– Independent Work Report Spring 2022 –

Text Summarization with small data

Lucas Irwin
Adviser: Ilia Sucholutsky

Abstract

Text summarization is perhaps the natural language processing application with the most obvious real-world applications. From law to government to finance, countless industries stand to benefit from the accurate and efficient summarization of documents. In the past 5 years, new pre-trained models such as Google's BERT [6] (Bidirectional Encoder Representations from Transformers) and OpenAI's GPT-3 [13] (Generative Pre-trained Transformer) have made it possible to train text summarization models on much smaller data sets by allowing users to simply fine-tune pre-trained models using transfer learning. At the same time, the release of newer, higher quality data sets such as the Big Patent data [5] set has improved the performance of the more challenging abstractive summarization task. Given these developments, my project aims to improve the performance of abstractive text summarization even further by fine-tuning Facebook AI's BART model [4] on the new, previously unused Big Patent data set. The improvements in abstractive summarization performance seen in this project bring us closer to democratizing the power of natural language processing by allowing smaller companies, NGOs and local governments to use the latest, most advanced models with smaller amounts of data so as to avoid accruing exorbitant training costs.

Contents

1. Motivation and Goal:

Among natural language processing applications, few have as many obvious real world applications as text summarization. Whether it be the summarization of news articles to inform financial investments or the summarization of legal documents to accelerate the work of pro-bono legal firms and NGOs, text summarization has enormous potential to impact society. Over the past 5 years, new Transformer-based models such as Google's BERT [6] (Bidirectional Encoder Representations from Transformers)and OpenAI's GPT-3 [13] (Generative Pre-trained Transformer) have made it possible to achieve unprecedented text summarization performance by fine-tuning pre-trained models on just tens of thousands of data points using a couple of hundred dollars, saving AI practitioners the millions of dollars and billions of data points required to train natural language understanding models from scratch. The process is known as "transfer learning" and it holds the potential to truly democratize AI and machine learning by allowing smaller SMEs, NGOs and local governments to use the most advanced natural language processing models without accruing exorbitant training costs.

At the same time, the release of newer, higher quality data sets such as the Big Patent data set [5] has created the potential to improve text summarization performance even further. This is due to the fact that new data sets have improved underlying structures which allows natural language models trained on them to better take advantage of the global features of the data set. Given these developments, I hypothesize that fine-tuning and training Transformer models on structurally superior data sets such as the Big Patent data set will have a significant impact on text summarization performance, and hence provide a strategy for improving text summarization performance in the future.

The goal of my project is to train BART (Bidirectional and Auto-Regressive Transformers)[4] – a state-of-the-art, pre-trained natural language model from Facebook AI – on the previously unused

Big Patent data set [5] in an attempt to achieve better abstractive summarization performance. The improved performance is expected due to the structural improvements inherent in the Big Patent data set which improves upon the weaknesses of existing text summarization data sets from the news domain by (i) ensuring that important information is dispersed throughout input documents and (ii) by minimizing the amount of information that is repeated verbatim from the input documents in the predicted summaries. To enable the training of the BART model on Big Patent, I utilize the Pytorch Lightning library to fine-tune the model via transfer learning and evaluate the results using the NLTK and Hugging Face datasets libraries. Success is measured by comparing the ROUGE metric scores [2] achieved in my project with the Big Patent data set to the ROUGE scores achieved by the authors of BART on previous news data sets such as the CNN/Daily Mail [1] and XSumm [2] data sets. The substantial improvements in ROUGE Precision scores achieved by training BART on just $10-20\%$ of the Big Patent data set confirm my hypothesis that the structure of the data matters for the performance of text summarization models, and inform future strategies on how to improve performance in the text summarization field.

2. Problem Background and Related Work:

Previous work in the field has centered around fine-tuning pre-trained, natural language models on abstractive text summarization. In 2017, Pointer-Generator Networks were introduced by See et al. as an improvement over sequence-to-sequence models, exceeding previous abstractive summarization performance by 2 or more ROUGE points on the CNN/Daily Mail data set. More recently, Liu and Lapata introduced the BERTSUM model [1] – a fine-tuned version of Google's BERT – which achieved state-of-the art summarization performance on the CNN/Daily Mailand NYT data sets. However, this performance was ultimately outdone by Lewis et al.'s BART model [4] which beat BERTSUM on the CNN/Daily Mail data set later that year. BART's superior, state-of-the-art performance influences my choice to use it as my pre-trained model in my project. [8]While

[1] https://huggingface.co/datasets/cnn_dailymail
[2] https://huggingface.co/datasets/xsum

previous work has centered heavily on news data sets such as the CNN/Daily Mail and XSumm data sets, my work takes advantage of the novel Big Patent data set due to its superior structural features.

