International Journal of Intelligent Networks 4 (2023) 53–61

Contents lists available at ScienceDirect

International Journal of Intelligent Networks

journal homepage: www.keaipublishing.com/en/journals/international-journal-of-intelligent-networks

HNTSumm: Hybrid text summarization of transliterated news articles

Padhma Muniraj [a,*], K.R. Sabarmathi [b], R. Leelavathi [b], Saravana Balaji B [c]

[a] System Programmer, Kyndryl Inc., India
[b] Department of Information Technology, Bannari Amman Institute of Technology, Sathyamangalam, India
[c] Department of Information Technology, Lebanese French University, Erbil, Iraq

ARTICLE INFO

Keywords:
Automatic text summarization
Natural language processing
Word embeddings
Transliteration
Deep learning

ABSTRACT

Data generated from social networking sites, blogs, digital magazines, and news websites is the largest human-generated data. Summarization is the process of extracting the crux of a document which when done manually can be tedious and deluging. Automatic text summarization is an approach that encapsulates long documents into a few sentences or words by enwrapping the gist and the principal information of the document. With the growth of social networking sites, eBooks, and e-Papers, the prevalence of transliterated words in text corpora is also on the rise. In this paper, we propose a word embeddings-based algorithm called HNTSumm by combining the advantages of unsupervised and supervised learning methods. The proposed algorithm HNTSumm algorithm is an imminent method for automatic text summarization of huge volumes of data that can learn word embeddings for words transliterated from other languages to English by utilizing weighted word embeddings from a Neural Embedding Model. Further, the amalgamation of extractive and abstractive approaches yields a concise and unambiguous summary of the text documents as the extractive approach eliminates redundant information. We employ a hybrid version of the Sequence-to-sequence models to generate an abstractive summary for the transliterated words. The feasibility of this algorithm was evaluated using two different news summary datasets and the accuracy scores were computed with the ROUGE evaluation metric. Experimental results corroborate the higher performance of the proposed algorithm and show HNTSumm outperforms relevant state-of-the-art algorithms for datasets with transliterated words.

1. Introduction

The mid-twentieth century witnessed a radical shift from the conventional industries engendered by the industrial revolution to an industry that was primarily based on information technology. The advent of this so-called "information age", also known as the digital age, fuelled by the internet, digitized information in a way that most industries capitalized information. The amount of digital information that was generated witnessed exponential growth over time. According to current estimates, as of today, 2.5 quintillion bytes of data are generated every day. This explosion in the amounts of data that are being generated has led to information overload. We are confronted with a world of data where we have more than we could handle. This rise in the amounts of data and the technologies developed to solve problems using data has opened the human mind to new possibilities that were previously inconceivable. Among the large volumes of generated data, unstructured text data tends to be the largest human-generated data. We come across many different kinds of unstructured text data in our everyday life, which includes emails, social media posts, blog posts, live chats, instant messaging, eBooks, digital magazines, newspapers, medical records, and much more. Natural Language Processing (NLP), one of the subfields within Artificial Intelligence, helps process unstructured text data. Recent advances in NLP enable us to solve many fundamental problems in computational linguistics such as knowledge extraction and summarization, machine translation, machine transliteration, sentiment analysis, question answering, and much more. In this paper, our work focuses primarily on two fundamental problems in the field of computational linguistics - summarization and transliteration.

Automatic text summarization (ATS) is a subfield of NLP that aims to condense lengthy documents into a brief and precise summary by retaining the relevant context of the document [1]. ATS has been used in various applications over the past 20 years, spanning a broad range of industries and verticals. Some of the applications include but are not limited to, generating automatic review summaries for scientific papers

* Corresponding author.
E-mail addresses: padhmasahithya@gmail.com (P. Muniraj), sabarraju@gmail.com (K.R. Sabarmathi), leelavathi.rajan@gmail.com (R. Leelavathi), saravanabalaji.b@lfu.edu.krd (S. Balaji B).

https://doi.org/10.1016/j.ijin.2023.03.001
Received 11 August 2022; Received in revised form 29 December 2022; Accepted 3 March 2023
Available online 5 March 2023

[2], email summarization, summarizing long documents such as books [3], analyzing and summarizing customer reviews to identify customer concerns [4], creating summaries of news articles, generating abstract for scientific literature based on the manuscript [5], tweet summarization, automated content creation and much more. The process of ATS can be classified into two methods: extractive and abstractive summarization [6]. Extractive summarization methods work by identifying important elements of text in a document and literally extracting a few sections of the document. This method suffers from the limitations of suboptimal coherence between sentences and inefficacy in simplifying long and involved sentences. Abstractive summarization methods use advanced natural language techniques to generate new shorter content that conveys the gist of the original text in the document. Abstractive summaries are more related to human-generated summaries but generating them is a strenuous task due to the limited availability of quality datasets. The advent of sequence-to-sequence models [7] and the success of transformer-based models [8] have made abstractive summarization a thriving area of research. Still, the limitation of these models is that they require vast amounts of data. Hence, the research in this area has been primarily confined to high-resource languages like English. Recently, many researchers and scholars have proposed methods for summarizing texts written in different languages using many state-of-the-art models. However, it is still a challenge to generate summaries that are grammatically meaningful.

The authors of [9] compiled a professionally annotated dataset for 44 languages and proposed a multilingual text summarization approach using the pretrained mT5 transformer that shows higher performance for even low-resource languages. An extractive summarization technique to summarize Arabic text using the pre-trained AraBERT model in combination with sentence embeddings was put forward by the authors of [10]. In Ref. [11], the authors utilize multilingual learning by consolidating language model training, auto-encoder training, translation, and reverse translation training for five high-resource and two low-resource languages. This method creates a unified model to summarize news articles through abstractive summarization. The above references show significantly higher performance for high-resource and low-resource languages. Nevertheless, these approaches lack the means to recognize and account for words that are not English phonetically, hence they cannot be used on corpora that contain a mixture of English and other languages.

