Text Summarization from basic to advanced approaches

Nattapat Juthaprachakul
Simon Fraser University
301350117, njuthapr@sfu.ca

Siyu Wu
Simon Fraser University
301395909, swa246@sfu.ca

Abstract

In this paper, we explored several neural network-based approaches to tackle the challenge of abstractive text summarization on news articles. The methods include a vanilla LSTM sequence-to-sequence model without attention [1], BART transformer models with two different sizes [2], and T5 transformer models with three different sizes [3]. As these models have different architectures and complexity, we would like to evaluate and compare their performance using ROUGE metrics [4]. Through our experiments, we conclude that a LSTM sequence-to-sequence model lacks the ability to generalize with unseen data compared with fine-tuned and pre-trained transformers.

1 Introduction and Progress

Text summarization is the task of producing a concise and fluent summary of texts or documents while preserving the key information and overall meaning. The goal of this project is to implement different text summarization techniques on the same dataset and compare their performance based on the same evaluation metrics such as ROUGE [4]. There are three main reasons why we choose this project.

First, thanks to huge amount of available data, text summarization is becoming an important and useful task in several applications such as business analysis and data mining. For example, we always hear someone say, “I don’t want a full report, just give me a brief summary of the results.” With text summarization, we can gain key information from articles or documents without reading through the whole articles. This helps us save much time and effort.

Second, text summarization is a relatively new and very interesting topic in this course. We have gone through the list of final projects from the last term and term before last term and found out that this topic has never been implemented before in this class.

Lastly, we now have an opportunity to implement several techniques we have learnt from this class such as sequence-to-sequence models and transformer models with real-world dataset. Also, we have tried different techniques beyond scope of this course such as a text-to-text transfer network. We compare how these more advanced methods could improve the performance over the general approaches.

2 Related Work

Text summarization is the process of creating a summary of a certain document that contains the most important information of the original one [10]. There are two main strategies for text summarization namely summarization by extraction, which consists of concatenating source sentences into a summary, and summarization by abstraction, which involves generating novel sentences for the summary [11].

For the extractive summarization, the earliest paper is in the 1950s [12]. The paper proposed a simple approach to count the frequency of words in documents. Words that occur often are likely to be the main topic of the document. However, this approach does not account for words in different contexts.

For neural network-based techniques, they are generally either sentence-extractive (choosing a lot of sentences as a summary) or abstractive (creating a summary from a sequence-to-sequence model).

In this paper, we focus on the abstractive summarization. Though neural network-based models work very well in several NLP tasks such as Neural Machine Translation and text classification, abstractive text summarization remains a major challenge, especially for a sequence-to-sequence model.

Nonetheless, recent advances in neural network-based models with different architectures allow us to work with abstractive summarization more efficiently. For example, Pointer-generator networks [13] are introduced to deal with rare or out-of-vocabulary words, as well as repetitive words. An encoder-decoder model using LSTMs with the augmentation of hierarchical encoders and hierarchical attention can learn word and sentence level attention [14]. A paper proposed by [15] uses a transformer-based model to do abstractive summarization for a very long sentence.

3 Approach

In the summarization task, given a news article, we would like to generate a shorter version of the story while preserving important points of the articles. More formally, for an input sequence $x = \{x_1, x_x, \dots., x_N\}$ of N words, we would like to generate $y = \{y_1, y_x, \dots., y_M\}$ such that $M < N$ while y preserves the essence of x. The words in y comes from the same vocabulary used in x and may not occur in the original text.

We have tried an extractive approach such as frequency-based techniques like TextRank, but we find that the results are not suitable for comparison with the neural network-based approaches. In addition, CNN/Daily Mail dataset [5] is commonly used for the abstractive summarization.

3.1 Model 1: LSTM sequence-to-sequence model

Our baseline model is a vanilla LSTM model without attention [1] which is a general Deep learning-based architecture used in many NLP sequence-to-sequence tasks. Generally, we use LSTM with an encoder-decoder architecture inspired by Neural Machine Translation and summaries are generated from the decoder using target vocabulary. We train them from scratch and this model provides us a baseline of model training time and accuracy; thus, it helps us better understand how much the improvement of other Deep learning-based approaches could offer regarding both accuracy and efficiency.

