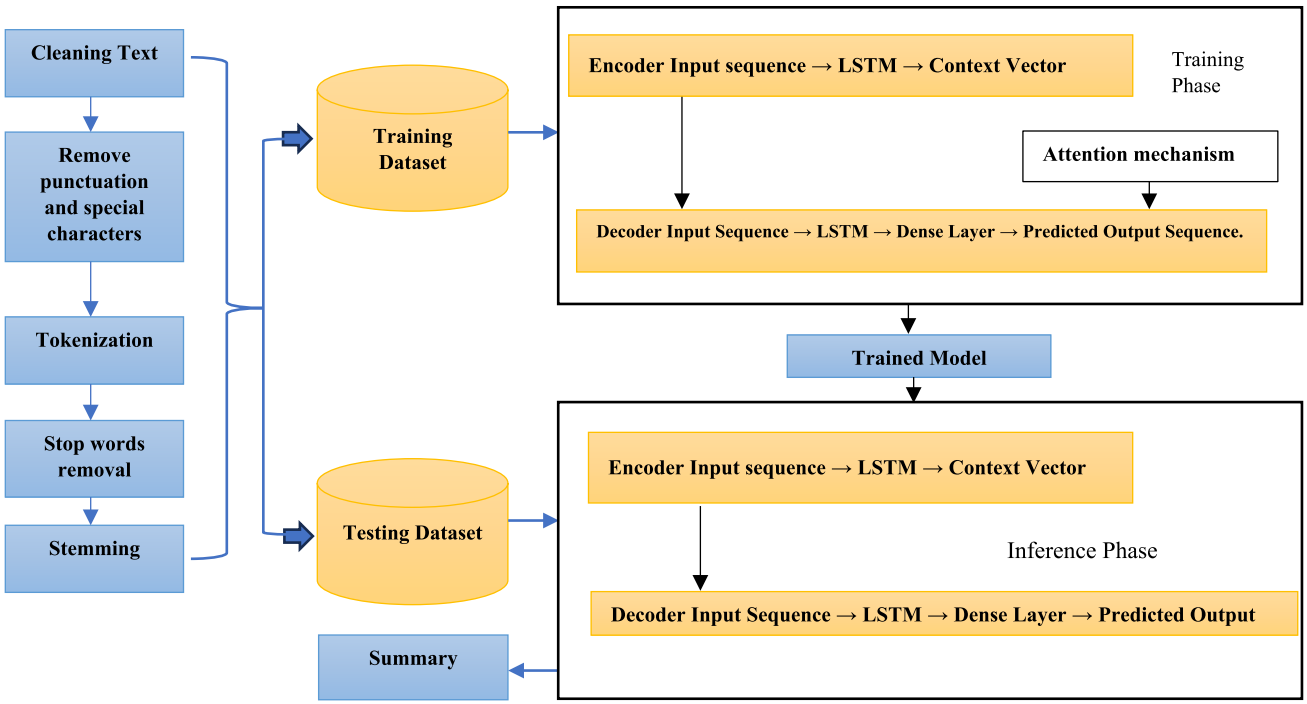

Fig. 1 Sequence to sequence model for Kannada abstractive text summarization

Stemming is essentially taking a word and reducing it to its core word. For instance, the word "Eating" has the suffix "ing." If we take "ing" out of "Eating," we obtain the base word, or root word, which is" Eat."For example: "ನೈಸರ್ಗಿಕವಾಗ"(Naturally) is reduced to its root word "ನೈಸರ್ಗಿಕ "(Natural). Stemming increases the density of the training data in machine learning- based natural language processing.

Encoder-Decoder Architecture

The proposed model leverages the encoder-decoder architecture used for Kannada abstractive text summarization utilizing a sequence-to-sequence (seq2seq) model with Long Short-Term Memory (LSTM) networks. The model is divided into two primary components: the encoder and the decoder as shown in Fig. 2.

In the encoder section, the input sequence, consider the example "ಬೆಂಗಳೂರು ನಗರದಲ್ಲಿ ಇತ್ತೀಚೆಗೆ ಬೃಹತ್ ಮಳೆಗೆ ತೀವ್ರ ಬದಲಾವಣೆಗಳಾಗಿವೆ" ("Bengaluru city has seen drastic changes recently due to heavy rains") which undergoes preprocessing and get reduced to tokens with removal of punctuation marks and stop words such as [ಬೆಂಗಳೂರು, ನಗರದಲ್ಲಿ, ಮಳೆ ಗೆ, ತೀವ್ರ, ಬದಲಾವಣೆಗಳಾಗಿವೆ] termed as (a1, a2, a3..a5) in the Fig. 2 which represents the original text that needs to be summarized. This sequence is fed sequentially into a series of LSTM cells. Each LSTM cell processes its respective input token (ai) along with the hidden state (hi-1) from the previous cell, producing a new hidden state (hi). The first LSTM cell takes the first input (a1) and an initial hidden state to generate (h1). This process continues with the second LSTM cell taking (a2) and (h1) to produce (h2), and the third cell taking (a3) and (h2) to produce (h3). The final hidden state (h5), known as the context vector, encapsulates the contextual information of the entire input sequence.

In the decoder section, the context vector serves as the initial hidden state for the decoder LSTM cells. The decoder generates the summarized text by producing output tokens one at a time. Initially, the encoder vector and an initial input (often a start token) are fed into the first LSTM cell of the decoder to generate the first output token (b1). This output token or a subsequent input is then fed into the next LSTM cell to generate the next output token, and this process continues to form the complete summarized text.

This seq2seq model with LSTM networks is effective for abstractive text summarization as it can handle varying lengths of input and output sequences, capturing the essential information from the input text and generating a coherent summary.

