Extractive Text Summarization for E-commerce product reviews using Transformers

Sumedh Khodke
SEAS, University at Buffalo
sumedhk@buffalo.edu
UB Person# 50419157

Hamza Abubakar Kheruwala
SEAS, University at Buffalo
hamzaabu@buffalo.edu
UB Person# 50418479

Snigdha Srivastava
SEAS, University at Buffalo
srivast3@buffalo.edu
UB Person# 50365547

Kshitij Subhash Chhajed
SEAS, University at Buffalo
kchhajed@buffalo.edu
UB Person# 50432890

Abstract

NLP has seen increasing applications over the last few years, owing to the rise of digital media and solutions through technology. In particular, text summarization has been widely used, especially for E-commerce applications, movie reviews, scientific papers, and much more. In this paper, we have implemented text summarization using some of the latest state-of-the-art techniques viz. BART, Pegasus, and T5 summarization models. These models have been pre-trained based on generalized datasets. In this paper, we have improved on the generalization to adapt to out of domain text inputs through fine-tuning.

Keywords: Natural Language Processing (NLP), Automatic Text Summarization (ATS), BERT, BART, T5, Transformers, General Language Understanding Evaluation (GLUE)
Code Release: Github - github.com/sumedhkhodke/CSE_676_project

1 Introduction

The advent of technology and digital media has led to an exponential increase in data in different forms over the last few years. With the constant exposure to large sets of data, it has become imperative to evaluate and extract valuable information with the latest technology and methodologies.
Text Summarization is one of the most important tools when it comes to different NLP-based techniques. Applications such as movie reviews, E-commerce reviews, news articles, blogs, etc use different summarization to extract integral information from otherwise large sets of data. For the same, ATS aims to generate concise and fluent summaries with the help of different summarization algorithms. With the help of extension, abbreviation, splicing and cutting, etc, this finds keypoints for the generation of smooth and dynamic short excerpts of data.
For evaluation of summarization, we have human and automated evaluation. In human evolution, experts go through the generated summary and talk about points like importance, coverage, grammatical score, and non-redundancy of the points covered. In case of automatic evaluation, ROUGE is used for the comparison of the generated summary with human-generated summaries.

ATS (given in Figure 1) can be majorly divided into 3 different categories: *i. extractive summarization* which generates summaries from the source text through combination and extraction of important keywords *ii.abstractive summarization* which uses neural networks for concise summary generation through expansion, imitation, substitution, etc *iii.hybrid summarization* that uses both extractive and abstractive types of text summarization.

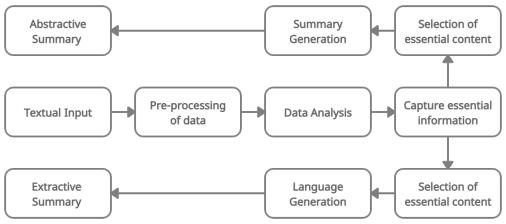

Figure 1: Automatic Text Summarization (ATS) techniques

The work proposed in this paper builds on extractive text summarization with the help of transformers. The primary models implemented here include BART, PEGASUS, and T5 models for summarization. With the adaptability of these models, we will fine-tune the parameters to suit our needs. The dataset used here includes product reviews from Amazon, which has been discussed further in coming section.

2 Literature Review

With increasing emphasis on NLP and text summarization, we have witnessed advanced models being developed since the last few years. In comparison to abstractive summarization, extractive summarization is shown to prove more logical with efficient results, and therefore has been given the greater focus for the same [15]. Some of the most recent work done for text summarization have been proposed in Table 1.
For the same, we have seen some popular weighing methods to highlight important parts in the given input. TF-IDF [18], term frequency–inverse document frequency is based on Bad of Words and is used for text vectorization. This model assigns a lower probability to the words that appear the most frequently, thereby removing stop words that are of no relevance to the generated summary. Similarly, a word appearing uniquely in the document is conversely given a high weightage. To support user modelling, text mining and information retrieval, the same is used as a numerical statistic as the weighing factor.
The Text Rank algorithm (proposed by Mihalcea *et al.* [14]), is based on the Page Rank algorithm from Google, and summarizes by assigning importance to vertices in the graph. Applying the same idea as that of graph-based algorithms, the importance assigned to a vertex is based on recursive global information, in place of local information. This way, it captures the relation between various entities in the text, hence using recommendation for the generation of summaries.