Transliteration, a subfield of computational linguistics, is the process of mapping from one system of writing to another based on phonetic similarities. Unlike translation, transliteration is not concerned with the meaning of the word being transliterated rather it is dependent on the pronunciation of the word and aims to retain the phonological structure of the target language. For example, in translating between Hindi to English, the Hindi word धन्यवाद is translated as “Thank you” whereas, in transliteration, it is represented as “dhanyavaad”. With the exponential surge of social networks, the use of transliterated words has become more prevalent than ever before. They are more pervasive among social networks such as Twitter and Facebook, video transcripts, news articles containing interviews from non-English speakers, textbooks for bilingual children, etc., Current research in transliteration includes the use of deep learning architectures [12] such as RNN and CNN to map characters from source language to target language. In Ref. [13], the authors present a new Arabic-to-English transliteration dataset and an attentional sequence-to-sequence model (Seq2seq) with RNNs to transliterate from Arabic to English. In social networks, an LSTM model [14] has been used to transliterate Twitter data to make users more cognizant of the content and to improve social security. In Ref. [15], the authors propose a transliteration model that aims to translate from Tamil to English and then generate a summary. However, most of these works are centered around transliterating from one language to another language or identifying the presence of transliterated words in Latin scripts, or applying ATS after transliterating from source to target language. Applying ATS to datasets comprising English and transliterated words from other languages is a relatively new and understudied area of research.

Considering the above-mentioned works, the objective of our proposed work is to automatically generate concise and succinct summaries for datasets with transliterated words. We utilize online news articles to validate and assess the methodology proposed in this paper as they are a rich source of transliterated words.

The following is a brief summary of our contribution.

- We propose an integrated approach to ATS that incorporates both extractive and abstractive summarization methods in combination with a neural embedding model for word embeddings called Hybrid Neural Text Summarization (HNTSumm).
- The Neural Embedding Model learns custom word embeddings tailored for our dataset with transliterated words.
- In extractive summarization, we use the TextRank algorithm, an unsupervised learning technique. For the abstractive summarization, a hybrid version of the Seq2seq model is used.
- The Hybrid Seq2seq model uses a Bidirectional LSTM (Long Short-Term Memory) for encoders, an LSTM for the decoder, and the Beam Search strategy for decoding the test sequences. Furthermore, the attention mechanism is utilized to assign weights so that the decoder can translate it more accurately and make better predictions.
- The results of the experiments demonstrate the viability of the suggested method, greater ROUGE scores in comparison to previous summary methodologies, and that the results are semantically meaningful comparatively.

The rest of the paper is structured as follows: Section 2 discusses the relevant research work and methodologies in text summarization. In section 3, we outline the proposed methodology and explain each of its phases extensively. In section 4, we discuss our experimental results and discussions. Finally, section 5 concludes our proposed research work.

2. Related works

The following sections briefly explain the literature survey for text summarization. The discussion is mainly around word embeddings, supervised algorithms, unsupervised algorithms, and evaluation metrics.

2.1. Word embeddings

An essential step in most, if not all, NLP algorithms is the representation of words and documents. While techniques such as Bag of Words (BOW), CountVectorizer, and Term Frequency(TF) - Inverse Document Frequency (IDF) encode each word in the vocabulary into numeric vectors, they depend on the number of words in the corpus and the size of the vector is the number of words in the vocabulary, sometimes resulting in a sparse matrix where most of the elements are zero and also fail to capture the underlying syntactic or semantic information of the corpus which makes them inefficient.

The authors in Ref. [16] utilize word embeddings for document summarization with the Latent Semantic Analysis method. They propose a local weighting scheme with word embeddings for terms in a sentence and a global weighting scheme with word embeddings for terms in the document. This approach combines word embeddings with traditional weighting schemes.

In [17], the authors propose a weighted word embedding method for word vector representation to capture semantic meaning from text including linguistic and statistical features. To perceive the underlying semantic meanings in text, this work implements distributional representation vectors and generates diverse document summary by excluding semantically distinct sentences.

2.2. TextRank algorithm

TextRank is an unsupervised graph-based learning algorithm mostly used for extractive summarization in NLP. It is based on Google's PageRank algorithm used to rank the web pages in the search engine's results based on hyperlinks [18]. TextRank algorithm extracts key phrases from a document by building a graph with the sentences in the documents as nodes. The edges in the graph are undirected and weighted which indicates some degree of similarity between the two nodes [19]. The weights between two nodes are computed using similarity measures such as cosine similarity, or Jaccard Similarity.

2.3. Sequence-to-sequence model

Seq2seq models were first introduced by Google in 2014 [7] that aim to map a fixed-length input sequence to a fixed-length output sequence where the input and output lengths can be different. These models are used in Google Translate, Speech Recognition, Text Summarization, Image, and Video Captioning, Online Chatbots, etc. Seq2seq models have an encoder-decoder architecture to implement sequence conversion where Recurrent Neural Networks (RNNs) or variants of RNNs such as Gated Recurrent Neural Networks (GRU) or Long Short Term Memory (LSTMs) are used as encoder and decoder components [20]. There are 3 elements in the Seq2seq model: an encoder, a context vector, and a decoder. The encoder contains a stack of several RNNs (sometimes LSTMs or GRUs) where each one of them accepts a single element of the input sequence and summarizes all the information in the context vector. After processing all the input sequences, the encoder passes the context vector to the decoder [21]. The context vector is the initial hidden state of the decoder and is calculated using the formula (a),

$$h_t = f\left(W^{(hh)} h_{t-1} + W^{(hx)} x_t\right) \tag{a}$$

The decoder accepts the context vector and tries to predict the desired output sequence using the formula (b),

$$y_t = \text{soft max}(W^s h_t) \tag{b}$$

In Seq2Seq, the attention mechanism enhances the weight of critical features and decreases the weight of low-impact features by computing the weight of features. This mechanism facilitates the model to pay greater attention to effective features and glean most useful information. The internal formula is given by (c),

$$l_t = \sum_{t=1}^{n} (\text{softmax}(\sigma(W_a h_t + b_a)))x_t \tag{c}$$

2.4. ROUGE score

ROUGE - Recall-Oriented Understudy for Gisting Evaluation is a set of evaluation metrics in NLP to evaluate automatic text summarization and machine translation. It includes the measures to automatically evaluate the quality of a summary generated through automatic text summarization by comparing it against other 'gold-standard' summaries created by humans [22]. ROUGE score is yielded by combining precision, recall, and f-Score. ROUGE-1 evaluates the overlapping unigrams between the system-generated and the reference summary. ROUGE-2 evaluates the overlapping bigrams between the system-generated and the reference summary. ROUGE-l measures Longest Common Subsequence (LCS) that compares the sentence-level similarity between the system summary and the human summary [23]. ROUGE-1 achieves high accuracy among the various measures of ROUGE-n.

