# Abstractive Text Summarization and Unsupervised Text Classifier

Aditya, Akanksha Shrivastava, Saurabh Bilgaiyan

School of Computer Engineering, KIIT DU, Bhubaneswar, Odisha, India
{adikid1996, shivi.s98, saurabhbilgaiyan01}@gmail.com

**Abstract.** In this day and age, as the internet gets increasingly cluttered for content, the comprehension of generated huge texts is growingly becoming a source of major inconvenience for a normal viewer and classifying it is another gruesome task. Text summarization and classification proves to be a boon as it abridges and categorize the massive text into sizeable length without removing the key information within it and assign it to its genre. In this paper, the authors have tried to implement the abstractive type of summarization using Sequence-to-Sequence RNN. Using this model, the output perceived is a summary which is short, lossy, and whose length does not necessarily depend on source text length and then the authors have used Bidirectional LSTM to find the best one to assign a particular genre.

**Keywords:** text summarization, RNN, Sequence-to-Sequence models, lossy text, text classification, Bidirectional LSTM, Machine Learning

## I Introduction

Text summarization is the process of abridging very large texts into a concise and fluent summary while keeping the key information and overall meaning intact [1]. The area of implementation of text summarization is becoming increasingly popular because be it anything, a small institution, a hefty website, a business firm, all require shortening of the vast texts into readable summaries [2, 3]. It could be anywhere from machine translation to speech recognition, from image captioning to language identification, etc. [4].

Text Summarization are of two types: Extractive Summarization and Abstractive Summarization [3]. In extractive summarization, important sections of the text are identified and then a summary is generated which is a subset of the sentences from the original text [3]. It has been a widely researched subject and people have extensive work in this area and has almost reached its maturity stage [1]. The world is now turning towards abstractive text summarization [1]. In contrast to extractive summarization, abstractive approach involves understanding the intent and writes the summary in one's words, meaning rephrasing of words is used.

Abstractive summarization is further classified into Structure-Based, Semantic-Based, Deep-Learning with Neural Network Based and Discourse and Rhetoric Based Structure Based Summarization [1, 3].

In this paper, we have attempted Abstractive Text Summarization with Attention Mechanism on Sequence-to-Sequence model with LSTM. A question could be why not just use Machine Translation Sequence to Sequence model [5]. The problem with Machine Translation Sequence to Sequence model is that the target summary does not necessarily depend on source length [1, 5]. It varies according to the length of the source text. Also, the summary the authors got in Sequence-to-Sequence RNN model is lossy or so as to say the source text content is partially discarded and inexactly approximated to encode data [5].

Next, Text Classification is the process of assigning the text into one or better-defined categories. It is one of the most widely and extensively used natural language processing applications [6]. Some of the examples of text classification are understanding audience sentiment from social media, detection of spam and non-spam emails, auto tagging of customer queries, categorization of news articles into defined topics, health related classification using social media in public health surveillance, language identification, etc. [6, 7]. There are several types of text classification methods which include: (1) Support Vector Machines, (2) Naive Bayes, (3) Rough Set-Based Classifier, (4) Instantaneously Trained Neural Networks, etc. [6, 7].
In this paper the authors have implemented Bidirectional LSTM to classify processed summary and assign its genre or category [6]. The authors have divided the content into the following parts— (2) Section 2 contains the explanation modus operandi by the authors and the abstract mentioning of the all previous work done in this particular topic. (3) Section 3 contains an explanation of the working of attention mechanism in sequence-to-sequence RNN and the algorithm used to implement the model. (4) Section 4 contains a brief description of Bidirectional RNN model implemented. (5) Section 5 is used this section to describe the code for Rogue’ s Summary and showcase their model accuracy. (6) Section 6 is describing how the program functions on the whole. (7) Section 7 describes the conclusion and future improvements which could be implemented further to improve the current model used by the authors. (8) Section 8 contains mentioning of all the papers which were referred by the authors.

## II Related Works

The authors are proposing a model for text classification and summarization of huge text using Sequence-to-Sequence RNN and Bidirectional LSTM. There are many researches already done in this field but the task was to merge both the broad scenarios and provide results at one singular place.