3 Methodology and Concepts

In the last few years, we have witnessed some state-of-the-art techniques when it comes to various NLP tasks. These models have proved to be complex yet robust, generalized to multiple NLP use-cases, with results similar to that of humans. The following sections will talk about some of these models we have fine-tuned and implemented on the given dataset, the comparison of which is given in Table 3.

3.1 T5 - Text-To-Text Transfer Transformer — Google [17]

The T5 model combines multiple NLP tasks into a single text-to-text format which consists of text strings, both as input and output. This framework can be applied to any NLP task, even with the same set of hyper parameters, loss function, and model. Figure 2 shows multiple applications that can be fulfilled with the same T5 framework.
With 11-billion parameters, this model uses relative scalar embedding and is a mix of encoders and decoders and is pre-trained on both supervised and unsupervised tasks, with each of them converted to a text-to-text format. The training here is done with the help of teacher enforcing, where we use an input sentence and its corresponding target sequence. The target sequence is then added to the EOS (End of Sequence) token.

Table 1: Related works - Comparison

SR No.	Author	Year	Objective	Methodology	Finding(s)/Future Scope(s)	Dataset(s) used	1	2
1.	Rawat *et al.* [19]	2021	Dynamic Summarization for a changing text corpus	Transformers & Graph Reduction	Improvement of Bleu score based on batch size threshold	Amazon Product Reviews	✗	✓
2.	Gu *et al.* [9]	2021	Policy text title summarization	Keyword-fusion Pointer-Generator Network	Better performing than Seq2Seq, former pointer-generator network	Self-generated electricity policy datasets	✗	✓
3.	Chen *et al.* [4]	2021	News Text Summarization	BART-Text Rank Model	Improvement over Text Rank and BART model	CNN & Daily Mail data set	✗	✓
4.	Gupta *et al.* [10]	2021	Document-level summarization & encoding	LSA & Sentence-based modelling with BERT	Fluent summaries with semantic aspect comparison	Kaggle Dataset	✓	✗
5.	Nguyen *et al.* [15]	2021	Summarization for Biomedical Question-Answering System	Extractive-Abstractive hybrid (BART)	Proposal extensible for deep question-analysis, sentence clustering	MEDIQA-AnS Dataset	✓	✓
6.	Jiang *et al.* [11]	2021	Hybrid Automatic Text Summarization	Enhanced Bi-LSTM & Attention	Future optimization of the evaluation indexes,	LCSTS & TTnews	✗	✓
7.	Ma *et al.* [13]	2021	Topic-Aware Text Summarization	T-BERTSum (T-BERTSum)	Future improvement on long multi-topic data	CNN/Daily mail & XSum	✓	✓
8.	Farahani *et al.* [8]	2021	Persian Abstractive Summarization	ParsBERT & Pre-trained mT5	Absence of work comparison due to novelty	Self-generated (pn-summary)	✗	✓
9.	Su *et al.* [20]	2021	Variable-Length Abstractive Summarization	Two-Stage Transformers	Inaccurate results owing to length of document	WIKI-727K dataset	✗	✓
10.	Batra *et al.* [3]	2021	News Summarization Application - SARS-CoV-2	Deep NLP Transformers	Incorporation of multiple NLP models in the future	COVID-19 Public Media Dataset	✓	✗

1. Extractive Summarization 2. Abstractive Summarization

Table 2: Comparison of Proposed Models

SR No.	Model	Author	Dataset Used	Pre-training Objective	Metrics	Results
1.	T5	Raffel *et al.* [17]	GLUE benchmark	Fill-in-the-blank-style de-noising objective	ROUGE	Highest parameter count yet lower computation cost and better performance
2.	BERT	Devlin *et al.* [6]	English Wikipedia & Brown Corpus	Masked Language Modeling & Next Sentence Prediction	Accuracy	7.6% improvement on GLUE & 93.2% F1 score
3.	BART	Lewis *et al.* [12]	English Wikipedia & Book Corpus	Token Classification	ROUGE	1.2 ROUGE-L improvement, 6 points over previous state-of-the-art
4.	PEGASUS	Zhang *et al.* [23]	C4 & HugeNews	Gap Sentences Generation	ROUGE	State-of-the-art improvement over considered 12 diverse downstream datasets

In sum, the T5 model contains 5 variants which have been trained on 3 billion and 11 billion parameters. The performance is quite similar to that form humans, and it is adaptive enough to be applied to multiple NLP tasks.