$$\text{recall} = \frac{\text{Number ofoverlapping words}}{\text{Total words in reference summary}} \tag{d}$$

$$\text{precision} = \frac{\text{Number of overlapping words}}{\text{Total words in system summary}} \tag{e}$$

$$\text{F} - \text{Measure} = 2 * \frac{\text{precision x recall}}{\text{precision} + \text{recall}} \tag{f}$$

In (d), Recall measures how much of the reference summary is captured in the system summary. In (e), Precision measures the relevancy of the system summary with respect to the reference summary. In (f), F-measure is a combination of both Recall (d) and Precision (e), and it offers a balanced score of the two measures.

2.5. Hybrid models

In [24], the authors propose a hybrid model architecture including both extractive and abstractive text summarization approaches. Post preprocessing, they use the extractive summarization approach where they obtain relevant sentences from input documents using 2-layers of RNN. The output from the previous phase is fed to the abstractive summarization phase which generates the text summary output. For the abstractive method, they have used a Pointer-generator network model.

In [25], Li et al. introduce a guiding generation model for abstractive text summarization. Initially, they retrieve keywords from the source texts using an extractive approach. Then, a Key Information Guide Network (KIGN) is introduced which encodes the keywords obtained from the previous step and generates the final summary by integrating it into the abstractive model. For the guidance aspect, they use the attention mechanism and pointer mechanism in the KIGN model.

The authors of [26], propose a multi-document summarization model that leverages extractive summarization to remove redundant information and abstractive summarization to generate a summary. They utilize BART and T5 pre-trained models to generate two summaries and favor the model with higher diversity in the output sentences.

3. Proposed system

The following section outlines the algorithms used and the steps in generating a summary from the news articles. Initially, datasets are collected as raw unstructured text data as they are available in the real world. Then, the dataset is subjected to data cleaning to make it suitable for the algorithms. Finally, the cleaned and structured data is passed through the three phases of the HNTSumm learning algorithm to generate a summary.

3.1. Data collection

The process of data acquisition can be of various methods. In the past few years, several datasets have been compiled and produced for the study and research of NLP. In order to formulate and verify our proposed algorithm, we have used the NEWS SUMMARY dataset [27] from Kaggle as this dataset has been compiled specifically for text summarization purposes. This dataset comprises news stories from The Hindu, Times of India, the Guardian, and various other sources along with human-generated abstractive summaries. The corpus contains a total of 102,915 documents along with a human-generated summary.

The second dataset named RegNEWS was manually compiled by our team by scraping 50,246 news articles from the web. This dataset contains stories from Deccan Herald, Deccan Chronicle, The Asian Age, and Central Chronicle. We have collated this dataset specifically for transliterated words. The corpus for this dataset contains a total of 50,246 documents and the headline for each article is considered as the summary. We have estimated that the proportion of transliterated words in the NEWS SUMMARY dataset is around 24% (192K words) and nearly 43% (236K words) in the RegNEWS dataset.

3.2. Preprocessing

Unstructured text data often contains noise in the form of inconsistencies or meaningless elements that can't be interpreted by many

machine learning algorithms. The first phase in our methodology involves cleaning the input data with NLP preprocessing libraries such as NLTK and SpaCy and making it suitable for the HNTSumm algorithm by following the below steps.

- Removing duplicates or Nan values: Any duplicate documents are removed from the corpus to avoid redundancy followed by removing Nan values.
- Removing HTML Tags: Text data often contains HTML tags if they are obtained through web scraping or crawling. These are removed as they don't add much value to understanding the text.
- Expanding Contractions: Contractions are shortened words that are formed by combining two words by dropping some letters and replacing them with an apostrophe. Expanding contractions aids in Text Standardization.
- Tokenization: Tokenization is splitting each document in the corpus into sentences and then into words.
- Removing Stopwords: Stopwords are the set of words that have little meaning but are used very often. They are removed with the help of a pre-defined word list in the NLTK package.
- Removing characters other than alphabets: Special characters such as symbols, punctuations, accented characters, and numbers are removed.
- Converting Case: All words in the corpus are converted to lowercase.

The preprocessing steps are outlined in the flow chart in Fig. 1. The final cleaned text data obtained from the last step is fed to phase 1 of the algorithm.

3.3. Phase 1 - Neural embedding model

Word embeddings provide us with an efficient, dense representation where individual words are represented in a predefined vector space as real-valued vectors. This enables them to capture meaning in the text as similar words have similar representations. There are two methods to obtain word embeddings.

1 Learning embeddings for the corpus by setting up a neural network with random initial weights and then the embedding layer learns weights for all the words from the training dataset.
2 Loading pre-trained word embeddings into the model and freezing the embedding layer of the neural network by disabling the trainable attribute [28].

For the HNTSumm algorithm, we have used the first method and trained an embedding layer from scratch using Keras Embedding Layer. After the execution of the preprocessing steps outlined above, the corpus contains more than 8 Million unique words which are fed to the embedding layer as an input of a 2D tensor of integers, with each input as a sequence of integers. These sequences are padded to have the same length as they are packed into one tensor. The embedding layer is instantiated with 3 parameters.

