News Text Summarisation

Svachuta Siva Sai Krishna Prasad Gollavilli, M.S. Data Science
University of New Haven
sgoll10@unh.newhaven.edu

Sravan Kumar Reddy, Pebbeti, M.S. Data Science
University of New Haven
spebb1@unh.newhaven.edu

Under the guidance of
Prof. Vahid Behzadan, Ph.D.
Department of Electrical and Computer Engineering and Computer Science
Tagliatela College of Engineering
University of New Haven

Abstract

Text summarisation comes under the domain of Natural Language Processing (NLP), which entails replacing a long, precise and concise text with a shorter, precise and concise one. Manual text summarising takes a lot of time, effort and money and it's even unfeasible when there's a lot of text. Much research has been conducted since the 1950s and researchers are still developing Automatic Text Summarisation (ATS) systems. In the past few years, lots of text-summarisation algorithms and approaches have been created. In most cases, summarisation algorithms simply turn the input text into a collection of vectors or tokens. In this study, we aim to explore and compare various neural sequence-to-sequence models for abstractive text summarization. These models are trained on the CNN/Daily Mail dataset, a large dataset containing over 1 million news articles. The experimental setup includes several key components such as data pre-processing, model architecture, training settings, and evaluation metrics. We report our findings, which demonstrate the effectiveness of advanced sequence-to-sequence models for abstractive text summarization. This research also suggests that incorporating attention mechanisms can significantly enhance the quality of the generated summaries. We compare our results with the state of the state of the art model T5 calculate the Rouge scores of the pretrained model to get a better insight and finally, we provide insights into future research directions in the field of neural summarization models.

Introduction

The term “natural language” pertains to a language that humans employ as their primary means of communication. NLP, or natural language processing, involves the examination of linguistic information through computational intelligence, often in the form of textual data such as articles. Natural language processing addresses the inherent ambiguities present in languages. To effectively handle these ambiguities, NLP requires components like knowledge representation, grammatical properties, and a set of grammatical rules, along with supplementary information like synonyms and abbreviations. By applying linguistic principles, NLP seeks to construct a textual representation that imparts structure to the typically unstructured nature of natural language.

For over five decades, the field of NLP has been actively exploring text summarization. Text summarization entails the extraction of vital details from a source text document to generate a concise yet informative version. Applications like field stream digest and the creation of news headlines illustrate instances where automatic text summarization proves particularly beneficial. The numerous features of automatic text summarization can be categorized based on their objectives (generic, domain-specific, or query-based), source type (single or multi-document), and return type (extractive and abstractive). Given the exponential growth of textual information on the internet, scholars have pro-posed various text summarization algorithms. It’s crucial to acknowledge that summaries can be either extractive or abstractive, with the extractive process identifying and generating important text passages.

Self-supervised learning methods like BART have shown promise for generating text summaries with good results. BART is a pre-trained model that combines bi-directional and autoregressive transformers. It works by corrupt-ing text documents and then training the model to maximize the reconstruction loss. BART uses a transformer architecture with an encoder-decoder structure, where text sequences are input and a summary output is produced. The key components of transformers are attention layers, which modify the embedding of each token by considering the remaining input tokens. While other parts of the model scale linearly with sequence length, the attention layer in the encoder scales quadratically. The local transformer in BART’s encoder processes input sequences separately in fixed size chunks, reducing complexity to O(nkd). A major issue is that local attention prevents information flow between chunks.

BART is a denoising auto-encoder for sequence-to-sequence models used for pre-training. BART is trained by corrupting text using a random noise func-tion and learning to reconstruct the original text. BART performs well on comprehension tasks and works especially well when fine-tuned for text genera-tion [?]. Unlike traditional transformers that use Rectified Linear Unit (ReLU) activations, BART employs the Gaussian Error Linear Unit (GeLU) activation function instead [?]. BART follows the standard transformer encoder-decoder architecture [?], using 6 layers in the encoder and decoder networks, except for larger models where 12 layers are used. To train BART, documents are corrupted and the cross-entropy loss between the decoder output and original document is minimized to learn how to reconstruct the original text.

