International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 05 | May 2020 www.irjet.net p-ISSN: 2395-0072

Deep Learning Approach for Text Summarization

Mihir Vaidya[1], Varad Ahirwadkar[2], Vaibhav Pale[3], Vivek Patil[4], Anjali Shejul[5]

[1]Student, Information Technology Department, MIT- College of Engineering, Pune, Maharashtra, India,
[2]Student, Information Technology Department, MIT- College of Engineering, Pune, Maharashtra, India,
[3]Student, Information Technology Department, MIT- College of Engineering, Pune, Maharashtra, India,
[4]Student, Information Technology Department, MIT- College of Engineering, Pune, Maharashtra, India,
[5]Assistant Professor, Information Technology Department, MIT- College of Engineering, Pune, Maharashtra, India

***

Abstract - *Text summarization refers to the technique of cropping long pieces of text. Neural network models have been provided a new feasible approach for abstractive text summarization. Abstractive means generating new sentences from the original text which might not be present in the original text. These neural network models have two defects: They are likely to reproduce factual details inaccurately and they tend to repeat themselves. In this work we propose a novel design that augments the standard sequence-to-sequence attentional model in two orthogonal ways. Primarily, we employed a pointer-generator network that selects words from the given text by pointing, which helps accurate replication of data, keeping the ability to introduce new words by the means of the generator. Secondly, we use a coverage mechanism to keep track of what has been summarized, which discontinues repetition. We apply our model to CNN/Daily mail dataset to perform training the model. Accuracy of the generated summary is checked by ROUGE points. The system provides additional features such as downloading the generated summary in PDF file as well as text-to-speech conversion.*

Key Words: **sequence-to-sequence attentional model, Abstractive summarization, Extractive summarization, pointer-generator network.**

1. INTRODUCTION

The main idea behind automatic text summarization is to be able to find a short subset of the most essential information from the entire set and present it in a human-readable format. Exponentially increase of data, brought the need for summarization. There are two broad categories of summarization that are extractive and abstractive. The extractive method identifies the important sentences or phrases from the original text and extracts only those sentences from the text while Abstractive methods may form the new sentences which are initially not available in the original text, it restricts to selecting and rearranging sentences-as a human-generated summary. The extractive approach is easier as compared to the abstractive approach because it copies the sentences from the source document and makes sure baseline levels of grammatically and accuracy.

Owing to the difficulty caused in developing abstractive summarization, mostly the previous works have been done using the extractive summarization techniques [1]. Lately, with the help of a sequence-to-sequence model, we can read as well as freely generate the text by virtue of recurrent neural networks(RNNs) to create an abstractive summary of the given text [2][3].

We apply our model (CNN model) on the Daily Mail dataset, which contains news articles (39 sentences on average). The output of this model results with a multi-sentence summary. It is observed that we can increase the performance of the abstractive system by ROUGE scores, using the rouge package in python.

Our model is a hybrid pointer generator network that allows extracting the words from the source text [4] and also is capable of generating new words. The ability of the system to copy words from the source text not only improves its accuracy but also handles the words (keywords) which are not in the vocabulary. For short text summarization, [5] Forced-Attention sentence compression is applied. To control and track coverage of source document, we have used a coverage vector from the Neural Translation. It results in the reduction of the repetitive summary that is generated [6].Fig-1 shows the typical pointer-generator model.

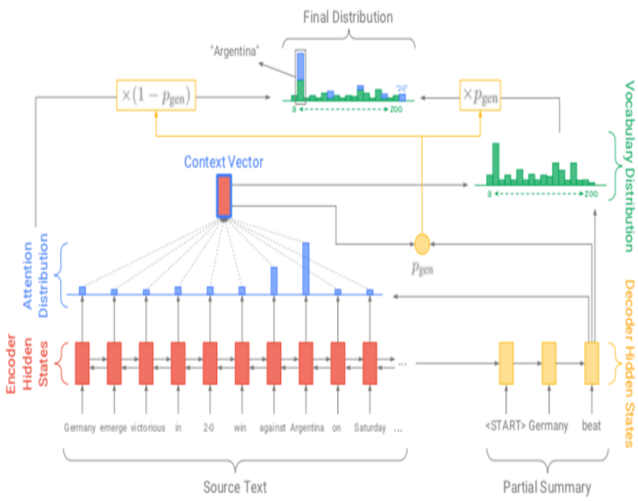

Fig -1: Pointer-generator model [15]

2. LITERATURE REVIEW

Prior to the advent of neural networks in the field of text summarization, automatic summarizers usually relied on sentence extraction rather than sentence abstraction [7][8]. This approach was recently re-explored by R. Mihalcea et. al. [8], who proposed 'TextRank', a novel graph-based ranking model for processing of text. Jing et. al. [9] explored sentence reduction, i.e., removing unwanted phrases from the extracted sentences.