- input_dim: a vocabulary size of
- 800,000 + 1 (for NEWS SUMMARY dataset) and
- 550,000 + 1 (for RegNEWS dataset)
- output_dim: input_dim ** 0.25
- input_length: length of the longest sentence in the 2D tensor

It is a general rule of thumb to take the fourth root of input_dim for the output_dim. The output_dim produces dense 3D tensor embedding. Once the model is fully trained, we get an embedding matrix with a dense structure that is specialized for our dataset.

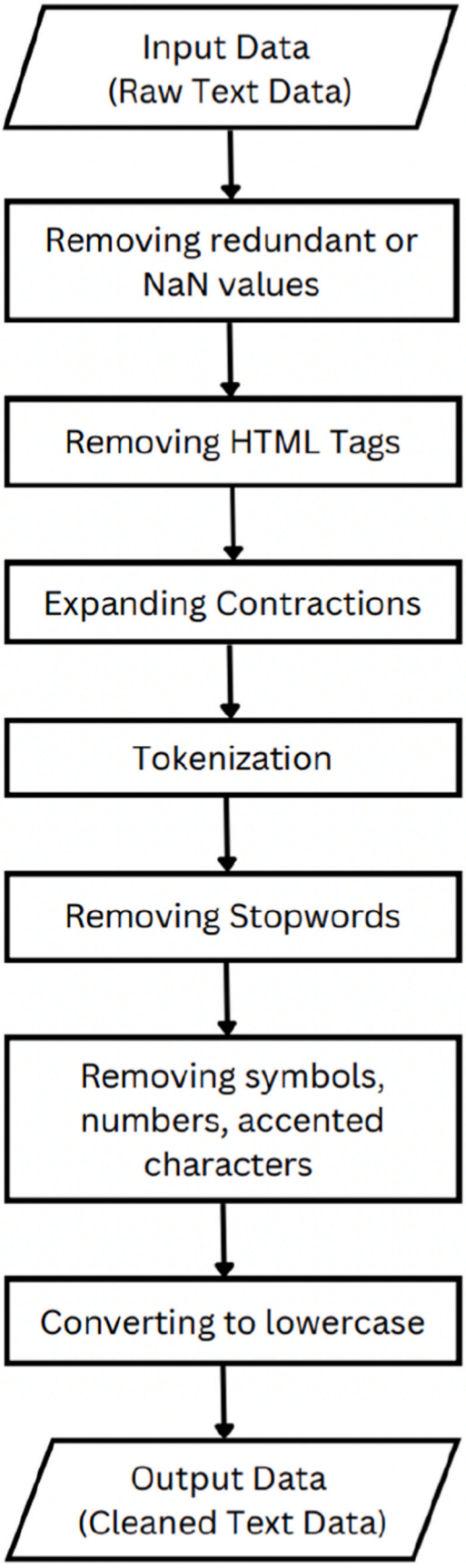

Fig. 1. Text wrangling and preprocessing.

3.4. Phase 2 - Unsupervised phase

The embedding matrix obtained from the neural embedding model is fed to the next phase of the algorithm for extractive summarization. From the corpus, each document is split into sentences and a zero matrix of dimensions (n * n) is initialized, where n is the number of sentences in each document. Then, cosine similarity is computed between pairs of sentences in each document with the embedding matrix, and the zero matrices are updated with the similarity scores of sentences. A weighted graph is constructed with the similarity matrix where the vertices represent sentences, and the edges represent the similarity scores between them [18]. Using the NetworkX python package, the PageRank algorithm is applied over the graph to arrive at the sentence rankings. The output from this phase is the most important sentence in each document of the corpus based on the TextRank algorithm.

3.5. Phase 3 - Supervised phase

The summary extracted from the previous phase using the extractive summarization algorithm is passed as the input for the supervised phase where the abstractive summarization algorithm is applied using the Seq2seq model. Since the Seq2seq models are a class of supervised models, the input data is split into a training set, a validation set, and a test set. For the HNTSumm algorithm, we have used a hybrid Seq2seq model where the encoder consists of three Bidirectional LSTM units and the decoder consists of one LSTM unit. The Bidirectional LSTM in the encoder captures the context of the input sequences before encoding the article's content to its vectors of hidden states which acts as the initial hidden state for the decoder that decodes and generates the summary. When the model reaches the discriminator, it begins to decode the abstract words using decoder LSTM with the beam search approach, where each word is decoded using the encoded states and the words that have already been decoded. Each word that the decoder LSTM decodes then takes part in an attention mechanism to create a context vector that produces the output. This hybrid model is trained for 100 epochs with early stopping on the training set (80% of the dataset). The complete working of the HNTSumm algorithm is outlined in Fig. 2.

4. Results and discussion

All the phases in the HNTSumm algorithm were implemented using Python3 Jupyter Notebook in Google Colab using GPU. The original document from the dataset is preprocessed and fed to the neural embedding model which outputs an embedding matrix that serves as an input for the unsupervised phase. The output from the unsupervised phase acts as an intermediate output and as an input for the supervised phase which generates the final summary as shown in Fig. 3. The generated output is concise and more similar to the human-generated summary.