The two main types of text summarization are extractive and abstractive summarization, both of which I define below:

2.1. Extractive Summarization:

If we have a document, D, consisting of a set of sentences $[s_1, s_2, ..., s_n]$, extractive summarization can be defined as a binary classification task which assigns a label $y_i \in 0, 1$ to each sentence depending on whether the model believes it should be included in the summary or not. [1] In other words, the model selects a subset of sentences from D to be included in the summary.

2.2. Abstractive Summarization:

In contrast to extractive summarization, abstractive summarization models generate a rewritten, paraphrased version of the summary using their own knowledge of vocabulary. Abstractive summaries benefit from being more analogous to the summarization process we use as humans, but suffer from being more challenging and less reliable to evaluate. This project will focus on improving abstractive summarization due to its readability and its greater similarity to the human summarization process, as well as the focus placed on it in previous work.

2.3. Transfer Learning:

Transfer Learning hinges on the idea that it is possible to store knowledge acquired by a learning algorithm in solving one task to solve another different but related task. This mirrors how we learn as humans since we are often able to apply the knowledge we have gained in one task to another similar task without needing to re-learn how to do it from scratch. Formally, we can define transfer learning using source and target domains and source and target learning tasks.

If we have a source domain, D_s, with its associated learning task, T_s, as well as a target domain D_τ and its associated learning task, T_τ and

$$D_s \neq D_\tau; T_s \neq T_\tau \tag{1}$$

then transfer learning learns the function, $f_\tau(\cdot) \rightarrow$ for the target domain by taking advantage of the source domain (D_s) and the source learning task (T_s). [3]

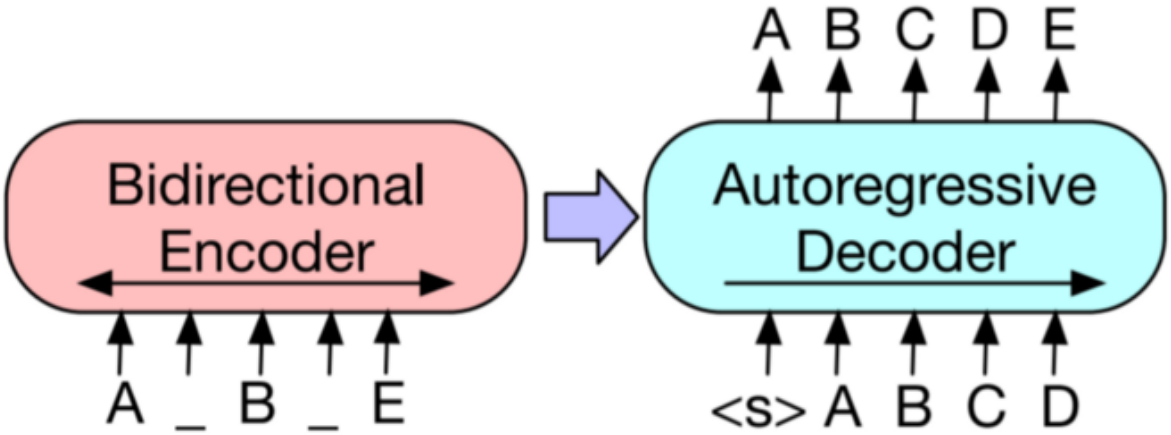

Figure 1: Bidirectional Auto-Regressive Transformers (BART)

2.4. BART model:

The BART model was developed by Lewis et al. as part of Facebook AI's research.

2.4.1. Architecture: BART uses a regular sequence-to-sequence architecture, consisting of a bidirectional encoder that mirrors Google's BERT as well as a left-to-right decoder like OpenAI's GPT-3. This encoder-decoder architecture draws on the benefits of both models. [4] [10]

2.4.2. Pre-training: BART is pre-trained on the cross-entropy loss calculated between the decoder output and the input data. The pre-training phase consists of a masked language modeling (MLM)

[3] https://livebook.manning.com/book/transfer-learning-in-action/copyright-2021-manning-publications/v-1/