BERT, originally published by [?], was developed to allow bidirectional conditioning on all contexts across layers during pre-training. BERT can be fine-tuned by adding a single output layer to create models for various downstream tasks. Conceptually simple but empirically powerful, BERT has two key pro-cesses - pre-training and fine-tuning. Pre-training involves training the model on unlabeled data using different pre-training objectives. This allows the model parameters to be initialized from the pre-trained data before all parameters are fine-tuned on labeled data during the fine-tuning stage. So first BERT's parameters are set up through pre-training, then fully adapted during task-specific fine-tuning.

Despite the fact that encoder-only designs have performed exceptionally well on a number of Natural Language Processing tasks, pre-trained encoder-decoder transformer models like BERT [?], BART [?], and T5 performed better on automatic summarization tasks. However, these models find it difficult to handle problems with longer input sequences when scaling [?]. The Long-Former encoder decoder, or LED, is a Long-Former variant that is suggested in this regard. It uses an efficient local plus global attention pattern of the Long-Former in place of using full self-attention in the encoder portion and has both encoder and decoder transformer stacks. The decoder pays close attention to every encoded token and all previously decoded locations. Due to the high expense of pre-training LEDs, their architecture—which included the number of layers and hidden size—was chosen, and their original characteristics were borrowed from BART. LED was evaluated on the summary job with the use of the ArXiv summarizing dataset, which is based on thorough document summaries in the scientific sector. 90% of document lengths are made up of 14.5 thousand tokens. Global attention is concentrated on the first token, whereas local attention is employed with a window size of 1,024 tokens.

Model Architecture

Most competitive neural sequence transduction models have an encoder-decoder structure. Here, the encoder maps an input sequence of symbol representations (x1, ..., xn) to a sequence of continuous representations z = (z1, ..., zn). Given z, the decoder then generates an output sequence (y1, ..., ym) of symbols one element at a time. At each step the model is auto-regressive [10], consuming the previously generated symbols as additional input when generating the next.

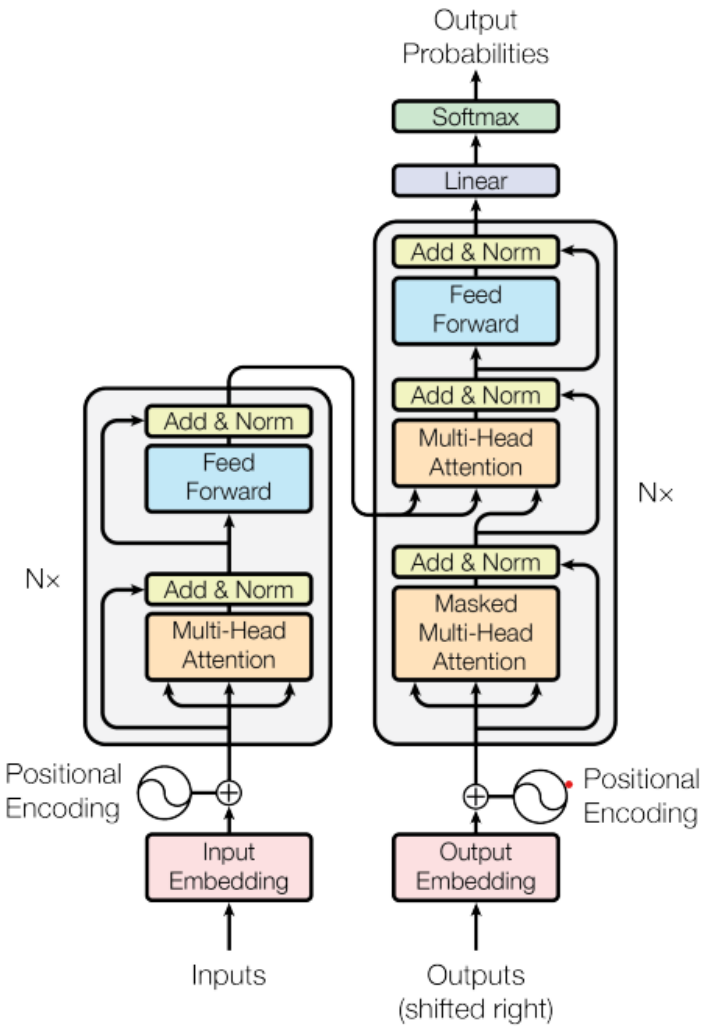

Figure 1: The Transformer - model architecture.