In order to obtain an algorithm that fits the training data well and generalizes well to new data, the performance of the hybrid model was evaluated with the validation set (10% of the dataset). Dropout is a regularization technique that avoids overfitting and improves model performance by randomly excluding some neurons from the network probabilistically. Dropout values typically fall between 0.1 and 0.5. The higher the value, the higher the number of neurons that are discarded. When the value is 0.5, 50% of the neurons are removed. When the dropout is applied to the recurrent input signal on the LSTM units, it is a

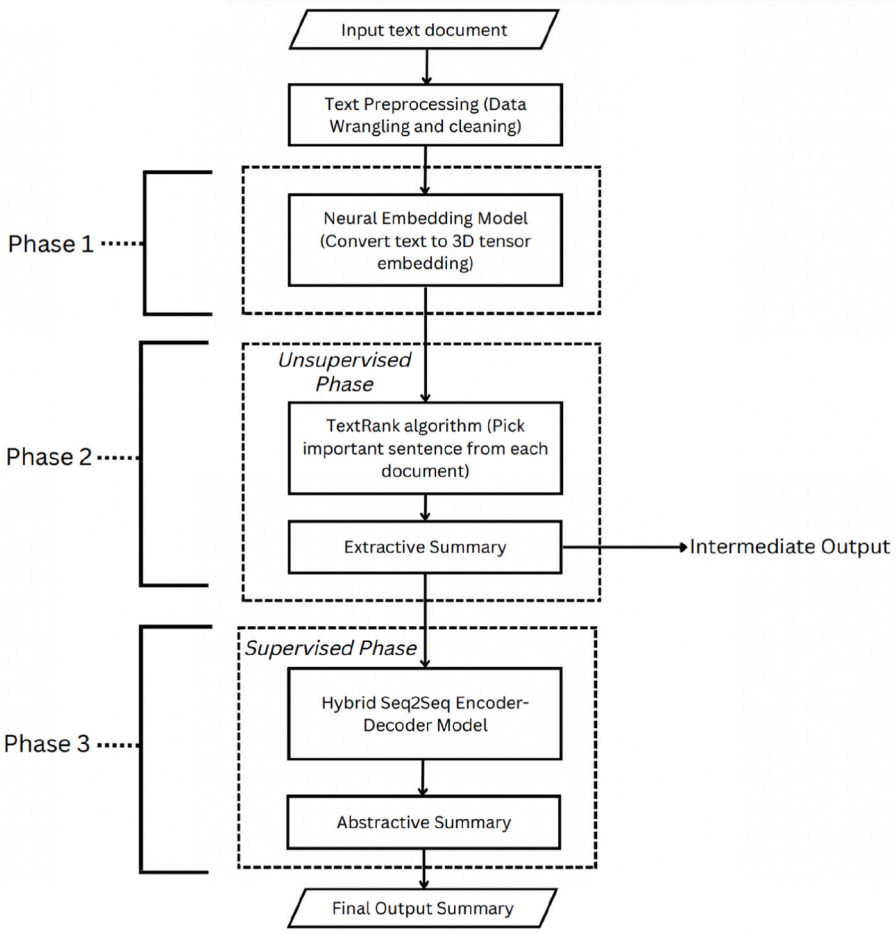

Fig. 2. Phases of proposed HNTSumm algorithm.

Original Input Document: Bengaluru Metro has announced that its first creche will be available at the staff quarters of Baiyappanahalli Metro depot, in line with Maternity Benefit (Amendment Bill) 2016. Officials said the service will be extended to other stations depending on the demand. Metro employees' union said that they had demanded creche facilities at 4 stations.
Unsupervised Summary (Phase 2): *Bengaluru Metro has announced that its first creche will be available at the staff quarters of Baiyappanahalli Metro depot, in line with Maternity Benefit (Amendment Bill) 2016.*
Supervised Summary (Phase 3): *bengaluru metro to get its first creche station*

Sample 2
Original Input Document: On the day of 'Kaanom Pongal', the last day of Pongal festival, jallikattu was conducted against the ban by Supreme Court at Kanjirangudi, Ramanathapuram. People of Kanjirangudi conducted jallikattu against the ban declaring that it is their traditional sport and it can't be banned. The bull specifically trained for 'Yerudhukattu', a form of sport where the bull would be tied to a rope allowing it to move inside a 15ft circumference was conducted under the orders of 'Yeruthu Kattu Sangam' leader, Adithan.
Unsupervised Summary (Phase 2): *The bull specifically trained for 'Yerudhukattu', a form of sport where the bull would be tied to a rope allowing it to move inside a 15ft circumference was conducted under the orders of 'Yeruthu Kattu Sangam' leader, Adithan.*
Supervised Summary (Phase 3): *yerudhukattu bull sport performed orders of yeruthu kattu sangam leader*

Sample 3
Original Input Document: All these communities have understood that in the name of pichda, they (the SP and the BSP) were promoting their own people. For 30 years these people would say 'hathi lao, cycle badlo, cycle lao'. But when Dr. Sanjay started his cadre, they came to their senses and realised that they will no longer get kicked around.
Unsupervised Summary (Phase 2): *For 30 years these people would say 'hathi lao, cycle badlo, cycle lao'.*
Supervised Summary (Phase 3): *people said hathi lao cycle badlo cycle lao*

Fig. 3. Sample input document and output summary from HNTSumm algorithm.

recurrent dropout.

Table 1(a) shows the performance of the model with various dropout and recurrent dropout rates for the NEWS SUMMARY dataset. We observe that the model does require some amount of dropout as the dropout rate of 0 shows the lowest scores. As the dropout rate increases, there is a general increase in performance. The optimal parameters for the NEWS SUMMARY dataset is a dropout rate of 0.4 and a recurrent dropout rate of 0.2.

Table 1(b) shows the performance of the model for the RegNEWS dataset with various dropout and recurrent dropout rates. This dataset also shows similar performance as the NEWS SUMMARY dataset, making the configuration of higher dropout rates preferable. The optimal parameters for the RegNEWS dataset is a dropout rate of 0.2 and a recurrent dropout rate of 0.2.

The Neural Embedding Model (NEM) is the core component of the HNTSumm algorithm. We conducted experiments by replacing the Neural Embedding Model (NEM) with GloVe pre-trained word embeddings and Term Frequency-Inverse Document Frequency (TF-IDF) and fed it to the Phase 2 (Unsupervised Phase) for the TextRank (TR) algorithm. It was observed that an embedding layer trained from scratch using Keras (NEM) outperforms GloVe embeddings and TF-IDF. Our hypothesis is that the Keras embedding layer performs better because it learns embeddings targeted specifically for the task and data at hand. Table 2(a) summarizes various ROUGE scores for the unsupervised phase of the NEWS SUMMARY dataset and Table 2(b) shows ROUGE scores for the RegNEWS dataset. The package used for calculating the ROUGE score is rouge-1.0.1 and it comprises Precision (P), Recall (R), and F1-Score (F).