The paper also elucidates several explanations about Text Summarization. The work by Gupta, et al. is an overview in which they have clarified the facts about abstractive text summarization [1]. They have thoroughly demonstrated the broad categories of techniques of Text Summarization and have explained them in-depth [2, 3]. The paper also infers similar work done by Islam et al. in which they used Sequence-to-Sequence RNN to translate Bangla text [4]. They have furthered their study LSTM and have worked to implement this technique in Python framework [4]. The work in this paper is largely inspired work from the paper of Nallapati et al. where they proposed the system to Attention Encoder-Decoder RNN, and achieve state-of-the-art performance and were able to address many critical problems in summarization [5]. Azmi et al. have implemented this to convert Arabic text using user granularity [8]. The text is segmented text topic wise and applied to an enhanced extractive summarizer inferred to as rules-based sentence reduction technique or RST- based extractive summarizer and the length of the extract summary was modified to control the size of predicted summary. The results were better in comparison to an abstractive model. In the work of Bhargava et al., the authors have implemented the sentiment infusion to implement abstractive text summarization [9]. The work done by Sahoo et al. implements sentence clustering using Markov clustering principle followed by sentence ranking [10]. The then top ranked sentences from the clusters are worked through linguistic rules to form the summary. Text summarization was also implemented in the work of Negi et al. in which the authors have used NLP and ML to implement a mix of syntactic featuring to comprehend the writing style of incoming reports for supporting timely report monitoring and perceive authoritative statements [11].
On the other hand, work by Mirończuk et al. demonstrates the type of text classification techniques available and how can they be used [6]. Liu et al. have worked to implement bidirectional LSTM with attention mechanism and convolutional layer to classify text [7]. This is another approach for classifying text which provided better results than normal supervised techniques. The semantic text classification done by Altinel et al. demonstrates advantages of semantic text classification over text classification which included extraction and using in-words latent relationship and textual semantic understanding, etc [12]. The work also compares performance comparison of knowledge-based approaches [13].

## III. Summarization Algorithm

### A. Model for the Attention Mechanism in Sequence-to-sequence RNN

The sequence-to-sequence model consists of an encoder, a decoder, and an attention mechanism.

**Encoder-** It is bi-directional Long Short-term Memory (LSTM) layer that extracts information from the original text. It reads one word at a time and as it is an LSTM, it updates hidden state based on the current word and the words it has read before [1]. In short, all encoder does is it takes the input data, and passes the last state of its recurrent layer as an initial state to the decoder' s first layer (recurrent).

**Decoder-** It is Unidirectional Long Short-term Memory (LSTM) that summaries one word at a time. The decoder LSTM starts working once it gets signaled that the entire source text has been read. It creates a probability distribution over the next word using information from the encoder and previously written text [1]. In short, the decoder takes the last state of the encoder's last recurrent layer and uses it as its first recurrent layer' s initial state.

**Attention Mechanism-** Though this mechanism, the decoder can access the intermediate hidden states in the encoder and use all that information to decide which word is next [3].

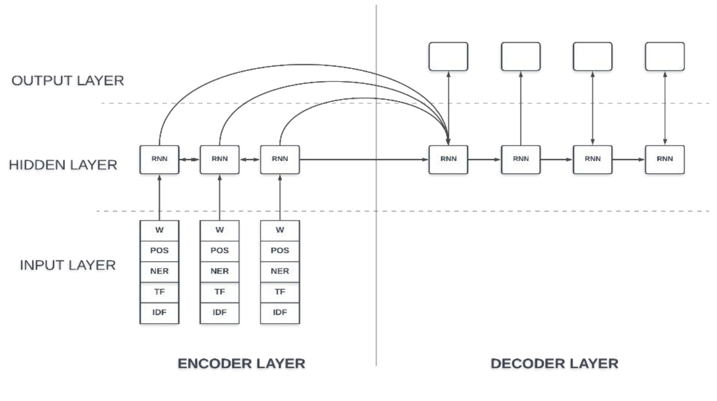

**Fig. 1.1.** Attention Mechanism for Sequence-to-Sequence RNN

The algorithm is implemented in the following the steps:

1. The input text is encoded and the decoder is initialized with internal states of the encoder. The <start> token is passed as an input to the decoder.