Encoder: N = 6 identical layers are stacked to create the encoder. Every layer comprises two sub-layers. A feed-forward network that is basic, fully connected, and position-wise is the second mechanism, while a multi-head self-attention mechanism is the first. After layer normalization [1], we use a residual connection [11] around each of the two sub-layers. In other words, each sub-layer's output is LayerNorm(x + Sublayer(x)), with Sublayer(x) representing the function that the sub-layer itself implements. In order to support these residual connections, the model's embedding layers and all of its sub-layers generate outputs with a dimension of dmodel = 512.

Decoder: Despite the fact that encoder-only designs have performed exceptionally well on a number of Natural Language Processing tasks, pre-trained encoder-decoder transformer models such as BERT [4], BART [6], and T5 performed better on automatic summarization tasks. However, these models find it difficult to handle problems with longer input sequences when scaling [9]. The Long-Former-encoder-decoder, or LED, is a LongFormer variant that is suggested in this regard. It uses an efficient local plus global attention pattern of the Long-Former in place of using full self-attention in the encoder portion and has both encoder and decoder transformer stacks. The decoder pays close attention to each encoded token and to all previously decoded locations.

The LSTM network

A possible alternative to the RNN algorithm for handling the inability of the algorithm to learn long-term dependant information is the LSTM network. A type of chain of repeating neural network modules is present in every RNN. This repeating module in the typical RNN only has a very basic structure, like a tanh layer (Xiupeng et al., 2018; Zhonghe et al., 2019). While repeated modules have a distinct structure, LSTM has the same structure. LSTM features four neural network layers that interact in a unique way, as opposed to just one.

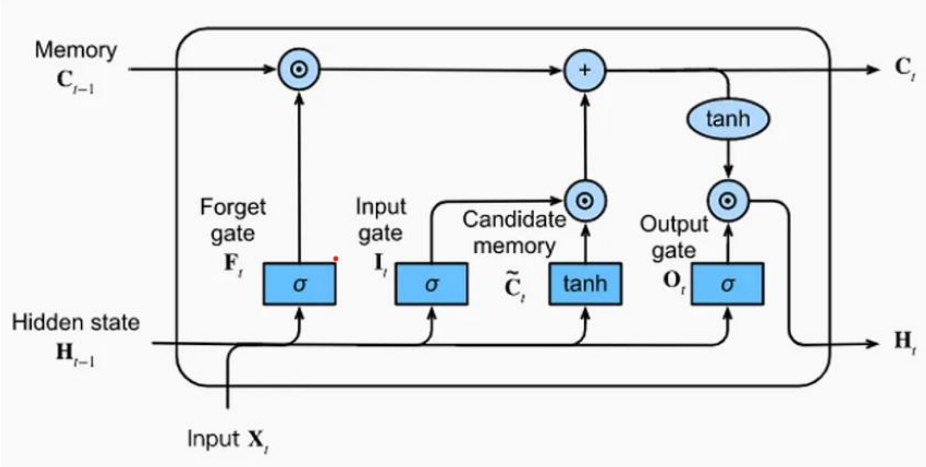

Figure 2: LSTM Unit

In actual operation, the long-term memory (cell state, C) of the LSTM unit is updated by new information received from the outside world (the input vector, X) and recent past information (the short-term memory, H). In the end, it updates the short-term memory (the hidden state, H) using the long-term memory (the cell state, C). The output of the LSTM unit in instant t is also the hidden state that was discovered in that instant. It is what the external party receives from the LSTM in order to do a particular task. Stated differently, the behaviour serves as the basis for evaluating the LSTM's performance.

Sequence-to-sequence learning and the encoder decoder framework

One of the most significant RNN variations is the Seq2Seq model, which compares input and output sequence lengths N and M. Another name for this structure is the encoder-decoder model. Sequence lengths are required for the initial N vs. N RNN. But the majority of the problem sequences we run into are not equal. For instance, in machine translation, the length of the sentences in the source and target languages frequently varies. In order to achieve this, the input data is first encoded into a context vector c using the encoder decoder structure. Another RNN network decodes it after obtaining c. The decoder is the term for this component of the RNN network. The precise procedure is to provide c as the starting state to the decoder.