LSTM Architecture

The input text is processed with various gates in the LSTM cell which manages the information flow as shown in Fig. 3. The decision regarding the retention or discard of the information from previous state is done by first gate known as forget gate (depicted with a first orange hexagon labeled δ) as per Eq. (1). The current text from the input and previous hidden state will influence decision of forget gate. The current input state and previous hidden state will navigate through the input gate (second orange hexagon labeled δ) and it is the responsibility of input gate to identify the correct information from current data that has to be included into the cell state.

$$f_g = \sigma(Wt_f * \left[hid_{t-1}, input_{cur}\right] + bv_f \quad (1)$$

where.

f_g: forget gate output.

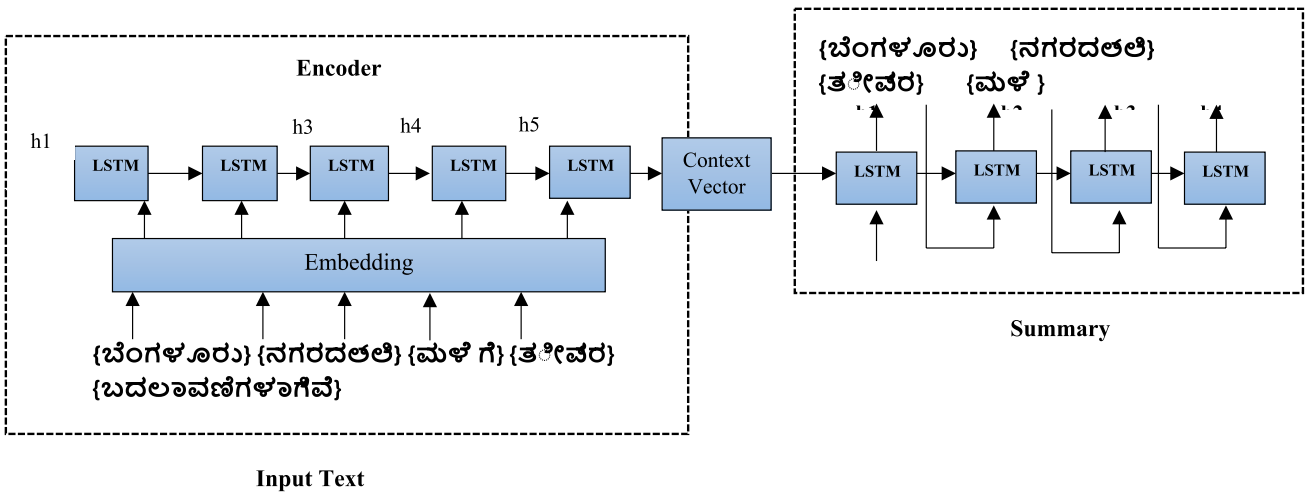

Fig 2. Encoder-Decoder Architecture

Fig. 2 Encoder-Decoder Architecture

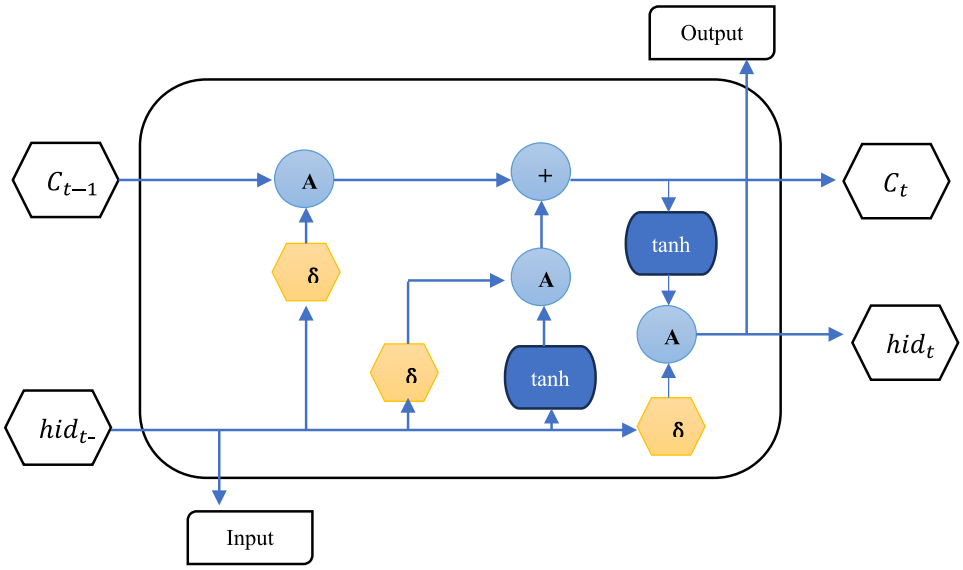

Fig. 3 LSTM Architecture for text summarization

Wt_f: weight matrix for forget gate.
hid_{t-1}: previous hidden state.
$input_{cur}$: current input.
bv_f: bias vector for the forget gate.
σ: sigmoid activation function.

The output of input gate is modulated by applying tanh function that results in candidate values required for updating cell state as per Eq. (2) and Eq. (3). Point wise operation (represented by a circle with a plus sign) is operation is applied between candidate values and output of forget gate. This generates the updated cell state Ci-1 by integrating both the archived data from the previous state and the new relevant information as per Eq. (4). The hidden state is concluded with help of output gate (third δ gate) as per Eq. (5). The process continues the updated cell state Ci and the new hidden state hi on the next iteration. This mechanism of analyzing and integration between the current data and previous information facilitates the LSTM network to capture and manage the long-range dependencies in the data, helping it particularly well-suited for tasks like abstractive text summarization, where understanding context and sequential information is crucial.

$$ig_o = \sigma(Wt_i * \left[hid_{t-1}, input_{cur}\right] + bv_i) \tag{2}$$

$$C\}_t = \tanh(Wt_c * \left[hid_{t-1}, input_{cur}\right] + bv_c) \tag{3}$$

where.

ig_o: input gate output.
Wt_i: weight matrix of input gate.
$C\}_t$: candidate cell state.
Wt_c: weight matrix for candidate values.
bv_i, bv_c: bias vectors for input gates and candidate values.
tanh: tangent activation function

$$C_t = f_g * C_{t-1} + ig_o * C\}_t \tag{4}$$

where.