3.2 Bidirectional Encoder Representations from Transformers (BERT) — Google [6]

This model (pipeline given in Figure 3) was developed in response to the shortage of human-labelled and annotated data, when most NLP techniques were found to perform much better from comparatively larger amounts of data. To counter this problem, pre-training was done on the massive amount of annotated data on the internet, which was then fine-tuned to service specific NLP tasks, thus giving us a performance much better than that of models built from scratch.
This model was built on pre-training contextual representations, and is the first of its kind to be trained on a plain text corpus. BERT was the first implementation of bi-directionality, which was done by masking some words in the input. Following this, each word is conditioned bidirectionally for the prediction of the masked words.
When compared to other NLP models, BERT performed all tasks with almost no specific changes for the respective tasks, while still improving over the state-of-the-art on the GLUE Benchmark. The data here ranged from 2500-400000 examples, and this models gives improved results on all of them.

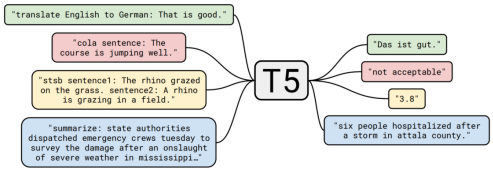

Figure 2: T5 Model Framework [1]

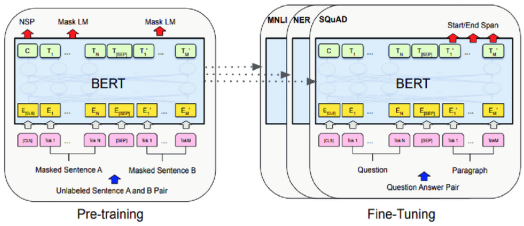

Figure 3: Fine-tuning and Pre-training - BERT

3.3 BART - Denoising Sequence-to-Sequence Pre-training for Natural Language Understanding — Facebook [12]

This model follows a machine learning architecture based on transformers, and is the generalization of GPT, BERT, and several different pre-training techniques. This model is trained through the corruption of text with the help of a noising function, and reconstruction of the original text.
As given in Figure 4, we replace different parts of the textual data with masked symbols. A bidirectional encoder is used to encode the corrupted document, and the auto-regressive decoder on the right helps calculate the likelihood of the original document. Here, fine-tuning is done by providing both the encoder and the decoder with the uncorrupted document as input. Finally, the final hidden state helps with the representations here. It is known to have achieved better downstream performance on generation tasks, like abstractive summarization and dialogue, with two changes:

1. Add a causal decoder to BERT's bidirectional encoder architecture. Each layer of the decoder additionally performs cross-attention over the final hidden layer of the encoder (as in the transformer sequence-to-sequence model)
2. Replace BERT's fill-in-the blank cloze task with a more complicated mix of pre-training tasks.

BART uses the standard sequence-to-sequence Transformer architecture from BERT, except the replacement of ReLU activation functions to GeLUs, and initialise parameters from N (0, 0.02). For the base model, there are 6 layers in the encoder and decoder, and for the large model there are 12 layers in each. In total, BART contains roughly 10 percent more parameters than the equivalently sized BERT model. During the fine-tuning phase for summarization tasks, the input sequence is the text sample we want to summarize, and the output sequence is a ground truth summary. Seq2Seq architectures can be directly fine-tuned on summarization tasks, without any new randomly initialized heads.

3.4 PEGASUS - Pre-training with Extracted Gap-sentences for Abstractive Summarization — Google [23]

PEGASUS in essentially an abstractive text summarizer made through pre-trained encoder-decoder models on massive datasets through concentration on self-supervision (Architecture given in Figure 5). PEGASUS uses an encoder-decoder model for sequence-to-sequence learning. In such a model, the encoder will first take into consideration the context of the whole input text and encode the input text into something called context vector, which is basically a numerical representation of the input text. This numerical representation will then be fed to the decoder whose job is decode the context

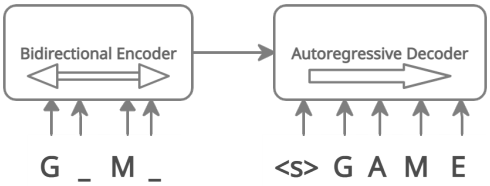

Figure 4: BART Model Framework [12]

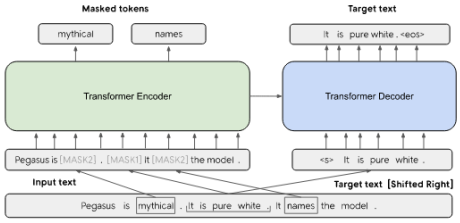

Figure 5: PEGASUS - Model Architecture [23]

vector to produce the summary. Here, the input document is scouted for important sentences, which are then either masked or removed. Following this, these are then generated as a single output sequence from the remaining sentences, thereby generating an extractive summary. This model is shown to show better results (similar to human performance), even when running on low-resources and lower amounts of training examples.