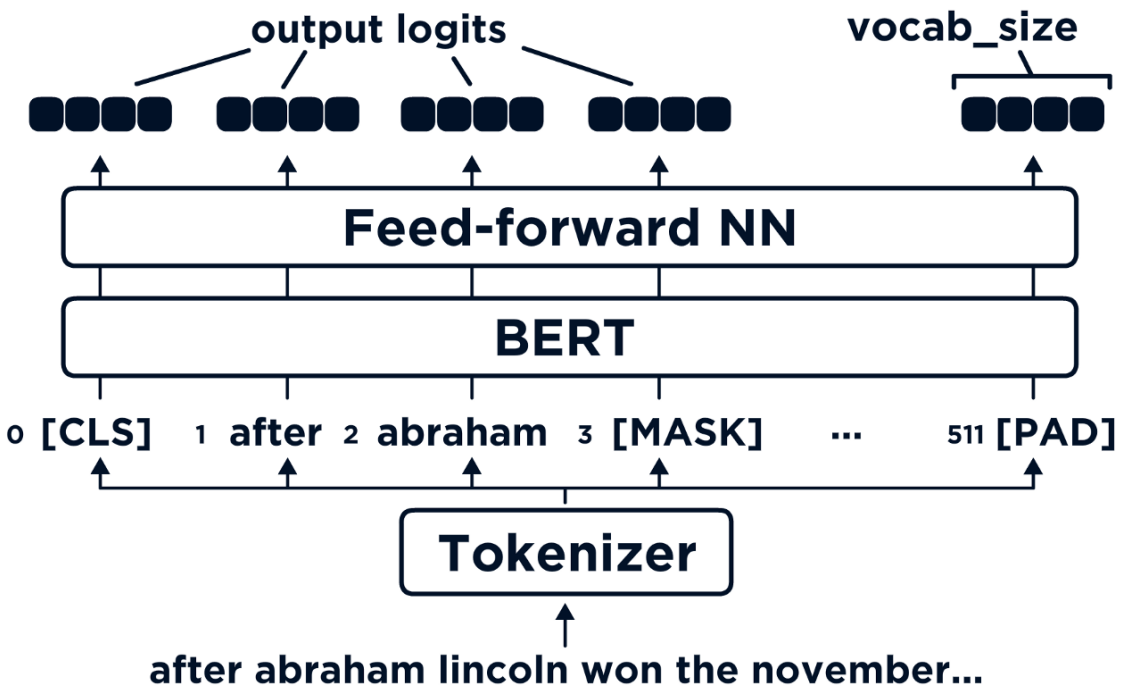

Figure 2: Masked Language Modeling [4]

process in which input sentences are shuffled randomly and spans of text are replaced with mask tokens ([*MASK*]). The model then predicts the original tokens in place of the ([*MASK*]) tokens. [4]

2.5. ROUGE score:

The most commonly used evaluation metric for both extractive and abstractive summarization in previous work [4] [6] is the ROUGE [2] (Recall-Oriented Understudy for Gisting Evaluation) score. ROUGE consists of a set of metrics which compare the overlap between n-grams in the model's summaries and the reference summaries used in training. In particular, ROUGE-1 compares the unigrams present in the model and reference summaries, while ROUGE-2 compares the bigrams and ROUGE-L calculates the longest common subsequence shared between the model and the reference summaries.

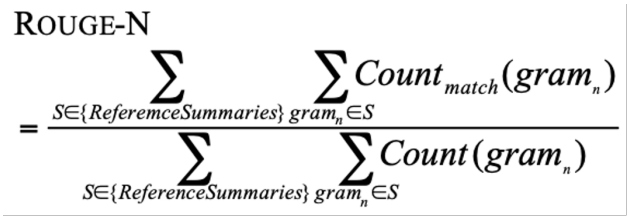

Figure 3: ROUGE metric

2.6. Evaluation Metrics

Since text summarization can be defined as a binary classification task, ROUGE calculates precision, recall and F-1 scores on the predicted summaries. Specifically, I define precision, recall and F_1 score as:

$$precision = \frac{TP}{TP+FP}, recall = \frac{TP}{TP+FN} \tag{2}$$

$$F_1 = 2\frac{precision.recall}{precision+recall} = 2\frac{TP}{2TP+FP+FN} \tag{3}$$

where TP = true positive, FP = false positives, FN = false negatives.

In the context of text summarization

$$precision = \frac{shared}{shared+incorrect}, recall = \frac{shared}{shared+missing} \tag{4}$$

where shared = n-grams appearing in both the model and the reference summary, missing = n-grams present in the reference summary but not the model summary and incorrect = n-grams appearing in the model but not the reference summary. [12]

Hence, precision captures the proportion of relevant n-grams in the predicted summary and recall captures how many n-grams were missed in the predicted summary that are present in the reference summary. [12]

3. Approach

3.1. Data Set

In this project, I make use of the Big Patent dataset [5] :

https://huggingface.co/datasets/big_patent

The data set consists of 1.3 million patent documents along with their human-written abstractive summaries. There are nine different classification categories ranging from textiles to electricity each with between 10K to 250K training pairs. The data set improves upon previous data sets from the news domain such as the CNN/Daily Mail and XSum data sets by addressing inherent structural issues in two key ways:

3.1.1. Dispersed summary material: First, relevant summary material is dispersed throughout the documents and not just concentrated at the beginning of the documents as it is in news data sets. This ensures that learning algorithms which can exploit the global structure of a data set in training