Fig. 4 displays the performance comparison of various embeddings methods for the NEWS SUMMARY as well as the RegNEWS datasets parallelly. We can observe that the NEM outperforms GloVe embeddings and TF-IDF and the RegNEWS dataset performs better than the NEWS SUMMARY dataset as our model is trained specifically to function better when transliterated words are present. Hence, the output from the NEM model is utilized for the final phase of the algorithm.

In order to verify the advantages of HNTSumm, we evaluated the performance of various combinations of the Seq2Seq model proposed in various studies with the test set (the remaining 10% of the dataset). As part of the comparisons, we evaluated our first model consisting of the Seq2seq model with three units of LSTMs for the encoder and one unit of LSTM for the decoder. This was followed by combining the attention mechanism with the Seq2seq model and then by employing two different strategies for the decoding of the test sequences-greedy search and the beam search. The same procedure was followed for the Hybrid model that consists of three units of Bidirectional LSTMs for the encoder and one unit of LSTM for the decoder. The Hybrid model was also tested by combining and replacing various mechanisms such as attention,

Table 1
F1-score of models with different dropout rates. (a) Dropout rates for NEWS SUMMARY dataset (b) Dropout rates for RegNEWS dataset.

(a)

Dropout	Recurrent Dropout		
	0	0.2	0.4
0	0.02121	0.02453	0.04981
0.2	0.17651	0.30563	0.22121
0.4	0.19267	**0.32341**	0.29101

(b)

Dropout	Recurrent Dropout		
	0	0.2	0.4
0	0.13178	0.18192	0.14192
0.2	0.21761	**0.42315**	0.30214
0.4	0.29019	0.39213	0.31320

Table 2
ROUGE-1, ROUGE-2, ROUGE-L scores for other word embedding methods against Neural Embedding Model (NEM) (a) NEWS SUMMARY dataset (b) RegNEWS dataset.

(a)

Models	Rouge-1			Rouge-2			Rouge-L		
	F	P	R	F	P	R	F	P	R
Glove + TR	0.112	0.083	0.175	0.016	0.011	0.037	0.097	0.071	0.158
TF-IDF + TR	0.140	0.152	0.131	0.076	0.054	0.132	0.198	0.146	0.311
NEM + TR	0.242	0.201	0.305	0.187	0.172	0.206	0.255	0.181	0.432

(b)

Models	Rouge-1			Rouge-2			Rouge-L		
	F	P	R	F	P	R	F	P	R
Glove + TR	0.121	0.091	0.181	0.112	0.112	0.114	0.153	0.168	0.141
TF-IDF + TR	0.146	0.155	0.138	0.136	0.128	0.146	0.219	0.153	0.389
NEM + TR	0.263	0.211	0.351	0.270	0.261	0.281	0.368	0.287	0.513

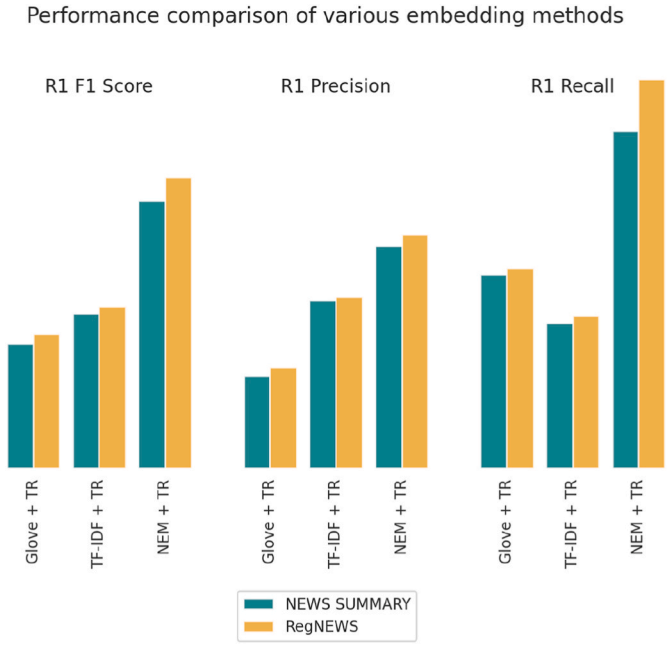

Fig. 4. Performance comparison of various embedding methods in the unsupervised phase.

greedy search, and beam search. Our final simulation was with the HNTSumm algorithm that combines the Hybrid model, attention mechanism, and beam search. All these algorithms were trained with the output from Phase 1(NEM) and Phase 2(TR) of the HNTSumm algorithm. Table 3 presents the ROUGE scores for various variants of the Seq2seq models for the NEWS SUMMARY dataset and Table 4 shows the ROUGE scores for the RegNEWS dataset. As can be observed from the performance of the Seq2seq model, which performs worse as the presence of transliterated words increases, the majority of existing algorithms do not attempt to handle the existence of transliterated words. Also, by comparing the results from Table 3 with Table 4, we can observe that the performance increases significantly for the HNTSumm algorithm.

Fig. 5 compares the Rouge 1 F1-score of various Seq2seq models for both datasets. We can spot a general increase in the output results as we either combine the attention mechanism or utilize a search strategy for the decoding. The proposed approach performs better with the increase in the presence of transliterated words as seen from the scores of the HNTSumm algorithm as well as validates the viability of the algorithm.

5. Conclusion

In this paper, the HNTSumm algorithm comprising a NEM model and hybrid Seq2seq model with an attention mechanism and combining the concepts of extractive and abstractive summarization is established for the summarization of transliterated words. The NEM model creates custom embeddings, the hybrid Seq2seq model generates summaries, and the attention mechanism improves the accuracy of the generated summary. Our experimental results showed that this approach results in an increased ROUGE-1 score of 73.46% compared to the traditional extractive methods and a 57.14% increase to that of the traditional abstractive methods.