Rush et al. [2] used an attention-based neural network model for abstractive text summarization. This approach was augmented by S. Chopra et al. [10], using recurrent neural networks and Y Kim et al. [11] using convolutional neural networks which processed input at a character-level instead of the traditional word-level ones. S. Dohare [12] proposed a pipeline that forms an Abstract Meaning Representation graph of the input, then converts it into a summary graph, out of which summary sentences are finally extracted as output. L Song [13] used a graph-to-sequence model and I. Konstas [14] used a sequence-to-sequence model for generating summary sentences from Abstract Meaning Representation (AMR). K Liao et al. [14] used AMR for the summarization of multiple documents. Nallaptti et al. [3] used attentional encoder-decoder RNN for abstractive text summarization and proposed several models for solving critical problems of sentence-to-word hierarchy, emitting uncommon words and keyword modeling.

Hybrid models based on pointer generator networks have been introduced extensively for text summarization. The architecture introduced by A. See et al. [15] outperformed other state-of-the-art models by 2 ROUGE points. X. Jiang et al. [16] used pointer generator networks for context-aware, topic-oriented text summarization. Yan Zhao et al. [17] proposed a reinforcement learning-based pointer generator network for text summarization and generation of story endings.

3. PROPOSED SYSTEM

Our system is a web Text Summarization application built using the Flask server. In this summarization process, the application takes inputs in two formats viz. text (direct upload or file upload), and the textual image. If it is a textual image then the process begins with converting an image file into the text file by applying it to the OCR model. The text file will be pre-processed. This text or the input given in the textual format is primarily converted into a binary file that is further processed. This binary file is fed as an input to the core model. After processing, the summary is produced, which is basically obtained in a textual format. The user can convert this summary into formats like PDF or in the audio (vocal translation of the summary) which they can download later.

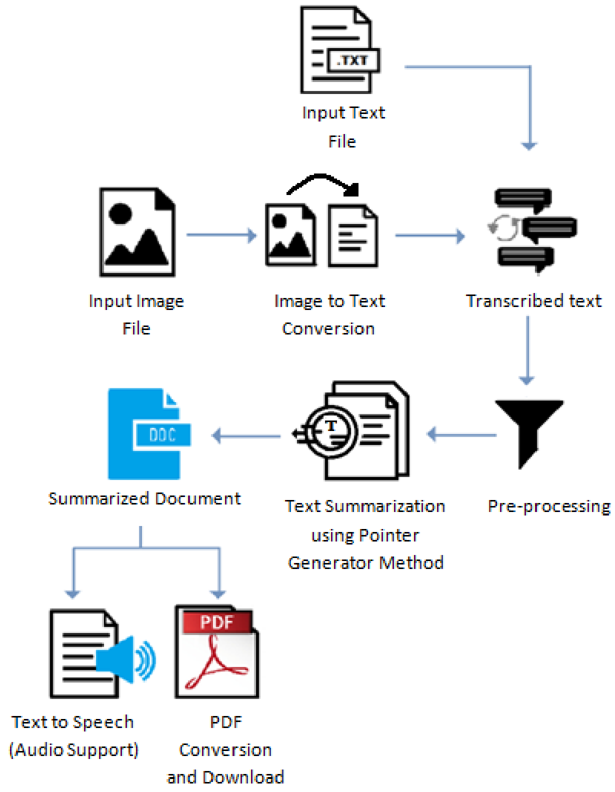

Fig -2: System Architecture

4. CORE MODEL

Our model is similar to sequence to sequence attention model which was introduced by Nallapati et al. [3]. This model has a single layer bidirectional LSTM encoder and a single layer unidirectional LSTM decoder. The encoder is fed with the token of the article w_i which generates a sequence of encoder hidden states h_i. On each time step t, the decoder receives the word embedding of previous words and has a decoder state s_t. Attention distribution is calculated as similar in Bahdanau et al. [18].

$$e_i^t = v^T \tanh(W_h h_i + W_s s_t + b_{attn}) \quad (1)$$
$$a^t = softmax(e^t) \quad (2)$$

Where v, W_h, W_s and b_{attn} are learnable parameters.