3.2 Model 2: Finetuned transformer

For our second model, we finetune the BART-base model on our pre-processed dataset. BART-base is a pre-trained transformer model by Facebook which is trained on XSum news dataset [6] with 139 million parameters [2]. BART uses a standard sequence-to-sequence structure (Neural Machine Translation architecture) with a bidirectional encoder (similar to BERT [7]) and a left-to-right decoder (similar to GPT [8]). It is a denoising autoencoder built with a sequence-to-sequence model that is applicable to a very wide range of end tasks.

BART pretraining has two stages as shown in Figure 1. First, text is corrupted with an arbitrary noising function. The second stage is that a sequence-to-sequence model is learned to reconstruct the original text. BART uses a standard Transformer-based NMT architecture, which can be seen as generalizing BERT because of the bidirectional encoder [7].

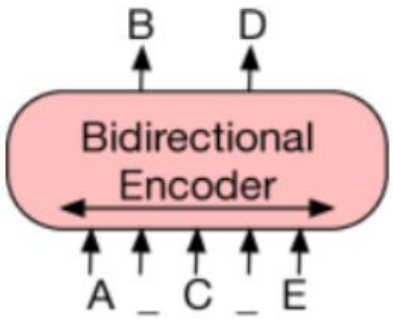

Figure 1: BERT with random tokens are replaced with masks, and the document is encoded bidirectionally.

A key advantage of this setup is the noising flexibility as arbitrary transformations can be applied to the original text such as arbitrary length adjustment. In addition, BART is particularly effective for text generation and comprehension tasks. Importantly, it is useful for text summarization tasks because it has an autoregressive decoder as it can be directly finetuned (Figure 2).

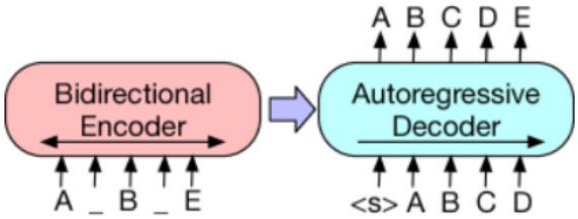

Figure 2: for BART, inputs to the encoder need not be aligned with decoder outputs.

Although BART architecture is closely related to BERT, there are two major differences. First, each layer of the decoder additionally performs cross-attention over the final hidden layer of the encoder. Second, BART does not use an additional feed-forward network before word prediction while BERT does. Therefore, BART contains about 10% more parameters than the equivalently sized BERT model [6].

There are several transformations used in BART as shown in Figure 3.

1. Token Masking: random tokens are sampled and replaced with [MASK] elements.
2. Sentence Permutation: a document is divided into sentences based on full stops and these sentences are shuffled in a random order.
3. Document Rotation: a token is chosen uniformly at random and the document is rotated so that it begins with that token. This task trains the model to identify the start of the document.
4. Token Deletion: random tokens are deleted from the input. Compared to token masking, the model must decide which positions are missing inputs.
5. Text Infilling: several text spans are sampled with span lengths drawn from a Poisson distribution. Each span is replaced with a single [MASK] token and zero-length spans correspond to the insertion of [MASK] tokens. Text infilling teaches the model to predict how many tokens are missing from a span.

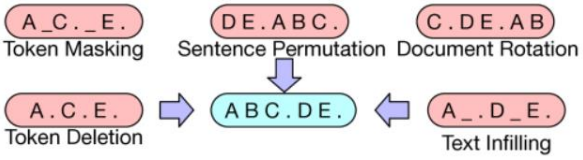

Figure 3: BART input transformations.

3.3 Model 3-6: Pre-trained transformers without finetuning

Text-to-text Transformer Model (T5) was invented by Google and pre-trained on C4 dataset, which is a cleaned version of Common Crawl's web crawl corpus. It is an encoder-decoder model pre-trained on a multi-task mixture of unsupervised and supervised tasks [3]. The goal of this model is to reframe all NLP tasks into a unified text-to-text-format where the input and output are always text strings.