2. The decoder is executed for one-time step with the internal states.
3. The output will be the probability for the next word. The word with the maximum probability will be selected.
4. The sampled word is passed as an input to the decoder in the next time step and the internal states is updated with the current time step.
5. The steps 3-5 is iterated until <end> token is generated to the maximum length of the target sequence is hit.

## IV. Classifier Algorithm

The algorithm is implemented in the following steps:

1. First the input layer is provided.
2. After adding the first embedding layer, a bidirectional LSTM layer is added.
3. The output layers are then added, i.e., Dense (activation = "ReLU"), Dropout, Dense (activation = "sigmoid") in consecutive order.
4. The optimizer used to train the model is Adam's optimizer and loss parameter is 'binary_crossentropy'.

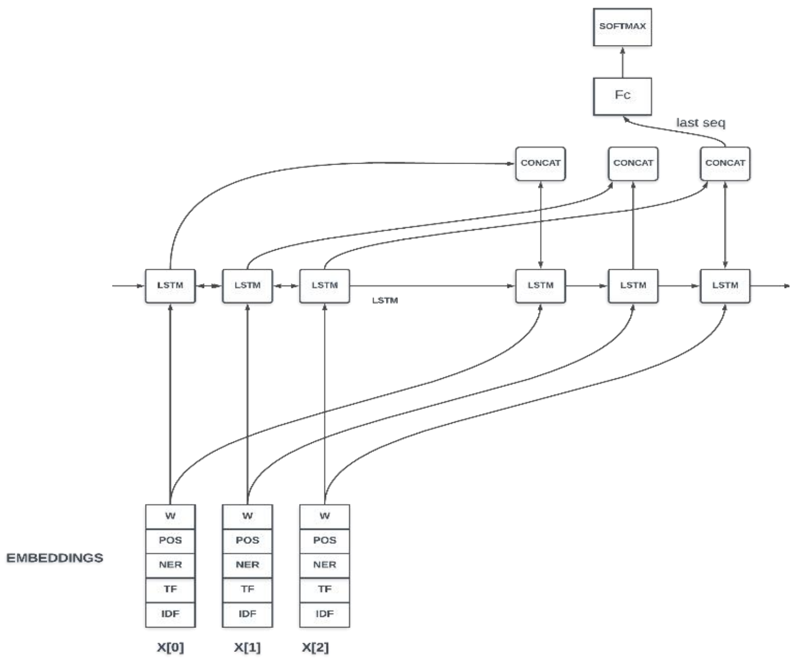

Fig. 2.2. Bidirectional LSTM

## V. Experimental Studies

**Performance benchmarks:**

The authors are using Rouge summary to compare the efficiency between their model and presently extractive summarization model.
ROUGE is an acronym which stands for Recall-Oriented Understudy for Gisting Evaluation. It is actually a system of measurement for appraising automatic summarization of texts. There are two score sets in ROUGE.

$$\text{Precision} = \frac{\textbf{number\_of\_overlapping\_words}}{\textbf{total\_words\_in\_reference\_summary}}$$

(1)

$$\text{Recall} = \frac{\textbf{number\_of\_overlapping\_words}}{\textbf{total\_words\_in\_system\_summary}}$$

(2)

```
Precision is: 0.38095238095238093
Recall is: 0.25
```

```
F Score is: 0.30188774460662915
Sum of ROUGE Score: 14.068020212266791
Average ROUGE Score = 0.3271632607503905
Count: 43
```

## V. Results and Discussion

The authors in this manuscript have used amazon-fine-food-reviews dataset to train the Sequence-to-Sequence model of Summarizer, then the results outcome of the model is used on Classifier to give the genre as well.