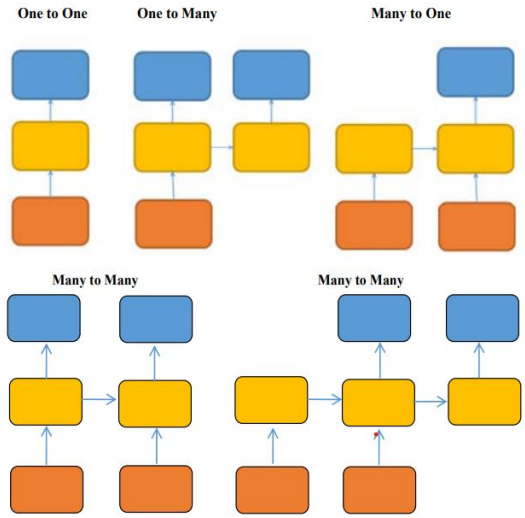

Figure 3: The encoder-decoder framework

The encoder-decoder framework operates by first encoding the input into the semantic space and producing a vector of fixed dimension that represents the input's semantics. Next, the decoder uses the encoded semantic vector to decode it and produce the desired output. The language model is often the decoder if the output is text. It is clear that this process has benefits and drawbacks. Specifically, the benefits

include its high degree of flexibility, which allows it to be applied to tasks where the input is an image and the output is text, as well as tasks where the neural network utilised by the encoder and decoder is not restricted. This is an end-to-end procedure that does not involve discrete processing but rather combines language generation and semantic understanding. An inherent limitation of the encoder is that it produces a vector with a fixed dimension and information loss when the input varies. It is obvious that the semantic vector used to generate each word in text generation is too simple.

Figure 3 illustrates the differences between the fourth and the fifth in the two many-to-many models. The classical RNN structure has a restricted range of applications and requires equal-length input and output sequences. The input and output sequences not being equal can be the fourth type. The Sequence to Sequence model that implements the conversion from one sequence to another is called the seq2seq model. For instance, Google implemented the translation function by combining the attention model and the seq2seq model. The chatbot conversation model can also be used. The seq2seq model exceeds the fixed size of the input and output sequences in the classic RNN model.

Approach

Introducing an attention mechanism is one potential solution to the aforementioned issue. According to the so-called attention mechanism, when creating a word, different words have varying weights of attention. As can be seen, the attention-based NMT is based on tradition. It associates the word that is currently predicted to be translated with the expression that is learned by each word in the source language (the traditional expression only after the last word). Their attention to design is what makes the connection. The alignment matrix of the source and target languages can be obtained once the model has been trained based on the attention matrix. Every word in the target language is connected to every word in the source language using a perceptron formula. It is then normalised by the soft function to produce a probability distribution, which is the attention matrix.

Figure 4 illustrates this concept by placing the input sequence on the lef and the output sequence on top. The attention given to the words in the input sequence during word generation is determined by the probability distribution that each word in the output sequence corresponds to. The probability distribution of the model when generating the word joint is represented by the column that shows up at the blue box in the figure. The input joint, or the input word that the model focuses on the most when creating the word joint, is represented by the darkest portion.

The process of creating a word in order for it to receive this probability distribution is known as "attention." This allows the generated word, "alignment," to be

more effectively condensed into a single word. According to the results, the effect has significantly improved when compared to the traditional NMT (RNN-Enc). Its ability to visually align and process long sentences more efficiently is its most notable feature.