3.5 Dataset Used

Being a leading E-commerce platform, Amazon has accumulated a huge number of reviews for millions of products posted on the platform. These reviews have proved to be a rich area of focus for different NLP tasks over the years. For the proposed work, we have considered the Amazon Customer Reviews dataset, with 130+ million reviews available for research. Given as TSV files, each row points to respective reviews (special characters removed). Under each review, we have fields like consumer ID, review ID, star rating, upvotes, headline, body, and date of posting.

4 Implementation - Github and Demo

- We have implemented the fine-tuned versions of pre-trained models of BART, Pegasus and the T5 model for summarization on the Amazon Reviews dataset [16].
- The model were run on Google Colaboratory, Tesla P100 GPU - NVIDIA PASCAL, RAM - 16 GB
- For demonstration purposes, we have deployed working implementation of 2 of the models on the huggingface hub where the models can be inferenced directly:
  - t5-base-amazonreviews
  - distilbart-cnn-12-6-amazonreviews

4.1 Novel tricks and techniques used for fine-tuning

During the finetuning process, several tricks for easily fitting the model on low resource systems were used. Some of them are as follows:

1. **8 bit Optimizers:** We made use of 8 bit Adam optimizers in the finetuning process which accelerates optimization compared to default 32 bit Adam optimizer but uses memory that

might otherwise be allocated to model parameters, thereby limiting the maximum size of models trained in practice. The 8-bit statistics maintain the performance levels of using 32-bit optimizer states.[5] [7]

2. **Efficient Batching and learning rate experimentation:** We experimented with a combination of smaller batch sizes and higher learning rates for the model to perform optimally based on the resources for training. The chosen combination ensured that the model training remained stable even when the learning rates were higher.[22]
3. **Gradient accumulation:** We implemented gradient accumulation which modifies the last step of the training process. Instead of updating the network weights on every batch, we save gradient values, proceed to the next batch and add up the new gradients. The weight update is then done only after several batches have been processed by the model. Gradient accumulation helps to imitate a larger batch size.[21]

4.2 Data Pre-Processing

The collected data consisted of reviews from Amazon, those of which contain several unwanted and redundant data that can be removed for efficiency and better results. We performed the following steps as part of data cleaning:

- Adding 'summarize' keyword at the beginning of the ground truth labels for training set
- White space removal
- Tagging and Toeknization
- Collating and Batching

4.3 Fine-tuning on T5 - Text-To-Text Transfer Transformer — Google

T5 has different model sizes like t5-small, t5-base, t5-large, t5-3b, t5-11b. The model is fine-tuned using t5-base. For large datasets, we use batch sizes to deliver the number of training/validation instances in a single forward/backward pass. There can be additional memory space as the batch size increases. Learning rate is required to keep the variance in the gradient expectation constant and is set to 1e-4. We set the number of training epochs as 3 and validation epochs as 1. These metrics are set keeping in mind, the large size of the dataset, as the for total number of iterations, each pass runs according to the number of batch size, for example, if there are 5000 training examples, and the batch size is 500, then it will take 10 iterations to complete 1 epoch. T5 pre-training dataset - C4 Common Crawl data. Some of the hyper-parameters are as follows:

1. learning_rate: 1e-04
2. train_batch_size: 8
3. eval_batch_size: 8
4. seed: 42
5. val_epochs: 3
6. model:t5-base

4.4 Fine-tuning on BART - Denoising Sequence-to-Sequence Pre-training — Facebook

There are multiple variants of the BART model based on the model architecture size and parameter count. For the purpose of prototyping a summarization model based on this architecture, we have employed the smaller BART model of distilBART which is created using knowledge distillation and model compression techniques. The distilled version manages to retain 90% of the base model performance for 40% of its original size. Given the computational resource constraint, this model is easier to fit and fine-tune further on GPUs with lower end memory capacity. BART pre-training dataset - ELI 5, CNN-DailyMail. Some of the hyper-parameters are as follows:

1. learning_rate: 2e-05
2. train_batch_size: 4

3. eval_batch_size: 4
4. seed: 42
5. optimizer: Adam with betas=(0.9,0.999) and epsilon=1e-08
6. lr_scheduler_type: linear
7. num_epochs: 1
8. mixed_precision_training: Native AMP

4.5 Fine-tuning on PEGASUS - Pre-training with Extracted Gap-sentences for Abstractive Summarization

Pegasus model has different versions available like pegasus-pubmed, pegasus-bigPatent, pegasus-multinews, etc. We have used the distill-pegasus-xsum-16-4 model. This model is a mixed and stochastic kind of model which uniformly samples a gap sentence ratio between 15% and 45%. The Gap Sentences Generation which forms the base of this model is based on the 3 strategies: Random, Lead and Principal. The Pegasus base model has been trained on C4 (Colossal and Cleaned version of Common Crawl) and HugeNews dataset. Different hyper-parameters involved in the implementation are:

1. learning_rate: 0.01
2. train_batch_size: 1
3. eval_batch_size: 1
4. seed: 42
5. num_epochs: 1
6. optimizer: Adam with betas=(0.9,0.999) and epsilon=1e-08

5 Results and Discussion

5.1 Performance metric - ROUGE

ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. It is essentially a set of metrics for evaluating automatic summarization of texts as well as machine translations. The ROUGE scores are dependent on precision, recall and F-measure. The precision aspect becomes really crucial when we are trying to generate summaries that are concise in nature. Following are the types of ROUGE scores:

- **ROUGE-N:** measures unigram, bigram, trigram and higher order n-gram overlap
- **ROUGE-L:** measures longest matching sequence of words using longest common subsequence. An advantage of using longest common subsequence is that it does not require consecutive matches but in-sequence matches that reflect sentence level word order. Since it automatically includes longest in-sequence common n-grams, you don't need a predefined n-gram length.
- **ROUGE-S:** Is any pair of words in a sentence in order, allowing for arbitrary gaps. This can also be called skip-gram concurrence. For example, skip-bigram measures the overlap of word pairs that can have a maximum of two gaps in between words. As an example, for the phrase "cat in the hat" the skip-bigrams would be "cat in, cat the, cat hat, in the, in hat, the hat".

Table 3 shows the comparative metrics for the different models implemented in this paper.

5.2 Qualitative Analysis of generated summaries

We tested a sample summary and compared the results for the implemented models. These can be qualitatively examined to compare them against the human-generated summaries (for human evaluation). Following are the outputs for the same:

Table 3: Comparative Analysis of Proposed Models

SR No.	Model	Training Loss	Epoch	Step(s)	Validation Loss	Rouge1	Rouge2	RougeL	RougeLsum	Gen Len
1.	T5	1.6533	3	1200	1.4829	0.502	0.423	0.488	0.488	29.8224
2.	BART	1.2875	1.0	5754	1.6294	11.009	7.4618	10.5573	10.8087	58.3382
3.	Pegasus	1.5467	1.0	4592	1.5869	9.456	4.2069	8.6453	2.4338	38.1824

- **Text Input [2]:** This dry shampoo is amazing. When the water gave out I was so glad I had this at the ready. And I am here to tell you it works at keeping you feel clean even in the 10 degree temperatures most of us are going through right now (Feb17th). The smell is amazing and NO WHITE RESIDUE!! Instant volume and truly the best dry shampoo I've used in a while. I tried this product on a whim, and boy am I glad I did. This has quickly become one of my favorite dry shampoos. I have tried many brands from ojon to good hair day, both great products bu the price has prohibitive. This does a great job and is affordable. It last all day too! Can't believe it's under $10.
- **Baseline Summary:** The smell is amazing and NO WHITE RESIDUE!! Instant volume and instant volume and truly the best dry shampoo I've used in a while . When the water gave out I was so glad I had this at the ready . Can't believe it's under $10.
- **BART:** This dry shampoo is amazing. When the water gave out I was so glad I had this at the ready. And I am here to tell you it works at keeping you feel clean even in the 10 degree temperatures most of us are going through right now (Feb17th)
- **T5:** the smell is amazing and NO WHITE RESIDUE!! Instant volume and truly the best dry shampoo I've used in a while
- **PEGASUS:** This dry shampoo is amazing. I have tried many brands from ojon to good hair day, both great products bu the price has prohibitive.