C_t: new cell state.
C_{t-1}: previous cell state

$$Og_o = \sigma(Wt_0 * \left[hid_{t-1}, input_{cur}\right] + bv_0 \tag{5}$$

$$hid_t = Og_o * \tanh(C_t) \tag{6}$$

where.

Og_o: output gate output.
Wt_0: weight matrix for the output gate.
bv_0: bias vector for the output gate.
hid_t: new hidden state.

Kannada Abstractive Text Summarization Algorithm

The following steps demonstrates the flow of execution for abstractive text summarization on Kannada language using sequence to sequence model.

```
Step 1:  Preprocessing Steps
        Input: Kannada Text (document)
        Output: Tokens (separated words)

        preprocess (input text):
                Break the sentence into tokens.
                Delete stop words from tokens.
                Apply stemming to generate root words
                Return processed_tokens

Step 2: Encoder
        Input: processed_tokens
        Output: Context Vector generated from hidden states

        function construct_encoder (processed_tokens):
                for tokens of each sentence:
                        Initialize Embedding Layer to convert token indices to embeddings.
                        Navigate the embedded sequence through LSTM layer.
                        Record the final hidden states and cell states of the LSTM.
                Return the context vector generated from hidden state and cell state.

Step 3: Decoder
        Input: Encoder context vector, previous token.
        Output: summary.

        function Construct_decoder(context vector):
                for each decoder input tokens:
                        Initialize Embedding Layer.
                        Apply LSTM
                        Apply Relu activation function to finalize the next token prediction.
                Return predicted token and updated hidden state for next decoding step.

Step 4: Seq2Seq Model
        Integrate the encoder and decoder into a sequence-to-sequence model.
        Input: A stream of tokens (input sentence in Kannada).
        Output: A stream of tokens (abstractive summary).

        function construct_Seq2Seq_Model(input_stream):
                for each sentence in input_stream:
                        navigate input_sequence through the encoder to get encoder state.
                        Use the encoder state as the initial state for the decoder.
                        for each token in sentence:
                                Iteratively pass tokens through the decoder to generate the summary.
                Return the generated sequence (summary).

Step 5:  Training
        Input: Pairs of input text and target summary (both tokenized).
        Output: Trained Seq2Seq model.

        function Train_Seq2Seq_model(input_texts, target_summaries):
                for each training pair (input_text, target_summary):
                        Preprocess input_text and target_summary.
                        Use the Seq2Seq model to generate predicted summaries.
                        Calculate loss using sparse categorical cross-entropy.
                        Perform backpropagation to update model weights.
                        Repeat for multiple epochs until convergence.

Step 6. Summarization (Inference)
        Input: A new input sentence in Kannada.
        Output: Abstractive summary of the input sentence.

        function Summarize(input_text):
                Preprocess the input_text.
                Pass the preprocessed text through the encoder to get the context vector.
                Initialize the decoder with the context vector and start generating tokens:
                        for each iteration:
                                Use the previous token and decoder's hidden state to generate the next token.
                                Update the hidden state.
                        Stop generation when an end-of-sequence token is produced or a max length is reached.
                        Detokenize the generated tokens to produce the final summary.
                Return final_summary
```

Results and Discussions

The primary objective of the proposed model is to assess the performance of sequence-to-sequence model based on LSTM for abstractive text summarization on Kannada language. The details of performance of the model are discussed as follows.

About Dataset

The Kannada dataset is created by collecting more than 10,000 news articles from various newspaper websites such as dailyhunt, vijayavani, prajavani etc. Dataset is constructed in CSV format which contains two columns such as article title and article content as shown in Table 1 and Table 2 consists the English translation of Table 1. The article_content column contains the input text to be summarized whereas article_title column contains the label in the form of short summary for the article_content data. Dataset is divided into 80% of training data and 20% of testing data. The dataset is comprised of data with various domains such as politics, sports, education etc. The model learns by recognizing patterns in the training data, such as which words and phrases are important for conveying the main idea of a document. The model's performance on unobserved data is assessed using testing data. It consists of Kannada text documents that the model has not been exposed to before. The model generates summaries for the testing data, and they are compared against human-written summaries to evaluate the model's correctness.

Performance Measure

The proposed model is assessed by using ROUGE (Recall-Oriented Understudy for Gisting Evaluation) score. The comparison of the words or sequence of words between the model generated summary and reference summary is evaluated using ROUGE. The most common variants are ROUGE-N, ROUGE-L, and ROUGE-S. Equations for evaluating ROUGE-1 and ROUGE-2 scores are discussed as follows.

ROUGE-1

ROUGE-1 is used to generate a value based on number of unigrams matched between the model generated summary and reference summary, precision is the measure representing the percentage of relevant unigrams retrieved as per model summary according to Eq. (7) where as recall is