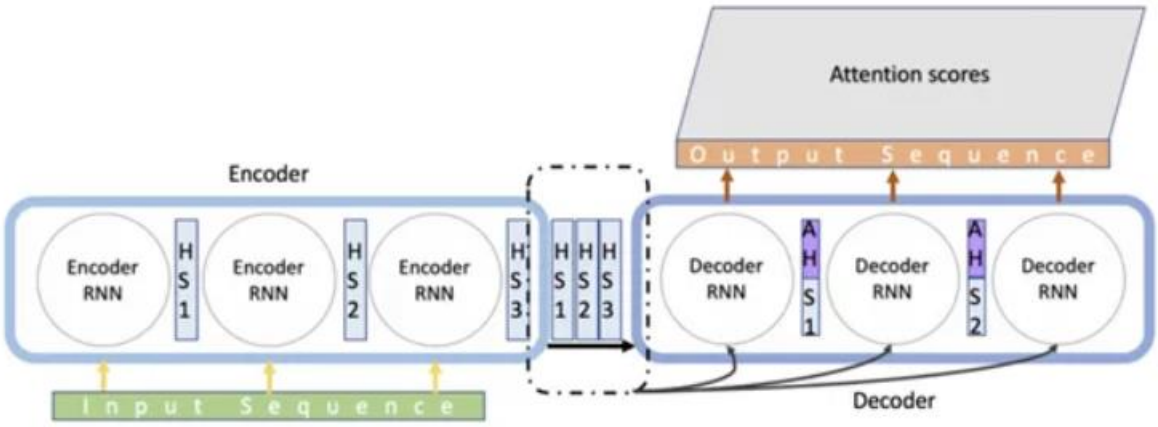

Figure 4: The attention mechanism based Enc-Dec Network

(source: https://medium.com/data-science-community-srm/understanding-encoders-decoders-with attention-based-mechanism-c1eb7164c581)

Experimental Setup

A. Dataset:

The CNN / DailyMail Dataset is an English-language collection of slightly more than 300,000 distinct reports authored by CNN and Daily Mail journalists. Although the initial version was designed for machine reading, comprehension, and abstractive question answering, the current version supports both extractive and abstractive summarization.

"**summarization**": A model for abstractive and extractive summarization may be trained using versions 2.0.0 and 3.0.0 of the CNN / DailyMail Dataset (version 1.0.0 was intended for machine reading and comprehension and abstractive question answering). The degree to which the output summary of a particular article's ROUGE score surpasses the highlight produced by the original article author serves as a gauge of the model's success. Zhong et al. (2020) found that a model trained for extractive summarization had a ROUGE-1 score of 44.41. For other models, view the Papers With Code leaderboard. For each instance, there is a string for the article, a string for the highlights, and a string for the id.

B. Implementation Details:

We used the TensorFlow library (tensorflow.org) to implement the proposed model, which employs data flow graphs. TensorFlow enabled us to make the most of the

hardware we had available. It is a versatile and portable library that supports Auto Differentiation. The cross-entropy function was used to calculate the error/loss from the correct prediction. We experimented with different learning rate and hidden layer size values to find the best balance of performance and computation time. This allowed us to achieve saturating accuracy after about 5 epochs. The batch size is 8 with a token length of 400 and 120 for the context and summary respectively in the LSTM model and pretrained T5 model which serves as a performance comparison in this experiment. The training model of the LSTM uses 'sparse_categorical_crossentropy' loss calculator and ‘RMS Prop’ optimizer which suited better in this experiment on a trial and error test basis.

C. Performance Evaluation

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is a widely used evaluation metric for text summarization tasks (Lin, 2004). ROUGE compares machine generated summaries to human written reference summaries using n-gram overlap statistics. This allows automated evaluation of how well the models capture the salient information. ROUGE has been shown to correlate highly with human judgement for assessing summarization quality in research studies. The original ROUGE paper by Lin in 2004 showed high correlation between ROUGE scores and human judgments across multiple datasets. Further empirical validation by Graham in 2015 established that ROUGE has high agreement with human ratings of summary quality.

ROUGE is simple and easy to implement for text summarization evaluation, given its n-gram matching approach. As a recall-based metric it rewards models that can cover as much of the salient information as possible. Several research papers use ROUGE for benchmarking text summarization systems - the Pegasus model paper by Zhang et. al. in 2020 uses ROUGE extensively to assess model performance. Narayan et. al. in 2018 also use ROUGE along with human evaluation for assessing quality and faithfulness of generated summaries. Paulus et. al. in 2017 employ ROUGE to measure performance on the CNN/Daily Mail summarization dataset specifically.

In summary, ROUGE is an effective proxy metric correlate for human assessment of summary quality. Its simplicity, recall focus and extensive use in literature make it a popular choice for evaluating text summarization solutions (Lin, 2004; Graham, 2015; Zhang et. al., 2020; Narayan et. al., 2018; Paulus et. al., 2017).