In the future, this work can be extended to summarize transliterated datasets that contain words from various low-resource languages. A limitation of the Seq2seq algorithms is that they require large datasets for training. By extending this work to transfer learning methods that don't require much data, we can curb the disadvantages while also enhancing the accuracy of the results.

Declaration of competing interest

The authors declare that they have no known competing financial

Table 3
ROUGE-1, ROUGE-2, ROUGE-L scores of the Seq2Seq models for NEWS SUMMARY dataset.

Models	Rouge-1			Rouge-2			Rouge-L		
	F	P	R	F	P	R	F	P	R
Seq2Seq	0.154	0.211	0.122	0.049	0.061	0.042	0.076	0.056	0.123
Seq2Seq + Attn.	0.143	0.182	0.119	0.099	0.083	0.124	0.132	0.091	0.245
Seq2Seq + Greedy Search	0.169	0.142	0.211	0.090	0.060	0.183	0.334	0.254	0.491
Seq2Seq + Beam Search	0.232	0.167	0.382	0.138	0.092	0.284	0.240	0.192	0.322
Hybrid Seq2Seq	0.276	0.291	0.264	0.163	0.131	0.218	0.130	0.103	0.179
Hybrid + Attn.	0.148	0.211	0.115	0.203	0.192	0.217	0.224	0.221	0.229
Hybrid + Greedy Search	0.147	0.215	0.112	0.153	0.181	0.133	0.205	0.153	0.311
Hybrid + Beam Search	0.229	0.232	0.227	0.189	0.166	0.221	0.340	0.323	0.361
HNTSumm	0.338	0.358	0.321	0.210	0.197	0.226	0.379	0.292	0.543

Table 4
ROUGE-1, ROUGE-2, ROUGE-L scores of the Seq2Seq models for RegNEWS dataset.

Models	Rouge-1			Rouge-2			Rouge-L		
	F	P	R	F	P	R	F	P	R
Seq2Seq	0.132	0.121	0.147	0.116	0.109	0.124	0.033	0.017	0.913
Seq2Seq + Attn.	0.183	0.152	0.233	0.164	0.143	0.194	0.254	0.211	0.321
Seq2Seq + Greedy Search	0.253	0.211	0.316	0.141	0.115	0.183	0.208	0.183	0.241
Seq2Seq + Beam Search	0.205	0.198	0.214	0.155	0.147	0.165	0.204	0.196	0.213
Hybrid Seq2Seq	0.156	0.274	0.110	0.144	0.163	0.129	0.267	0.271	0.264
Hybrid + Attn.	0.184	0.299	0.133	0.220	0.191	0.261	0.273	0.248	0.304
Hybrid + Greedy Search	0.280	0.251	0.317	0.283	0.295	0.272	0.300	0.283	0.321
Hybrid + Beam Search	0.296	0.263	0.341	0.257	0.249	0.267	0.372	0.318	0.450
HNTSumm	0.410	0.363	0.471	0.309	0.297	0.324	0.437	0.362	0.553

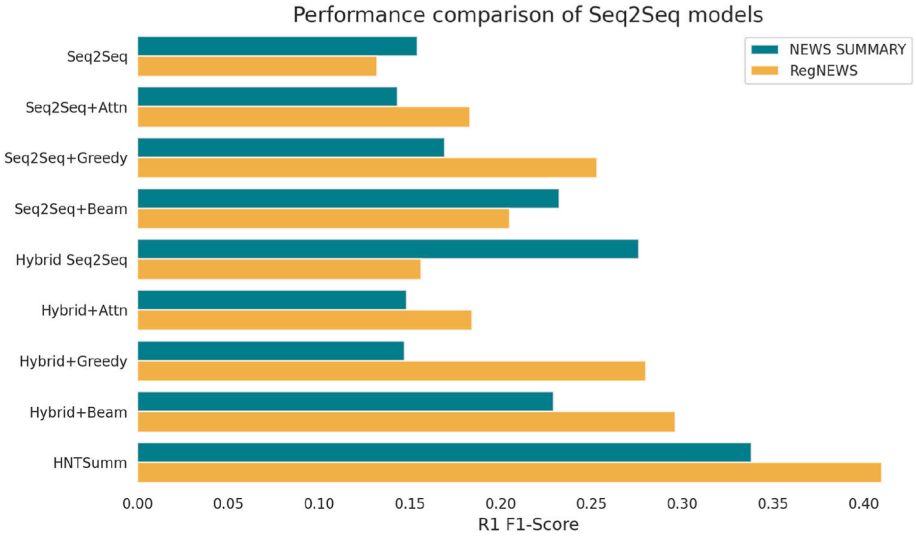

Fig. 5. Performance comparison of Seq2seq models with various strategies and mechanisms.

interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] M. Allahyari, S. Pouriyeh, M. Assefi, S. Safaei, E.D. Trippe, J.B. Gutierrez, K. Kochut, Text Summarization Techniques: A Brief Survey, July 7, 2017, https://doi.org/10.48550/arXiv.1707.02268.

[2] A.S. Uban, C. Caragea, Generating summaries for scientific paper review. https://doi.org/10.48550/arXiv.2109.14059, September 28, 2021.

[3] W. Zhang, J.C. Kit Cheung, J. Oren, Generating character descriptions for automatic summarization of fiction, in: Proceedings of the AAAI Conference on Artificial Intelligence, 33, July 17, 2019, pp. 7476–7483, https://doi.org/10.1609/aaai.v33i01.33017476, 01.

[4] A. Mabrouk, R.P.D. Redondo, M. Kayed, SEOpinion: summarization and exploration of opinion from E-commerce websites, Sensors 21 (2) (January 18, 2021) 636, https://doi.org/10.3390/s21020636.

[5] S. Subramanian, R. Li, J. Pilault, C. Pal, On extractive and abstractive neural document summarization with transformer language models. https://doi.org/10.48550/arXiv.1909.03186, September 7, 2019.