Table 1 Sample Kannada dataset for training

article_title	article_content
ಆಸೀಸ್ ವಿರುದ್ಧದ ಟೆಸ್ಟ್ ಪಂದ್ಯ ಸೋಲಿನ ಬೆನ್ನಲ್ಲೇ ಪಾಕ್‌ಗೆ ಡಬಲ್ ಶಾಕ್ ನೀಡಿದ ಐಸಿಸಿ	"ಪರ್ತ್: ಇಲ್ಲಿನ ಪರ್ತ್ ಸ್ಟೇಡಿಯಂನಲ್ಲಿ ನಡೆದ ಆಸ್ಟ್ರೇಲಿಯಾ ಹಾಗೂ ಪಾಕಿಸ್ತಾನ ನಡುವಿನ ಮೊದಲ ಟೆಸ್ಟ್ ಪಂದ್ಯದಲ್ಲಿ ಕಾಂಗರೂ ಪಡೆ ಗೆದ್ದು ಬೀಗಿದ್ದು, ಪಾಕ್‌ಗೆ 360 ರನ್‌ಗಳ ಸೋಲುಣಿಸಿದೆ. ಮೊದಲ ಟೆಸ್ಟ್ ಪಂದ್ಯದಲ್ಲಿ ಸೋಲಿನ ಬೆನ್ನಲ್ಲೇ ಐಸಿಸಿ ಪಾಕಿಸ್ತಾನಕ್ಕೆ ಡಬಲ್ ಶಾಕ್ ನೀಡಿದೆ. 2025ರಲ್ಲಿ ನಡೆಯುವ ವಿಶ್ವ ಚಾಂಪಿಯನ್ ಟೆಸ್ಟ್ ಚಾಂಪಿಯನ್‌ಶಿಪ್‌ನ ದೃಷ್ಟಿಕೋನದಿಂದ ಪಾಕಿಸ್ತಾನಕ್ಕೆ ಆಸ್ಟ್ರೇಲಿಯಾ ವಿರುದ್ಧದ ಗೆಲುವು ಅತ್ಯಗತ್ಯವಾಗಿತ್ತು. ಆದರೆ, ಶಾನ್ ಮಸೂದ್ ಪಡೆ ಕಾಂಗರೂಗಳ ವಿರುದ್ಧ ನಿರೀಕ್ಷಿತ ಪ್ರದರ್ಶನ ನೀಡಲು ವಿಫಲವಾಗಿದೆ. ಸೋಲಿನ ಹೊಡೆತದಿಂದ ಕಂಗೆಟ್ಟಿರುವ ಪಾಕ್‌ಗೆ ಐಸಿಸಿ ದಂಡ ವಿಧಿಸಿರುವುದಲ್ಲದೇ ಎರಡು ಅಂಕವನ್ನು ಕಡಿತಗೊಳಿಸಿದೆ. ನಿದಾನಗತಿಯ ಬೌಲಿಂಗ್ ಮಾಡಿದ ಪಾಕಿಸ್ತಾನದ ಆಟಗಾರರಿಗೆ ಐಸಿಸಿ ಪಂದ್ಯ ಶುಲ್ಕದ ಶೇಕಡಾ 10ರಷ್ಟು ದಂಡ ವಿಧಿಸಿದೆ.....
ರಾಜ್ಯದಲ್ಲಿ ಕಳೆದ ಒಂದು ವಾರದಲ್ಲಿ 81 ಕೋವಿಡ್ ಪ್ರಕರಣಗಳು ಪತ್ತೆ	"ಬೆಂಗಳೂರು : ನೆರೆಯ ಕೇರಳದಲ್ಲಿ ಕೋವಿಡ್ ರೂಪಾಂತರ ತಳಿ ಪತ್ತೆ ಬೆನ್ನಲ್ಲೇ ರಾಜ್ಯದಲ್ಲಿ ಕೋವಿಡ್ ಪ್ರಕರಣಗಳ ಸಂಖ್ಯೆ ಹೆಚ್ಚುತ್ತಿದ್ದು, ಕಳೆದ ಒಂದು ವಾರದಲ್ಲಿ 81 ಸೋಂಕು ಪ್ರಕರಣಗಳು ದೃಢಪಟ್ಟಿವೆ. ಡಿ. 12 ರಿಂದ 18ವರೆಗೆ ರಾಜ್ಯದಲ್ಲಿ ಕೋವಿಡ್ ಸೋಂಕು ಲಕ್ಷಣ ಹೊಂದಿರುವ 2,619 ಶಂಕಿತರ ಸ್ವಾಬ್ ಮಾದರಿ ಸಂಗ್ರಹಿಸಿ ಪರೀಕ್ಷೆ ನಡೆಸಲಾಗಿದ್ದು, ಈ ಪೈಕಿ 81 ಜನರು ಸೋಂಕಿಗೆ ಒಳಗಾಗಿದ್ದಾರೆ. ಈ ಅವಧಿಯಲ್ಲಿ ತೀವ್ರ ಉಸಿರಾಟದ ಸಮಸ್ಯೆ (ಸಾರಿ)ಯಿಂದ ಬಳುತ್ತಿರುವ 156 ಆಸ್ಪತ್ರೆಗಳಲ್ಲಿ ಚಿಕಿತ್ಸೆ ಪಡೆಯುತ್ತಿದ್ದಾರೆ. ಆದರೆ ರಾಜ್ಯದಲ್ಲಿ ಈವರೆಗೂ ಹೊಸ ರೂಪಾಂತರ ತಳಿ ಪತ್ತೆಯಾಗಿಲ್ಲ.....

Table 2 English Sample Dataset for training

article_title	article_content
"ICC gives Pakistan a double shock after Test defeat against Australia."	"Perth: The Kangaroos defeated Pakistan by 360 runs in the first Test between Australia and Pakistan at the Perth Stadium in Perth on Sunday. After losing the first Test, the ICC gave a double blow to Pakistan. A win over Australia was a must for Pakistan from the point of view of the 2025 World Champions Test Championship. However, Shan Masood's men failed to perform as expected against the Kangaroos. The ICC has imposed a fine on Pakistan and reduced them by two points. Pakistan players have been fined 10 per cent of their match fees by the ICC for slow bowling......"
"81 Covid cases detected in the state in the last one week."	" Bengaluru: With the detection of the Covid-19 variant in neighbouring Kerala, the number of COVID-19 cases in the state has been on the rise with 81 confirmed cases in the last one week. D. Swab samples of 2,619 suspected covid-19 patients were collected and tested in the state from June 12 to 18, out of which 81 were infected. During this period, he was undergoing treatment at 156 hospitals for acute respiratory distress syndrome (SARI). However, no new variant has been detected in the state so far.....".

the measure representing the percentage of retrieved unigrams that are relevant as per reference summary according to Eq. (8).