Attention distribution is used by the decoder for where to look to produce next word and used to produce fixed size representation context vector h_t^* which is the weighted sum of encoder hidden states,

$$h_t^* = \sum_i a_i^t h_i \quad (3)$$

Then probability distribution is calculated by concatenating the decoder hidden state s_t and context vector h_t^* and passed through two linear layers. It is the distribution all over vocabulary and used to predict the word.

$$P_{vocab} = softmax(V'(V[s_t, h_t^*] + b) + b') \quad (4)$$
$$P(w) = P_{voacb}(w) \quad (5)$$

Where V, V', b and b' are learnable parameters.

In the pointer generator model, it permits the copying of words from source text [15][4] as well as generating word from fixed vocabulary p_{gen} is calculated for each time step from encoder hidden state, context vector, and decoder input.

$$p_{gen} = \sigma(w_{h^*}^T h_t^* + w_s^T s_t + w_x^T x_t + b_{ptr}) \quad (6)$$

Where vectors w_{h^*}, w_s, w_x and scalar b_{ptr}.

For each decoder time step t. generation probability, $p_{gen} \in [0,1]$ is calculated which determines copying of word from source text by sampling from attention distribution or generating word from vocabulary by sampling from the probability distribution. Therefore, the probability distribution for this extended vocabulary is,

$$P(w) = p_{gen} * P_{vocab}(w) + (1 - p_{gen}) \sum_{i:w_i=w} a_i^t \quad (7)$$

It has the ability to produce out of vocabulary words by copying from source text but it has disadvantages of predefined vocabulary. If w is an out of vocabulary then P_{vocab} is zero else w does not appear in the source document, then $\sum_{i:w_i=w} a_i^t$ is zero.

Repetition of words or phrases is a pretty common issue with sequence-to-sequence models and especially when used for multi-sentence or long sentence text summarization [6][19][20]. To solve these issues we adopted the coverage model of Tu et al. [6]. Here we maintain a coverage vector c^t, which is the summation of attention distributions of all the previous decoder timesteps output.

$$c^t = \sum_{t'=0}^{t-1} a^{t'} \quad (8)$$

c^t is a probabilistic distribution of the words that shows the degree of coverage of words that have been received from the attention mechanism. The coverage vector is then used as an extra input to the attention mechanism, which changes the equation to:

$$e_i^t = v^T \tanh(W_h h_i + W_s s_t + w_c c_i^t + b_{attn}) \quad (9)$$

This ensures that the Attention mechanism doesn't repeatedly attend an equivalent word or phrases by remembering the previous decision made (previously covered phrases), thus it avoids repetitive text. Our loss function is more flexible because summarization shouldn't require uniform coverage; we only penalize the overlap between each attention distribution and also the coverage to date to forestall repeated word phrases. Finally, the coverage loss, reweighted by some hyper parameter λ, is included in the attention mechanism loss function to introduce a new composite loss function:

$$loss_t = -\log P(w_t^*) + \lambda \sum_i \min(a_i^t, c_i^t) \quad (10)$$

5. DATASET

We have used CNN/Daily Mail Dataset for our project. This dataset contains news articles along with the multiple sentence summaries. Each news article contains around 781 tokens. We have used this dataset and operated directly on original text which is non-anonymized. In all, this corpus has 286,817 training pairs, 13,368 validation pairs, and 11,487 test pairs.

6. RESULTS

We performed our execution on the sequence to sequence attention with a coverage mechanism having 256-dimensional hidden states and 128-dimensional word embeddings. The results are given in Table. We evaluated our model by using standard ROUGE metric, reporting the F1 scores for ROUGE-1, ROUGE-2, and ROUGE-L. We obtain our ROUGE scores using the rouge package. Table 1 shows the ROUGE scores.