T5 is different from BERT-based models that can only output either a class label or a span of the input. The advantage of this model is that it allows us to use the same model, loss function, and hyperparameters on any NLP task, including machine translation, summarization, question answering, and classification tasks.

T5 trains with the same objective as BERT which is the Masked Language Model (MLM) with a little modification. As the MLM is used in Bidirectional models like BERT, at any time *t* the representation of the word is derived from both the left and the right context of it. The difference between BERT and T5 is that T5 replaces multiple consecutive tokens with a single mask keyword while BERT uses mask tokens for each word.

In Figure 4, the original text is transformed into input and output pairs by adding different types of perturbations to it. Since the final objective is to train a model that inputs and outputs text, the targets are designed to produce a sequence. This is different from BERT that tries to output one word, which is itself, through final feed-forward and softmax at the output level.

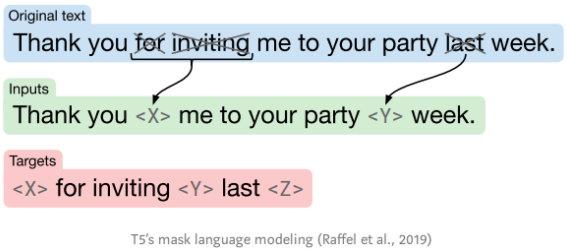

Figure 4: T5 Objective

BART-large is similar to BART-base but bigger in terms of dimensions and layers; thus, more numbers of parameters.

For these four pre-trained models, we obtain them from Huggingface [9] and directly use them without any finetuning. These models have different numbers of parameters and underlying pre-trained data sets as shown in Table 1.

Model	Number of Parameter	Pretrained data	Finetune data
LSTM	28M	-	CNN/Dailymail
BART-base	139M	XSum	CNN/Dailymail
BART-large	406M	XSum	-
T5-small	60M	C4	-
T5-base	220M	C4	-
T5-large	770M	C4	-

Table 1: Model settings

4 Evaluation

4.1 Dataset

We use raw data without data anonymization of the CNN/Daily Mail dataset [5], which consists of online news articles (or stories) paired with multiple summaries (or highlights). The first feature is a text of news articles which is used as the documents to be summarized while the second feature is the joined text of highlights which is the target text summarization as shown in Table 2.

The articles have an average of 781 tokens while the summaries have an average of 56 tokens. For pre-processing, we remove punctuation, number, and CNN/Daily Mail name tags at the beginning of every line. We also lower every letter in all words and remove noisy words that are not related to news articles such as advertisements.

Story	The label on the package claimed that it contained T-shirts and baby toys. When customs officials in Sydney scanned the parcel, they found five pythons and two venomous tarantulas. But when customs officials in Sydney X-ray scanned the parcel, they found instead five pythons and two venomous tarantulas. On Tuesday, authorities raided the house in Sydney to which the parcel had been addressed. Officials seized evidence but expect to file charges later, the customs agency said. Importing live animals without a permit is illegal in Australia and can yield a 10-year prison sentence and a fine of 110,000 Australian dollars ($92,000 U.S.). (cont.)
Summary	-Customs officials in Australia find pythons and tarantulas in package -The parcel had been sent from the United States -The creatures were later killed because they posed a quarantine risk

Table 2: Example of original dataset

After all pre-processing steps, we have total data of 305,758. We split them into train/validation/test sets with a fraction of 90/5/5. To save training and decoding time, we use only the first two sentences of stories and two summaries. The pre-processed summaries and stories now have around 12 and 53 words on average respectively.