[6] M. Mohsin, S. Latif, M. Haneef, U. Tariq, M.A. Khan, S. Kadry, H.-S. Yong, J.-I. Choi, Improved text summarization of news articles using GA-HC and PSO-HC, Appl. Sci. 11 (22) (November 9, 2021), 10511, https://doi.org/10.3390/app112210511.

[7] I. Sutskever, O. Vinyals, Q.v. Le, Sequence to sequence learning with neural networks. https://doi.org/10.48550/arXiv.1409.3215, September 10, 2014.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need. https://doi.org/10.48550/arXiv.1706.03762, June 12, 2017.

[9] T. Hasan, A. Bhattacharjee, M.S. Islam, K. Samin, Y.-F. Li, Y.-B. Kang, M.S. Rahman, R. Shahriyar, Xl-Sum, Large-scale multilingual abstractive summarization for 44 languages. https://doi.org/10.48550/arXiv.2106.13822, June 25, 2021.

[10] A. Abu Nada, E. Alajrami, A. Alsaqqa, S. Abu-Naser, Arabic Text Summarization Using AraBERT Model Using Extractive Text Summarization Approach, 2020. http://dspace.alazhar.edu.ps/xmlui/handle/123456789/633.

[11] Y. Cao, X. Wan, J. Yao, D. Yu, MultiSumm: towards a unified model for multi-lingual abstractive summarization, Proc. AAAI Conf. Artif. Intell. 34 (April 3, 2020) 11–18, https://doi.org/10.1609/aaai.v34i01.5328, 01.

[12] S. Kundu, S. Paul, S. Pal, A deep learning based approach to transliteration, in: Proceedings of the Seventh Named Entities Workshop, Association for Computational Linguistics, Stroudsburg, PA, USA, 2018, pp. 79–83, https://doi.org/10.18653/v1/W18-2411.

[13] M. Rosca, T. Breuel, Sequence-to-Sequence neural network models for transliteration. https://doi.org/10.48550/arXiv.1610.09565, October 29, 2016.

[14] M.K. Vathsala, G. Holi, RNN based machine translation and transliteration for twitter data, Int. J. Speech Technol. 23 (3) (September 12, 2020) 499–504, https://doi.org/10.1007/s10772-020-09724-9.

[15] C.R. Dhivyaa, K. Nithya, T. Janani, K.S. Kumar, N. Prashanth, Transliteration based generative pre-trained transformer 2 model for Tamil text summarization, 2022 international conference on computer communication and informatics (ICCCI), IEEE (2022) 1–6, https://doi.org/10.1109/ICCCI54379.2022.9740991.

[16] K. Al-Sabahi, Z. Zuping, Y. Kang, Latent semantic analysis approach for document summarization based on word embeddings, KSII Transactions on Internet and Information Systems 13 (1) (January 31, 2019), https://doi.org/10.3837/tiis.2019.01.015.

[17] R. Rani, D.K. Lobiyal, A weighted word embedding based approach for extractive text summarization, Expert Syst. Appl. 186 (December 2021), 115867, https://doi.org/10.1016/j.eswa.2021.115867.

[18] R. Mihalcea, Graph-Based Ranking Algorithms for Sentence Extraction, Applied to Text Summarization, Proceedings of the ACL Interactive Poster and Demonstration Sessions, Association for Computational Linguistics, Barcelona, Spain, 2004, pp. 170–173. https://aclanthology.org/P04-3020.

[19] G. Petasis, V. Karkaletsis, Identifying Argument Components through TextRank, Proceedings of the Third Workshop on Argument Mining (ArgMining2016), Association for Computational Linguistics, Stroudsburg, PA, USA, 2016, pp. 94–102, https://doi.org/10.18653/v1/W16-2811.

[20] Y. Zhou, J. Shen, X. Zhang, W. Yang, T. Han, T. Chen, Automatic source code summarization with graph attention networks, J. Syst. Software 188 (June 2022), 111257, https://doi.org/10.1016/j.jss.2022.111257.

[21] R. Prabhavalkar, K. Rao, T.N. Sainath, B. Li, L.M. Johnson, N. Jaitly, A Comparison of Sequence-To-Sequence Models for Speech Recognition, Interspeech, 2017 https:doi.org/10.21437/INTERSPEECH.2017-233.

[22] C.-Y. Lin, ROUGE: A Package for Automatic Evaluation of Summaries, Text Summarization Branches Out, Association for Computational Linguistics, Barcelona, Spain, 2004, pp. 74–81. https://aclanthology.org/W04-1013.
[23] J.-P. Ng, V. Abrecht, Better Summarization Evaluation with Word Embeddings for ROUGE, August 25, 2015, https://doi.org/10.48550/arXiv.1508.06034.
[24] J. Pei, R. Hantach, S. ben Abbes, P. Calvez, Towards hybrid model for automatic text summarization, in: 2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA), IEEE, 2020, https://doi.org/10.1109/ICMLA51294.2020.00160, 987–93.
[25] C. Li, W. Xu, S. Li, S. Gao, Guiding generation for abstractive text summarization based on key information Guide network, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2, Association for Computational Linguistics, Stroudsburg, PA, USA, 2018, pp. 55–60, https://doi.org/10.18653/v1/N18-2009 (Short Papers).
[26] A. Ghadimi, H. Beigy, Hybrid multi-document summarization using pre-trained language models, Expert Syst. Appl. 192 (2022), 116292, https://doi.org/10.1016/j.eswa.2021.116292 from, https://www.sciencedirect.com/science/article/pii/S0957417421015979.
[27] Kondalarao Vonteru, News Summary dataset, Kaggle. https://www.kaggle.com/datasets/sunnysai12345/news-summary, 2022.
[28] D. Petrovic, S. Janicijevic, Domain specific word embedding matrix for training neural networks, in: International Conference on Artificial Intelligence: Applications and Innovations (IC-AIAI), IEEE, 2019, pp. 71–714, https://doi.org/10.1109/IC-AIAI48757.2019.00022, 2019.