The program works in the following manner:
For Text Summarizer:-

```
Using TensorFlow backend.
```

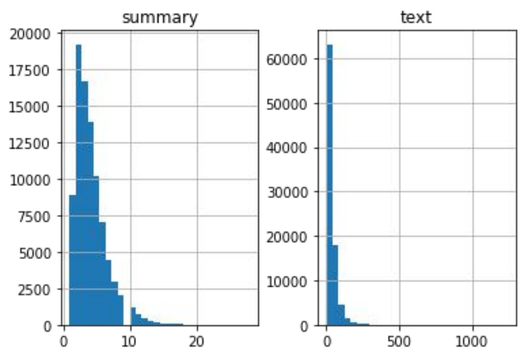

**Fig 2.1** Diagnostic plot to observe the distribution of length over the text

The length of the reviews and the summary for an idea about the distribution of length of the text.

```
Generating the proportion of length of the summaries:
0.9424907471335922
```

This means 94% of the summaries have length below 8
So, we can fix the maximum length of summary.

Preparing the text tokenizer:

```
% of rare words in vocabulary: 66.12339930151339
Total Coverage of rare words: 2.953684513790566
```

```
Preparing the summary tokenizer:
% of rare words in vocabulary: 78.12740675541863
Total Coverage of rare words: 5.3921899389571895

Model building:
_________________________________________________________________
__________________________________
Layer (type)                    Output Shape         Param #
Connected to
=================================================================
==================================
input_1 (InputLayer)            (None, 30)           0
_________________________________________________________________
__________________________________
embedding (Embedding)           (None, 30, 100)      844000
input_1[0][0]
_________________________________________________________________
__________________________
lstm (LSTM)                     [(None, 30, 300), (N 481200
embedding[0][0]
_________________________________________________________________
__________________________________
input_2 (InputLayer)            (None, None)         0
_________________________________________________________________
__________________________________
lstm_1 (LSTM)                   [(None, 30, 300), (N 721200
lstm[0][0]
_________________________________________________________________
__________________________________
embedding_1 (Embedding)         (None, None, 100)    198900
input_2[0][0]
_________________________________________________________________
__________________________________
lstm_2 (LSTM)                   [(None, 30, 300), (N 721200
lstm_1[0][0]
_________________________________________________________________
__________________________________
lstm_3 (LSTM)                   [(None, None, 300),  481200
embedding_1[0][0]
lstm_2[0][1]
lstm_2[0][2]
_________________________________________________________________
__________________________________
attention_layer (AttentionLayer [(None, None, 300),  180300
lstm_2[0][0]
lstm_3[0][0]
_________________________________________________________________
__________________________________
concat_layer (Concatenate)      (None, None, 600)    0
lstm_3[0][0]
attention_layer[0][0]
```

```
__________________________________________________________________
__________________________________
time_distributed (TimeDistribut (None, None, 1989)   1195389
concat_layer[0][0]
==================================================================
==================================
Total params: 4,823,389
Trainable params: 4,823,389
Non-trainable params: 0
__________________________________________________________________
__________________________________
Train on 41346 samples, validate on 4588 samples
Epoch 1/50
41346/41346 [==============================] - 85s 2ms/sample -
loss: 2.8152 - val_loss: 2.5780
Epoch 2/50
41346/41346 [==============================] - 79s 2ms/sample -
loss: 2.4859 - val_loss: 2.4072
==================================================================
…Epoch 19/50
41346/41346 [==============================] - 80s 2ms/sample -
loss: 1.7070 - val_loss: 2.0398
Epoch 00019: early stopping
```

In the following plotted, we can see the validation loss has incremented after 17th epoch and kept increasing for two epochs. So, we used early stopping at 19th epoch.