Results

The Sequence-to-Sequence model with LSTM encoder-decoder architecture is trained for 5 epochs on the dataset, achieving training and validation loss of 1.480 and 1.540 respectively. The model generates decent summaries but only obtains a Rouge-1 F1 score of 0.33 on the test set, indicating scope for improvement.

The T5 fine-tunes a pretrained T5 model, a Transformer model known to perform well for text summarization tasks. However, this is only trained for 20 epochs on a subset of data. Despite the limited training, the model produces relevant summaries and achieves a comparable Rouge score of 0.336. The implementation also demonstrates ease-of-use via the SimpleT5 wrapper library.

In summary, both the LSTM Seq-to-Seq and T5 models show promise but are undertrained. The T5 model leverages transfer learning so can likely achieve better performance with sufficient training. The LSTM Seq-2-Seq may also benefit from more epochs and regularization. In their current form, both display similar results quantified by overlap-based Rouge scores between system and reference summaries. But the T5 model seems better poised to improve as evidenced by adoption in state-of-the-art summarization techniques. More rigorous, long-term training would be required for a definitive comparison between the neural network architectures.

```
model_summaries[0] #predicted
```

```
["Rahul Kumar, 17, clambered over enclosure fence at zoo in Ahmedabad. He had been sitting near the animals when he made a rush for them.
 Intoxicated teenager ran towards the lions shouting 'Today I kill a lion'"]
```

```
text_summaries[0] #reference
```

```
[Drunk teenage boy climbed into lion enclosure at zoo in west India .Rahul Kumar, 17, ran towards animals shouting 'Today I kill a lion!'
Fortunately he fell into a moat before reaching lions and was rescued .]
```

Figure 5: Predicted and True summaries of T5 model

```
model_summaries[0] #predicted
```

```
Rahul Kumar, a 17-year-old, climbed over the enclosure fence at Ahmedabad Zoo. He'd been sitting close to the animals and suddenly he ran towards them.
A drunken teenager yelled, "Today I kill a lion," as she ran in the direction of the lions.
```

```
text_summaries[0] #reference
```

```
[Drunk teenage boy climbed into lion enclosure at zoo in west India .Rahul Kumar, 17, ran towards animals shouting 'Today I kill a lion!'
Fortunately he fell into a moat before reaching lions and was rescued .]
```

Figure 6: Predicted and True summaries of LSTM Seq-to-seq model

References

1. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin. Attention Is All You Need. https://doi.org/10.48550/arXiv.1706.03762.
2. Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. https://doi.org/10.48550/arXiv.1910.10683
3. Aakash Sinha, Abhishek Yadav, Akshay Gahlot. Extractive Text Summarization using Neural Networks. https://doi.org/10.48550/arXiv.1802.10137
4. Daniel O. Cajueiro, Arthur G. Nery, Igor Tavares, Maísa K. De Melo, Silvia A. dos Reis, Li Weigang, Victor R. R. Celestino. A comprehensive review of automatic text summarization techniques: method, data, evaluation and coding. https://doi.org/10.48550/arXiv.2301.03403
5. R. Haruna, A. Obiniyi, M. Abdulkarim and A. A. Afolorunsho, "Automatic Summarization of Scientific Documents Using Transformer Architectures: A Review," 2022 5th Information Technology for Education and Development (ITED), Abuja, Nigeria, 2022, pp. 1-6, doi: 10.1109/ITED56637.2022.10051602.
6. Rathi, Kartik and Raj, Saumy and Mohan, Sudhir and Singh, Yash Vardan, A Review of State-Of-The-Art Automatic Text Summarisation (April 04, 2022). International Journal of Creative Research Thoughts (2022), Available at SSRN: https://ssrn.com/abstract=4107774
7. Adhika Pramita Widyassari a b, Supriadi Rustad a, Guruh Fajar Shidik a, Edi Noersasongko a, Abdul Syukur a, Affandy Affandy a, De Rosal Ignatius Moses Setiadi. Review of automatic text summarization techniques & methods. https://www.sciencedirect.com/science/article/pii/S1319157820303712