Table -1: Results obtained

Model	ROUGE (F1 Score)		
	ROUGE 1	ROUGE 2	ROUGE L
Pointer Generator + Coverage	39.40	16.78	35.08

7. CONCLUSION

In this work, we presented hybrid pointer-generator architecture with coverage and showed that it reduces inaccuracies and repetition. We applied our model to a new and challenging long- text dataset and significantly outperformed the abstractive state-of-the-art result. Our model displays many abstractive abilities, but achieving higher levels of abstraction remains an open research question.

REFERENCES

[1] Horacio Saggion and Thierry Poibeau, "Automatic text summarization: Past, present, and future", Multi-source, Multilingual Information Extraction and Summarization, Springer, 2013, pp 3–21.

[2] Alexander M. Rush, Sumit Chopra, Jason Weston, "A Neural Attention Model for Abstractive Sentence Summarization", arXiv preprint arXiv:1509.00685, 3 Sept 2015.

[3] Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, C¸aglar Gulc¸ehre, and Bing Xiang, "Abstractive text summarization using sequence-to-sequence RNNs and beyond", Computational Natural Language Learning, 2016.

[4] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly, "Pointer networks", Neural Information Processing Systems,2015.

[5] Yishu Miao and Phil Blunsom, "Language as a latent variable: Discrete generative models for sentence compression", Empirical Methods in Natural Language Processing, 2016.

[6] Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li, "Modeling coverage for neural machine translation", Association for Computational Linguistics, 2016.

[7] F. Kyoomarsi, H. Khosravi et. al, "Extraction-Based Text Summarization Using Fuzzy Analysis", Iranian Journal of Fuzzy Systems, Vol. 7, No. 3, 2010, pp. 15-32.

[8] Rada Mihalcea, Paul Tarau, "TextRank: Bringing Order into Texts", Department of Computer Science University of North Texas.

[9] Hongyan Jing," Sentence Reduction for Automatic Text Summarization", Department of Computer Science, Columbia University, New York.

[10] Sumit Chopra, Michael Auli, Alexander M. Rush, "Abstractive Sentence Summarization with Attentive Recurrent Neural Networks", Proceedings of NAACL-HLT, San Diego, California, June 12-17, 2016, pp 93–98.

[11] Yoon Kim, Yacine Jernite, David Sontag, Alexander M. Rush, "Character-Aware Neural Language Models", Association for the Advancement of Artificial Intelligence,2016, pp 2741-2749.

[12] Shibhansh Dohare, Harish Karnick, Vivek Gupta, "Text Summarization using Abstract Meaning Representation", arXiv preprint arXiv:1706.01678, 17 July 2017.

[13] Linfeng Song, Yue Zhang, Zhiguo Wang, Daniel Gildea "A Graph-to-Sequence Model for AMR-to-Text Generation", arXiv preprint arXiv:1805.02473, 27 Aug 2018.

[14] Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, Luke Zettlemoyer, "Neural AMR: Sequence-to-Sequence Models for Parsing and Generation", arXiv preprint arXiv:1705.08381, 18 Aug 2017.

[15] Abigail See, Peter J. Liu, Christopher D. Manning, "Get To The Point: Summarization with Pointer-Generator Networks", arXiv preprint arXiv:1704.04368, 25 Apr 2017.

[16] Xiaoping Jiang, Po Hu, Liwei Hou, Xia Wang, "Improving Pointer-Generator Network with Keywords Information for Chinese Abstractive Summarization", CCF International Conference on Natural Language Processing and Chinese Computing, 2018, pp 464-474.

[17] Yan Zhao, Lu Liu, Chunhua Liu, Ruoyao Yang, Dong Yu, "From Plots to Endings: A Reinforced Pointer Generator for Story Ending Generation", CCF International Conference on Natural Language Processing and Chinese Computing, 2018, pp 51-63.

[18] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, " Neural machine translation by jointly learning to align and translate", International Conference on Learning Representations, 2015

[19] Haitao Mi, Baskaran Sankaran, Zhiguo Wang, and Abe Ittycheriah, "Coverage embedding models for neural machine translation", Empirical Methods in Natural Language Processing, 2016.

[20] Baskaran Sankaran, Haitao Mi, Yaser Al-Onaizan, and Abe Ittycheriah, "Temporal attention model for neural machine translation", arXiv preprint arXiv:1608.02927, 2016