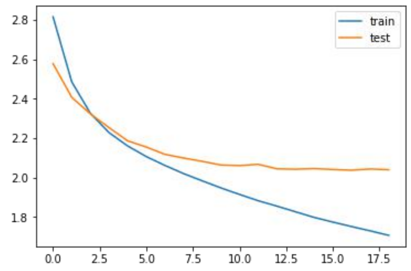

**Fig 2.2** Diagnostic plot to observe the behavior of the model

Here a few summaries generated by the model:

```
Review: gave caffeine shakes heart anxiety attack plus tastes
unbelievably bad stick coffee tea soda thanks
Original summary: hour
Predicted summary:  not worth the money

Review: got great course good belgian chocolates better
Original summary: would like to give it stars but
Predicted summary:  good

Review: one best flavored coffees tried usually like flavored
coffees one great serve company love
```

```
Original summary: delicious
Predicted summary:  great coffee

Review: salt separate area pain makes hard regulate salt putting
like salt go ahead get product
Original summary: tastes ok packaging
Predicted summary:  saltReview: really like product super easy
order online delivered much cheaper buying gas station stocking
good long drives
Original summary: turkey jerky is great
Predicted summary:  great product

Review: best salad dressing delivered promptly quantities last
vidalia onion dressing compares made oak hill farms sometimes
find costco order front door want even orders cut shipping costs
Original summary: my favorite salad dressing
Predicted summary:  great product
```

For Text Classifier: -

```
Epoch 1/1
10500/10500 [==============================] - 48s 4ms/step -
loss: 0.6389
Bidirectional-RNN, Word Embeddings 0.4318
```

## VII. Conclusion and Future Work

In this work, the authors applied Attention Mechanism on RNN to find the best possible summary. The model’s output was then transferred to bidirectional LSTM classifier model to assign a category. The former model could be further improved by using beam-strategy for decoding the test sequence. The accuracy could also be measured using BLEU score and pointer-generator networks and coverage mechanisms could be used to handle unique words. Then the classifier model can also be improved by using Hierarchical Attention Networks or using CNNs and RNNs with a greater number of layers.

## References

1. Gupta S., Gupta S. K., “Abstractive summarization- An overview of the state of the art”, Expert Systems with Applications, Elsevier, 121(1) (2019), 49-65.
2. Mahajani A., Pandya V., Maria I., Sharma D., “A Comprehensive Survey on Extractive and Abstractive Techniques for Text Summarization”, Ambient Communications and Computer Systems(Part of the Advances in Intelligent Systems and Computing), Springer, 904(1) (2019), 339-351.

3. Gambhir M., Gupta V., “Recent automatic text summarization techniques: a survey”, Artificial Intelligence Review, Springer, 47(1) (2017), 1–66.
4. Islam S., Mousumi S. S. S., Abujar S., Hossain S. A., “Sequence-to-sequence Bangla Sentence Generation with LSTM Recurrent Neural Networks”, International Conference on Pervasive Computing Advances and Applications – PerCAA 2019, Procedia Computer Science, Elsevier, 152(1) (2019), 51-58.
5. Nallapati R., Zhou B., Santos C., Gulçehre Ç., Xiang B., “Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond”, The SIGNLL Conference on Computational Natural Language Learning (CoNLL), (2011), 1-12.
6. Mirończuk M. M., Protasiewicz J., “A recent overview of the state-of-the-art elements of text classification,'' Expert Systems with Applications, 106(1) (2018), 36-54.
7. Liu G., Guo J., “Bidirectional LSTM with attention mechanism and convolutional layer for text classification”, Neurocomputing, 337(1), ( 2019), 325-338.
8. Azmi A. M., Altmani N. L., “An abstractive Arabic text summarizer with user controlled granularity”, Information Processing & Management, Elsevier, 54(6) (2018), 903-921.
9. Bhargava R, Sharma Y., Sharma G., “ATSSI: Abstractive Text Summarization using Sentiment Infusion”, Twelfth International Multi-Conference on Information Processing, (2016), 1-8.
10. Sahoo D., Bhoi A., Balabantaray R. C., “Hybrid Approach to Abstractive Summarization”, International Conference on Computational Intelligence and Data Science (ICCIDS 2018), (2018), 1-10.
11. Negi K., Pavuri A., Patel L., Jain C., “A novel method for drug-adverse event extraction using machine learning”, Informatics in Medicine Unlocked, Elsevier, In Press, Corrected Proof, (2019), 1-6.
12. Altinel B., Ganiz M. C., “Semantic text classification- A survey of past and recent advances”, Information Processing & Management, 54(6) (2018), 1129-1153.
13. Harish B. S., Udayasri B., “Document Classification: An Approach Using F eature Clustering”, Recent Advances in Intelligent Informatics (Part of the Advances in Intelligent Systems and Computing book series (AISC)), 235(1) (2014), 163-173.