$$Precision_{Rouge}1 = \frac{Numberofoverlappingunigrams}{Totalnumberofunigramsinmodelsummary} \quad (7)$$

$$Recall_{Rouge}1 = \frac{Numberofoverlappingunigrams}{TotalnumberofunigramsinReferencesummary} \quad (8)$$

F1-Score is the harmonic mean of precision and recall according to Eq. (9).

$$F1 - Score_{Rouge}1 = \frac{2 * Precision_{Rouge1} * Recall_{Rouge1}}{Precision_{Rouge1} + Recall_{Rouge1}} \quad (9)$$

ROUGE-2

ROUGE-2 is used to generate a value based on number of bigrams matched between the model generated summary and reference summary, precision is the measure representing the percentage of relevant bigrams retrieved as per model summary according to Eq. (10) where as recall is the measure representing the percentage of retrieved bigrams that are relevant as per reference summary according to (11) and F1-Score is computed using Eq. (12).

$$Precision_{Rouge}2 = \frac{Numberofoverlappingbigrams}{Totalnumberofbigramsinmodelsummary} \quad (10)$$

$$Recall_{Rouge}2 = \frac{Numberofoverlappingbigrams}{TotalnumberofbigramsinReferencesummary} \quad (11)$$

$$F1 - Score_{Rouge}2 = \frac{2 * Precision_{Rouge2} * Recall_{Rouge2}}{Precision_{Rouge2} + Recall_{Rouge2}} \quad (12)$$

ROUGE-L

ROUGE-L is used to generate a value based on number of Longest Aligned Sequence (LAS) matched between the model generated summary and reference summary. The Precision, Recall and F1-Score is computed as per Eq. (13), Eq. (14) and Eq. (15).

$$Precision_{Rouge}L = \frac{LAS}{Totalnumberofwordsinmodelsummary} \quad (13)$$

$$Recall_{Rouge}L = \frac{LAS}{TotalnumberofwordsinReferencesummary} \quad (14)$$

$$F1 - Score_{Rouge}L = \frac{2 * Precision_{RougeL} * Recall_{RougeL}}{Precision_{RougeL} + Recall_{RougeL}} \quad (15)$$

Word Length Distribution in Text Data

Figure 4 illustrates the distribution of text lengths within a dataset, presenting a visual representation of how many texts fall into various word count ranges. On the x-axis, you'll find the range of word counts, while the y-axis measures the frequency count of text falling within each range. Each bar on the histogram corresponds to a specific range of word counts, known as bins. The histogram suggests that there's a higher concentration of texts with shorter lengths, typically ranging between 200 to 400 words. Conversely, there are fewer texts with longer lengths, particularly those exceeding 600 to 800 words. Additionally, it's notable that there are very few instances of texts with either no words or an extensive word count surpassing 800. In essence, the histogram provides insights into the distribution of text lengths within the dataset, highlighting patterns and trends in text length frequencies.

Training and Testing loss of a model

Figure 5 illustrates the line graph showing the training and testing loss of a model. Epochs, or iterations throughout the whole training dataset, are represented by the X- axis. The Y-axis displays loss, a numerical figure that shows how well a model works in relation to a certain job. Better performance is indicated by a lower loss.

Train: The model's loss on the training set of data is shown by the blue line. The training loss usually decreases as the number of epochs rises since the model gains knowledge from the data.

Test: The model's loss on the testing data is shown by the orange line. As the model gains knowledge, the test loss ought to decrease as well. But over fitting may be indicated if the test loss begins to rise after a specific number of epochs.

Hyperparameters

The proposed sequence-to-sequence model for Kannada text summarization includes the hyperparameters such as epochs, batch size, latent dim and embedding dim and Table 3 details the values that are set during the implementation of this model.

Epochs are the number of cycles executed on the training data to allow the model to learn from the training data while reducing the loss and avoiding the overfit. 25 epoch cycles are used in this proposed model, beyond this value the model did not perform well significantly and also led to overfitting. This made model to behave differently with loss on the training set and the loss on the validation set over the time. Batch size indicates the number of samples processed during one cycle which affects gradient updates, convergence stability, and computational efficiency and this model performed well with batch size of 32. This model consumed larger training time with smaller batch sizes (e.g., 20) and led to noisier gradient updates, while larger sizes (e.g., 64) led to memory overflow. Smooth training curves and better ROUGE-L scores were achieved with batch size 32. The model's capability to extract the features is determined