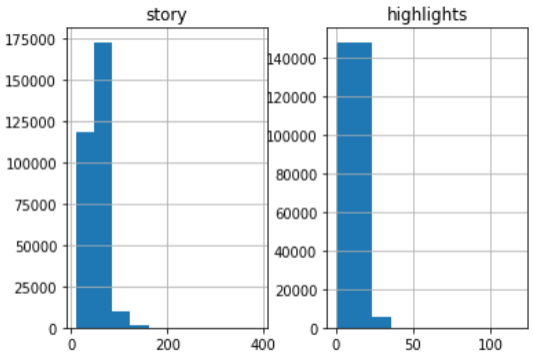

Figure 5: Preprocessed Dataset

4.2 Evaluation Metric

The widely used general evaluation metric for text summarization is Recall-Oriented Understudy for Gisting Evaluation (ROUGE) which automatically determines the quality of a summary by comparing the output texts produced by algorithms and the reference summaries [4]. In this paper, we use F1 from the ROUGE metric and select only three specific ROUGE metrics namely: ROUGE-1,

ROUGE-2, and ROUGE-L. The F1 score can be calculated as follows:

$$\text{Precision} = \frac{\text{number of overlapping words}}{\text{total words in model summary}}$$

$$\text{Recall} = \frac{\text{number of overlapping words}}{\text{total words in reference summary}}$$

$$\text{F1} = \frac{2 * (\text{Recall} * \text{Precision})}{\text{Recall} + \text{Precision}}$$

ROUGE-n: it is a recall-based metric that is based on comparison of n-grams between reference summaries and candidate summaries.

$$\text{ROUGE} - \text{n} = \frac{\sum_{S \in \text{ReferenceSummaries}} \sum_{\text{gram}_n \in S} \text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{S \in \text{ReferenceSummaries}} \sum_{\text{gram}_n \in S} \text{Count}(\text{gram}_n)}$$

ROUGE-L: it uses the concept of the longest common subsequence (LCS) between the two sequences of texts. The intuition is that the longer the LCS between two summary sequences, the more similar they are. Given reference summary X of length *m* and model summary Y of length *n*, the F1 score can be computed:

$$\text{Recall}_{\text{LCS}} = \text{R}_{\text{LCS}} = \frac{\text{LCS}(\text{X}, \text{Y})}{\text{m}}$$

$$\text{Precision}_{\text{LCS}} = \text{P}_{\text{LCS}} = \frac{\text{LCS}(\text{X}, \text{Y})}{\text{n}}$$

$$\text{F1}_{\text{LCS}} = \frac{1 + \beta^2 \text{R}_{\text{LCS}} \text{P}_{\text{LCS}}}{\text{R}_{\text{LCS}} + \beta^2 \text{P}_{\text{LCS}}}$$

4.3 Experiment Detail

We have implemented one sequence-to-sequence model, finetuned one transformer-based model (BART-based [2]), and used 4 different pre-trained transformers (BART-large [2], T5-small, T5-base, and T5-large [3]) from Huggingface [9] directly. Later, we use all these models to predict the summaries and evaluate them with ROUGE scores as shown in the Table 1 and 3.

We use Google Colab as our main training facilities. We have tried CPU, GPU (Tesla T4), and TPU. For training of our baseline model, it takes around 7, 2, and 0.5 hours respectively while it takes around 1 hour for pre-tuning the BART-base model [2] with our dataset. For testing, we use GPU for decoding which takes around 1-1.5 seconds per sample. Since we have 10,000 samples in the test set, it takes around 2-3 hours for this process per model.

4.4 Experiment Result

From table 3, we can see that the vanilla LSTM sequence-to-sequence model that was trained from scratch has a relatively disappointing performance in all ROUGE metrics compared with other models. The performance of Transformer-based models like the BART-large model and all three T5 models almost doubles that of the LSTM model. Nonetheless, they have very similar performance to each other in all metrics regardless of size of models and underlying datasets.

On the other hand, our finetuned Transformer-based BART-base model outperforms LSTM but still cannot compete with other Transformer models. This may be because of how we perform finetuning or data processing. Overall, the accuracy of the best model in our experiment is not high compared with the current state-of-the-art models. Nonetheless, in other experiments from different researchers using this same dataset, their best models hardly have an accuracy larger than 0.3 in ROUGE-2 and 0.4 in ROUGE-1 and ROUGE-L.