Figure 6: BART

Figure 7: T5

Figure 8: PEGASUS

Figure 9: Training vs Loss statistics for the models implemented

6 Conclusion

Text Summarization has been an integral subset of NLP for the last few years now. Some techniques for text sumarization have included sentence scoring, graphical approach, machine learning, etc. Despite the many applications available, the main focus has been towards the generation of quality summaries, those which compare to human-level implementation upon generation. On top of these, the other focus has been towards efficiency of computation and resources. With this, pre-training has been done with models like T5, BART and Pegasus for general datasets, due to which these nodels have been adapt to unseen datasets quickly. Here, we have focused on training the same models for **Out-Of-Domain adaptation**, particularly for our dataset, thereby tailoring them for the desired end tasks. We trained these models on the E-commerce review dataset from Amazon, thereby generating abstractive summaries (Pegasus generates both abstractive and extractive summaries) for the given inputs, with human-level summaries for comparison on the given set of parameters. As part of the future scope of this project, we look towards the implementation of BERT and its variants on the same dataset (something we could not implement due to resource and memory constraints).

References

[1] Exploring transfer learning with t5: The text-to-text transfer transformer, Feb 2020.

[2] Paris best tour: The eiffel tower, Mar 2022.

[3] Hunar Batra, Akansha Jain, Gargi Bisht, Khushi Srivastava, Meenakshi Bharadwaj, Deepali Bajaj, and Urmil Bharti. Covshorts: News summarization application based on deep nlp transformers for sars-cov-2. In *2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, pages 1–6, 2021.

[4] Yisong Chen and Qing Song. News text summarization method based on bart-textrank model. In *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, volume 5, pages 2005–2010, 2021.

[5] Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 8-bit optimizers via block-wise quantization, 2021.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[7] Facebookresearch. Facebookresearch/bitsandbytes: Library for 8-bit optimizers and quantization routines.

[8] Mehrdad Farahani, Mohammad Gharachorloo, and Mohammad Manthouri. Leveraging parsbert and pretrained mt5 for persian abstractive text summarization. In *2021 26th International Computer Conference, Computer Society of Iran (CSICC)*, pages 1–6, 2021.

[9] Ziyin Gu, Li Chen, Qingmeng Zhu, Lingbo Li, Zelin Zhang, and Xin Zhou. Keyword-fusion pointer-generator network for policy text title summarization. In *2021 IEEE 4th International Conference on Information Systems and Computer Aided Education (ICISCAE)*, pages 78–81, 2021.

[10] Hritvik Gupta and Mayank Patel. Method of text summarization using lsa and sentence based topic modelling with bert. In *2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS)*, pages 511–517, 2021.

[11] Jiawen Jiang, Haiyang Zhang, Chenxu Dai, Qingjuan Zhao, Hao Feng, Zhanlin Ji, and Ivan Ganchev. Enhancements of attention-based bidirectional lstm for hybrid automatic text summarization. *IEEE Access*, 9:123660–123671, 2021.

[12] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, 2019.

[13] Tinghuai Ma, Qian Pan, Huan Rong, Yurong Qian, Yuan Tian, and Najla Al-Nabhan. T-bertsum: Topic-aware text summarization based on bert. *IEEE Transactions on Computational Social Systems*, pages 1–12, 2021.

[14] Rada Mihalcea and Paul Tarau. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[15] Quoc-An Nguyen, Quoc-Hung Duong, Minh-Quang Nguyen, Huy-Son Nguyen, Hoang-Quynh Le, Duy-Cat Can, Tam Doan Thanh, and Mai-Vu Tran. A hybrid multi-answer summarization model for the biomedical question-answering system. In *2021 13th International Conference on Knowledge and Systems Engineering (KSE)*, pages 1–6, 2021.

[16] Nlp-With-Transformers. Notebooks/06_summarization.ipynb at main · nlp-with-transformers/notebooks, Mar 2022.

[17] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2019.

[18] Juan Enrique Ramos. Using tf-idf to determine word relevance in document queries. 2003.

[19] Rahul Rawat, Pranay Rawat, Vivek Elahi, and Amaan Elahi. Abstractive summarization on dynamically changing text. In *2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*, pages 1158–1163, 2021.

[20] Ming-Hsiang Su, Chung-Hsien Wu, and Hao-Tse Cheng. A two-stage transformer-based approach for variable-length abstractive summarization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2061–2072, 2020.

[21] Thomas Wolf. Gradient accumulation.

[22] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization, 2017.

[23] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization, 2019.