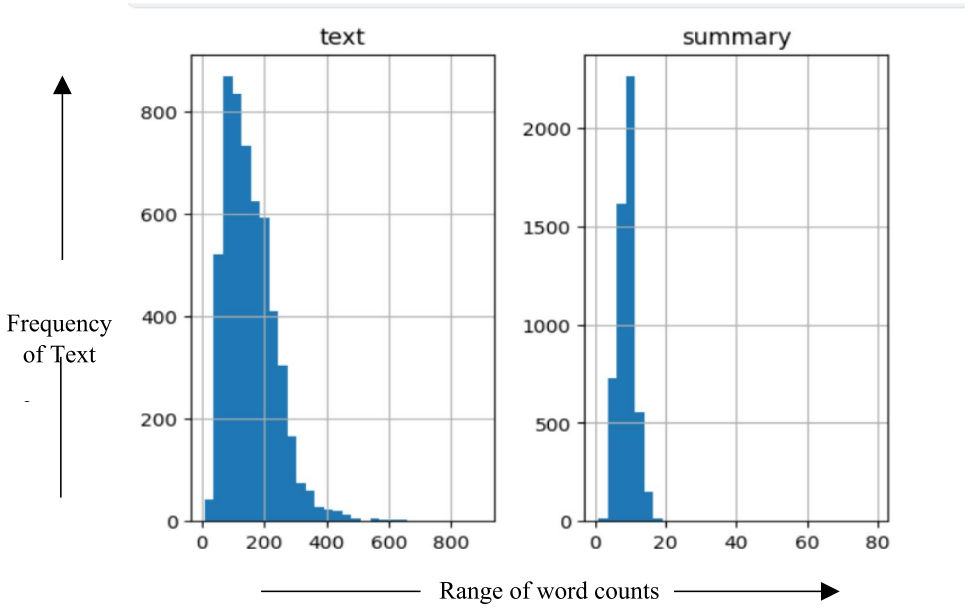

Fig. 4 Distribution of text lengths in documents

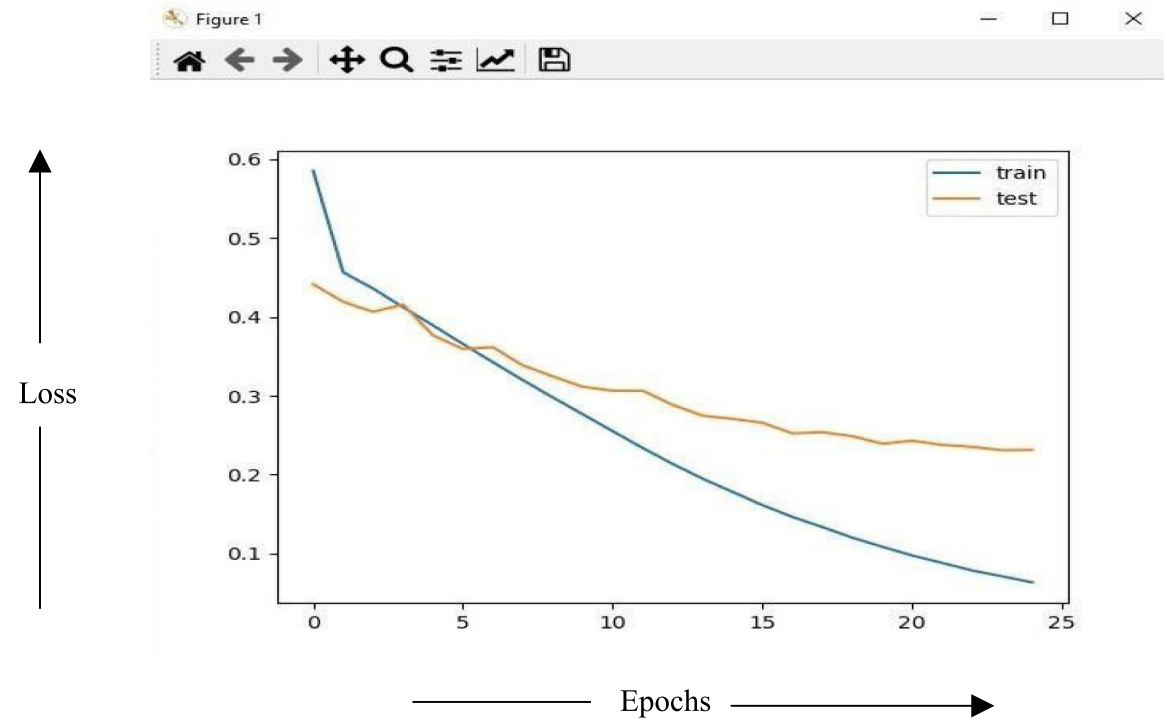

Fig. 5 Loss Curves for Kannada Text Summarization with Seq2Seq Model

Table 3 Hyperparameter values of proposed model

Parameters	Values
Epochs	25
Batch size	32
Latent dim	400
Embedding dim	300

with latent dimension (Latent dim) which finalizes the size of the hidden states in the encoder and decoder layers. This model performed good with 400 latent dim value and with this values model was capable to learn complex patterns, relationships, and structures in the data. Larger latent dimensions enable the model to capture complex relationships in the input data but increase computational cost and memory usage. Embedding dimension (Embedding dim) decides the size of words/tokens, larger the value of this hyperparameter will enable the model to capture richer semantic and syntactic information of data. This model was able to generate the summary with good semantic relation among the sentences with 300 values. Increasing this value to 500 did not contributed in performance improvement. Smaller value (200) did not generated summary with proper semantic relation.

Results

The Table 4. Shows the sample result of Kannada abstractive text summarization using sequence to sequence model. The first part consists the sample input in Kannada language and next part consists the reference summary which is prepared by the language expert, the last part is the model generated summary. Table 5. Shows the sample result translated in English language of Table 4.

Table 6 discusses the Rouge scores of the entire dataset and the scores are based on sequence-to sequence model using LSTM.

ROUGE-1, which is the measurement of common unigrams (individual words) between the generated text and the reference, the recall is 0.6075, meaning that 60.75% of the unigrams in the reference texts are present in the generated text. The precision is 1.0, indicating that all unigrams in the generated text are found in the reference texts. The F1 score, a harmonic mean of recall and precision, is 0.7046, reflecting a balanced performance between recall and precision.

ROUGE-2, which assesses the overlap of bigrams (two consecutive words), shows a recall of 0.4811, meaning 48.11% of the bigrams in the reference texts appear in the generated text. The precision is 0.8, so 80% of the bigrams in the generated text are present in the references. The F1 score for ROUGE-2 is 0.5500, indicating a moderate