Model	ROUGE-1	ROUGE-2	ROUGE-L
LSTM	0.12	0.02	0.11
BART-base	0.22	0.08	0.20
BART-large	0.27	0.13	0.25
T-small	0.27	0.13	0.26
T5-base	0.28	0.13	0.27
T5-large	0.28	0.13	0.26

Table 3: Experiment results

5 Discussion

Table 4 shows the examples of our model outputs compared with a reference summary. We can see that the summary output from LSTM sequence-to-sequence model is incorrect as seen in the word "*Manchester United*" repeating itself in both subject and object. In contrast, the outputs of other

models are correct and understandable in terms of meaning and grammar.
There are two key points in the reference summary. The first point is "*Manchester United have made Schweinsteiger as their top target*." BART base model and all T5 models identify this point successfully. The other point is "*Louis Van Gall eyes Scheweinsteiger reunion at Old Trafford*." Only the BART-large model can capture this detail.

Therefore, for Example 1 in Table 4, almost all the models except the LSTM model can correctly summarize one key point from the text. The sentences they generate are readable and concise. Although none of them capture both of two key points, we think that their performances are acceptable and helpful for automatic summarization tasks.

Reference summary	-manchester united have made bastian schweinsteiger their top summer target. -louis van gaal eyes bastian schweinsteiger reunion at old trafford.
LSTM seq2seq	-manchester united have been linked with a move to manchester united.
BART-base	-manchester united have made bastian schweinsteiger their top summer target.
BART-large	-bastian schweinsteiger is thought to be keen on joining his former boss louis van gaal at old trafford.
T5-small	-manchester united have made bastian schweinsteiger their top summer target. -the schweinsteiger is thought to be keen.
T5-base	-manchester united have made bastian schweinsteiger their top summer target. -the schweinsteiger is thought to be keen.
T5-large	-manchester united have made bastian schweinsteiger their top summer target. -schweinsteiger is thought to be keen on.

Table 4: Example 1 of Model outputs

Table 5 shows Example 2 of all model outputs compared with a reference summary. We can see that the summary output from the LSTM model is incorrect. In the predicted summary, the model is confused with the word "Liverpool", which is a football club, with a football player and the word "Premier League", which is a football league system, with a football club. The correct summary should be about "Liverpool", which already is in "Premier League", wanting to add a new player to their team.

On the other hand, BART-large can correctly output the first summary while it adds some additional information that is incorrect to the second summary, which is "but key could be the departure." In contrast, other models besides these two models can output the summaries that are both understandable and grammatically correct.

For the reference summary, there is some extra information that our models may have troubles dealing with. The first point is the word "reds" which refers to "Liverpool" not color. The other point is "Anfield club" which also refers to "Liverpool" even though "Anfield" is a small city in Liverpool city. In conclusion, same as Example 1, all the models except the LSTM model can correctly predict summary.

Reference summary	-the reds want swansea striker wilfried bony to boost their attack. -petr cech and jack butland are being considered by the anfield club.
LSTM seq2seq	-liverpool have been linked with a move to the premier league.
BART-base	-liverpool are continuing to discuss the possibility of new additions to their squad should brendan rodgers get money to.
BART-large	-liverpool are continuing to discuss the possibility of new additions to their squad. -a goalkeeper and striker remain the objectives but key could be the departure.
T5-small	-liverpool are continuing to discuss the possibility of new additions to their squad should brendan rodgers get money to spend.

T5-base	-liverpool are continuing to discuss the possibility of new signings. -a goalkeeper and striker remain the objectives for brendan rodgers.
T5-large	-liverpool are continuing to discuss the possibility of new additions to their squad should brendan rodgers get money to spend.

Table 5: Example 2 of Model outputs

6 Conclusion and Future Work

In this paper, we built several neural network-based text summarization models that can extract the main highlights from news articles. From our experiment, we found that transformer-based models have better performance than a vanilla LSTM sequence-to-sequence model. The accuracy of the LSTM model that is built from scratch is relatively low and its predicted summaries are repeating and inaccurate in term of meaning and grammar; thus, cannot be used as the summary of the given news article.