Table 4 Sample Result Kannada abstractive text summarization

Input	"ಕಾರವಾರ: ಮಹಾ ನಗರಗಳಿಗೆ ಮಾತ್ರ ಸೀಮಿತವಾಗಿದ್ದ ಕ್ರಿಪ್ಟೋ ಕರೆನ್ಸಿ ವ್ಯವಹಾರ, ಅವ್ಯವಹಾರ ಈಗ ಜಿಲ್ಲೆಯಲ್ಲೂ ವ್ಯಾಪಿಸಿದೆ. ಕ್ರಿಪ್ಟೋ ಕರೆನ್ಸಿ ಪಡೆದು ಹಣ ದ್ವಿಗುಣಗೊಳಿಸಿಕೊಳ್ಳುವ ಆಸೆಯಲ್ಲಿ ಜಿಲ್ಲೆಯ ವ್ಯಕ್ತಿಯೊಬ್ಬ ಮೋಸ ಹೋಗಿದ್ದು, ಈ ಬಗ್ಗೆ ಶಿರಸಿ ನಗರ ಠಾಣೆಯಲ್ಲಿ ದೂರು ದಾಖಲಾಗಿದೆ.ಕೊಪ್ಪಳ ಜಿಲ್ಲೆಯ ಹನುಮೇಶ ಹಿರೇಮನಿ, ಅಶೋಕ ಟಿ.ಕೆ., ಹಾಗೂ ಅನಿಲ ಕುಮಾರ ಎಂ. ಎಂಬುವರು ವಿರುದ್ಧ ಶಿರಸಿ ಅಂಬಾಗಿರಿ ಸಮೀಪದ ಲೋಕೇಶ ಹ್ಯಾಪನವರ್ ದೂರು ನೀಡಿದ್ದಾರೆ......"
Reference Summary	ಮೆಟ್ರೋ ನಗರಗಳಿಗೆ ಸೀಮಿತವಾದ ಕ್ರಿಪ್ಟೋ ಕರೆನ್ಸಿ ವ್ಯವಹಾರವು ಈಗ ಸಣ್ಣ ಪಟ್ಟಣ ಗಳಿಗೆ ವಿಸ್ತರಿಸಿದೆ, ಇದರ ಮೂಲಕ ಲೋಕೇಶ ಹಣವನ್ನು ದ್ವಿಗುಣಗೊಳಿಸುವ ಭರವಸೆಯೊಂದಿಗೆ ಮೋಸ ಹೋಗಿದ್ದಾರೆ. ಇದು ಪೊಲೀಸ್ ಠಾಣೆಯಲ್ಲಿ ದೂರು ಸಲ್ಲಿಸುವಂತೆ ಮಾಡಿತು.
Model Summary	ಮೆಟ್ರೋ ನಗರಗಳಿಗೆ ಸೀಮಿತವಾದ ಕ್ರಿಪ್ಟೋ ಕರೆನ್ಸಿ ವ್ಯವಹಾರವು ಈಗ ಸಣ್ಣ ಪಟ್ಟಣಗಳಿಗೆ ವಿಸ್ತರಿಸಿದೆ, ಲೋಕೇಶ್ ಹಣವನ್ನು ದ್ವಿಗುಣಗೊಳಿಸುವ ಭರವಸೆಯೊಂದಿಗೆ ಮೋಸ ಹೋಗಿದ್ದಾರೆ.

Table 5 Sample result translated in English language

Input	"Karwar: The crypto currency business, which was limited only to the big cities, has now spread to the district as well. A person from the district was cheated in the desire to double his money by getting crypto currency, and a complaint has been filed in Shirasi Nagar police station. Hanumesha Hiremani, Ashoka T.K., and Anila Kumar M. of Koppal district. Lokesha Happanavar of Shirsi Ambagiri has filed a complaint against the said person……
Reference Summary	The crypto currency business, which was limited to metro cities, has now expanded to small towns, through which Lokesh has been cheated with a promise to double money. This made him file a complaint at the police station
Model summary	The crypto currency business, which was limited to metro cities, Lokesh has been cheated with a promise to double money

Table 6 ROUGE scores for Kannada abstractive summarization using sequence to sequence model

	Recall	Precision	F1Score
Rouge-1	0.6075	1.0	0.7046
Rouge-2	0.48111	0.8	0.5499
Rouge-L	0.6075	1.0	0.7046

level of bigram overlap and balance between recall and precision.

ROUGE-L, which measures the longest common subsequence between the generated and reference texts, has the same recall and precision values as ROUGE-1 (0.6075 and 1.0, respectively), resulting in the same F1 score of 0.7046. This suggests that the generated text maintains a strong sequential correspondence with the reference texts.

The F1-Score for each Rouge metric is evaluated as a harmonic mean of precision and recall using Eqs. (9), (12) and (15).

The average ROUGE F1 score across these metrics is 0.6531, providing a single metric summarizing the model's overall performance. This average indicates that the generated text has a good balance of unigram and bigram overlap and sequential similarity with the reference texts, demonstrating the effectiveness of the model in producing text closely aligned with the references.

Table 7 discusses the analysis and comparison of results between the proposed model and other Indian regional languages. The contribution of text summarization is more in south Indian languages such as Telugu, Malayalam and Kannada. Summarization on Telugu language is implemented in which noun and verb count dominates the sentence rank with an accuracy of F1-score of 0.67 [3]. Abstractive summarization on Telugu language is proposed with limited dataset and achieved 37% accuracy [4]. The work demonstrates on Kannada language using extractive summarization with limited dataset size and achieved accuracy of F1-score 0.607 [7, 12, 15]. Proposed model works on Kannada language using sequence to sequence model based on LSTM and achieved better accuracy with average F1-score of 0.653. This model performed better than existing work with dataset size of 10,000 documents and good part is that it achieved the F1-score of 0.7046 with Rouge-L which indicated the model generated summary is closest to the reference summary with long sequence of sentences being matched.

Table 7 Comparison of results between the proposed model and other Indian regional languages

Year	Language	Dataset	Method	Accuracy
2023 [3]	Telugu	3000 samples	Extractive summarization using graph-based approach	F1-score of 0.67
2023 [4]	Telugu	Input data with average 10 sentences	Abstractive summarization using sequence to sequence model	37%
2022 [6]	Kannada	20 Sentences	Extractive text summarization sentence ranking algorithm	–
2022 [7]	Telugu	–	Extractive method using Genetic algorithm, Particle swarm optimization and Heuristic approach	58%
2022 [9]	Hindi	–	Abstractive summarization using Indic Bart	55%
2021 [12]	Kannada	8 articles with 10 lines each	Extractive summarization using POS tagging	F1-score of 0.607
2021 [13]	Malayalam	-	Extractive text summarization using LSA and text rank	F1-score of 0.667
2021 [14]	Hindi and Marathi	Hindi-2lakh Marathi-100	A stacked LSTM based sequence to sequence model	63.8%
2020 [15]	Kannada	40 documents	Extractive summarization using Jaccards similarity score	–
Proposed Model	Kannada	10,000 documents	Abstractive text summarization using sequence to sequence model based on LSTM	F1-score of 0.653

The proposed model achieved a good accuracy for Rouge-1 and Rouge-L as compared to Rouge-2. This implies that model performs good with capturing unigrams and sentence level structures and lags with bigram coherence which may affect in building semantic relationship among larger sentences with complex structure. The model can be further extended by assigning attention weights and combine sequence to sequence model with Bert model to achieve good accuracy for Rouge-2.