On the other hand, all transformer-based models can produce grammatically correct and meaningful summaries in most cases based on our experiments. Two model-generated example outputs shown in Table 4 and 5 are readable as their meanings are clear and there are no grammatical mistakes. Additionally, as there are more than one sentences in the reference summary which mentions multiple key information from the article, those summary sentences generated by the transformer models could at least capture one key points of the news article Therefore, to some extent, these models are helpful and useful for automatic text summarization. However, human involvement is still needed as to make sure the summaries generated by the models are correct and sufficient in some cases. Therefore, our experiment proves the importance of the transformers in the text summarization model.

Text summarization is still a challenging topic. The overall ROUGE scores of our models are below 0.30 which is still below the state-of-the-art models, which means there are rooms for major improvement. Nonetheless, we also noticed that in some other experiments, after implementing different models and using different datasets, researchers could hardly get an accuracy more than 0.40, which means that there is still a long way to go before we find out better text summarization methods..

For the future work, we could incorporate named-entity recognition to improve the performance of our models. In one of our output examples, we find that the models still have troubles in dealing with proper noun, such as “reds” which could indicate a kind of color or soccer clubs and in different scenarios, “reds” could mean different clubs, and “Anfield” which could indicate a county in Liverpool, a stadium or a soccer club. By implementing named-entity recognition, we could solve this problem and improve the performance of our models. In addition, we could incorporate techniques to deal with rare and out-of-vocabulary words that is a major problem for text summarization.

References

[1] Sutskever, I., Vinyals, O., and Le, Q.V. (2014). Sequence to Sequence Learning with Neural networks. In Advances in Neural Information Processing (NIPS), Montreal, Canada.

[2] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, & Luke Zettlemoyer. (2019). BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension.

[3] Colin Raffel and Noam Shazeer and Adam Roberts and Katherine Lee and Sharan Narang and Michael Matena and Yanqi Zhou and Wei Li and Peter J. Liu (2019). Exploring the Limits of Transfer Learning with a Unified Text-to-Text TransformerCoRR, abs/1910.10683.

[4] Lin, C.Y. (2004). ROUGE: A Package for Automatic Evaluation of Summaries. In Text Summarization Branches Out (pp. 74–81). Association for Computational Linguistics.

[5] R. N. et al., “Cnn/dailymail dataset,” CoNLL, 2016

[6] Shashi Narayan, Shay B. Cohen, & Mirella Lapata (2018). Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme SummarizationArXiv, abs/1808.08745.

[7] Jacob Devlin and Ming-Wei Chang and Kenton Lee and Kristina Toutanova (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language UnderstandingCoRR, abs/1810.04805.

[8] Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving language understanding by generative pre-training. OpenAI. 2018.

[9] Thomas Wolf and Lysandre Debut and Victor Sanh and Julien Chaumond and Clement Delangue and Anthony Moi and Pierric Cistac and Tim Rault and Rémi Louf and Morgan Funtowicz and Jamie Brew (2019). HuggingFace's Transformers: State-of-the-art Natural Language ProcessingCoRR, abs/1910.03771.

[10] X. Carreras, L. Màrquez, Introduction to the CoNLL-2004 shared task: Semantic role label-ing, in Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004(2004), pp. 89–97

[11] U.Hahn and I.Mani 2000. The challenges of automatic summarization. IEEE Computer, 33(11): 29-36

[12] Luhn, H. P. (1958) The automatic creation of literature abstracts, IBM Journal of Research and Development, vol. 2, no. 2.

[13] C. M. A. See, P. Liu, “Get to the point: Summarization with pointer-generator networks,” CoRR, 2017.

[14] Nallapati, R., Zhou, B., Santos, C. D., Gulcehre, C., and Xiang, B. (2016). Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond. In CoRR, arXiv:1602.06023.

[15] P. L. et al., “Generating wikipedia by generating long sequences,” ICRL, 2018.