Conclusion

The proposed work leverages sequence-to-sequence attention-based model for abstractive text summarization in Kannada language. This model was trained using a dataset containing Kannada news articles and their respective summaries with 10,000 documents. The model achieved a Rouge-1 F1 score of 0.7046, demonstrating its capability to generate relevant summaries. The model successfully learned to create summaries and performed better than the existing work especially in Kannada language as in this language only extractive method is experimented with very small dataset. The ROUGE-2 and ROUGE-L scores were 0.5500 and 0.7046, respectively, and the average ROUGE F1 score was 0.6531. The proposed model utilizes the ReLU activation function in order to filter the relevant information based on previous states. The model effectively handles morphologically rich text by leveraging optimized hyperparameters and pre-trained embeddings. The future direction includes using different architectures such as Transformer and Bi-LSTM and utilizing pre-trained language models for Kannada to improve the model's understanding of the language. Additionally, enhancing the model's ability to summarize multilingual numeric data is crucial for its broader applicability.

Authors Contribution Both the authors play a vital role in the research of proposed work. Dakshayani Ijeri: This author prepared data set and implemented the sequence-to-sequence model for abstractive text summarization on Kannada language. Prepared initial draft of manuscript. Pushpa B. Patil: This author performed the validation and analysis of results. Reviewed and edited the manuscript. Both the authors read and approved the final version of the manuscript.

Funding Authors acknowledges Karnataka State Council for Science and Technology (KSCST) Govt. of Karnataka (Grant No. 47S_BE_1049).

Data Availability The data supporting this study are available from the corresponding author upon reasonable request.

Declarations

Conflict of Interests The authors declare that they have no conflict of interest.

Research Involving Human and /or Animals Not applicable.

Informed Consent Not applicable.

References

1. Sharma S, Singh. Advancements in legal text summarization: integrating InLegalBERT for effective extractive summarization. Intern J Sys Assur Eng Manag. 2025. https://doi.org/10.1007/s13198-025-02783-8.

2. Aote SS, et al. Binary particle swarm optimization with an improved genetic algorithm to solve multi-document text summarization problem of hindi documents. Eng Appl Artif Intell. 2023. https://doi.org/10.1016/j.engappai.2022.105575.
3. Babu GLA, Srinivasu B. Extractive summarization of telugu text using modified text rank and maximum marginal relevance. Transact Asian Low-Res Lang Inform Process. 2023. https://doi.org/10.1145/3600224.
4. Babu GLA, Badugu S. Deep learning-based sequence to sequence model for abstractive telugu text summarization. Multimed Tools Applicat. 2023. https://doi.org/10.1007/s11042-022-14099-x.
5. Pawar S, Manjula Gururaj H, Chiplunar NN. Text summarization using document and sentence clustering. Procedia Comp Sci. 2022;215(1):361–9.
6. Pallavi P, and Sarvamangala DR. Kannada text summarization using extractive technique. 2022 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT). IEEE, 2022.
7. Rao M. Varaprasad, et al. Telugu text summarization using HS and GA particle swarm optimization algorithms. smart intelligent computing and applications, volume 1: Proceedings of Fifth International Conference on Smart Computing and Informatics (SCI2021). Singapore:Springer Nature Singapore,2022.
8. Durga CBV, Babu D. Telugu text summarization using histo fuzzy c-means and median support-based grasshopper optimization algorithm (msgoa). J Theor Appl Inform Technol. 2022;100(17):5418–38.
9. Agarwal,A, Naik S, and Sonawane S. Abstractive Text Summarization for Hindi Language using IndicBART. Working Notes of FIRE 2022-Forum for Information Retrieval Evaluation, Kolkata, India. 2022.
10. Tomer M, Kumar M. Multi-document extractive text summarization based on firefly algorithm. J King Saud Univers. 2022;34(8):6057–65.
11. Bani-Almarjeh M, Kurdy MB. Arabic abstractive text summarization using RNN-based and transformer-based architectures. Inform Proces Manage. 2023. https://doi.org/10.1016/j.ipm.2022.103227.
12. Jayashree R, Basavaraj S, Anami B and Poornima K. Text document summarization using POS tagging for Kannada text documents. 2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence). IEEE, 2021.
13. Jayan JP and Govindaru V. Automatic summarization of malayalam documents using text extraction methods.
14. Karmakar R, et al. Indian regional language abstractive text summarization using attention-based LSTM neural network. 2021 International Conference on Intelligent Technologies (CONIT). IEEE, 2021.
15. Jayashree R and Vinay SK. A Jaccards Similarity score based methodology for kannada text document summarization. 2020 International Conference on Advances in Computing, Communication & Materials (ICACCM). IEEE, 2020.
16. Zaki, AM, Khalil MI and Abbas HM. Deep architectures for abstractive text summarization in multiple languages. 2019 14th International Conference on Computer Engineering and Systems (ICCES). IEEE, 2019.
17. Dave H and Jaswal S. Multiple text document summarization system using hybrid summarization technique. 2015 1st International Conference on Next Generation Computing Technologies (NGCT). IEEE, 2015.
18. Khan A, Salim N, and Kumar YJ. Genetic semantic graph approach for multi-document abstractive summarization. 2015 Fifth International Conference on Digital Information Processing and Communications (ICDIPC). IEEE, 2015.