Dataset	# Doc	Comp. ratio	Dens.	Summary		Doc
				# word	# sent	# word
CNN/DM	312,085	13.0	3.8	55.6	3.8	789.9
NYT	654,788	12.0	2.4	44.9	2.0	795.9
NEWSROOM	1,212,726	43.0	9.5	30.4	1.4	750.9
XSUM	226,711	18.8	1.2	23.3	1.0	431.1
ARXIV	215,913	39.8	3.8	292.8	9.6	6,913.8
PUBMED	133,215	16.2	5.8	214.4	6.9	3,224.4
BIGPATENT	1,341,362	36.4	2.4	116.5	3.5	3,572.8

Figure 4: Big Patent vs news data sets [3]

achieve maximum possible performance. [5]

3.1.2. Smaller extractive segments: Second, whereas there are large parts of input documents which are repeated verbatim in the summaries of news data sets, the Big Patent data set contains much shorter summary extracts in its input documents and hence results in less regurgitation. This also improves the quality of training since the task of reconstructing abstractive summaries is more challenging than exracting information from the input document. [5]

In order to keep the training time reasonable and to ensure that the size of the training data was comparable to previous data sets such as the CNN/Daily Mail and XSum data sets, I chose to train BART on two subsets of the Big Patent data. First, I trained BART on category 'a' which corresponded to patents under the "human necessities" category and consisted of 174,134 training pairs, 9,675 testing pairs, and 9,674 validation pairs. Second, I also trained the model on category 'g' which corresponded to patents under the "physics" category and consisted of 258,935 training pairs, 14386 testing pairs, and 14385 validation pairs.

3.2. Data Processing

Thanks to the Hugging Face libraries, the data processing phase was simple and straightforward. Since the Big Patent data set is included in the Hugging Face datasets library, I simply loaded the data into Google Colab and separated it into training and testing sets. The BART tokenizer class took care of the encoding of the data in preparation for the fine-tuning process as well as the decoding necessary for evaluation.

3.3. Method:

The approach I used was a transfer learning process which consisted of three main steps:

1. **Fine-tuning** the BART model.
2. **Training** the BART model on the Big Patent data set.

3. **Evaluating** the predictions by calculating the ROUGE scores between the model predictions and the human-written reference summaries.

To train the model, I used Cross Entropy loss as the loss function and Adam as the optimizer with a learning rate of $2e^{-5}$. I first trained the model on a single epoch but then increased this to 5 epochs in an effort to improve performance. To evaluate the model, I calculated the ROUGE scores between the predicted summaries and the reference summaries. The detailed technical description of my implementation follows in the next section.

4. Technical Implementation:

4.1. Pytorch Lightning Module:

The first step in my implementation was to create a Pytorch Lightning module to be used for fine-tuning and training the model. Instead of writing the module from scratch, I adapted BART fine-tuning code from a different but related open-source project. [5] The methods included in the module were an initialization method (which took the learning rate, tokenizer, model and hyperparameters as arguments), a forward method, a training step, a validation step, an optimizer configuration step, a freeze parameter method and a freeze embedding method. The module fulfilled the function of a Pytorch training class while having the added advantage of automating much of the training process by handling tasks such as updating the optimizer and zeroing the gradients.

4.2. Data Loading Module:

Next, it was necessary to write the dataloading module which would be used to fit the model to the data using the Pytorch Lightning class. Once again I adapted code from the aforementioned project to accomplish this. The data loading module took the tokenizer, the data file, the batch size and the number of examples as arguments and fulfilled the tasks of splitting the data into training, testing

[5] https://colab.research.google.com/drive/1Cy27V-7qqYatqMA7fEqG2kgMySZXw9I4?usp=sharingpli=1

and validation sets, encoding sentences in the input documents and loading the different splits into Pytorch Dataset objects.

4.3. Loading the BART model:

After adapting these two modules, I loaded the pre-trained BART model from the Hugging Face Transformers library. I loaded the BartForConditionalGeneration base model as well as the tokenizer which was necessary for encoding and decoding the sentences in the input documents.

4.4. Loading the Big Patent data set:

Next, I used the Hugging Face datasets library to also load the Big Patent data set. I first loaded configuration 'a' which corresponded to the subset of patents under the category of human necessities and then configuration 'g' which corresponded to physics patents.

4.5. Data pre-processing:

Upon loading the Big Patent data, it was necessary to do some pre-processing in order to prepare the data for training and testing. First, I separated the training and testing sets. Next, I renamed the columns so that they matched the titles required by the Pytorch Lightning module in training (The module required the title "source" for the input documents and "target" for the output summary). Finally, I saved the files in csv format in preparation for loading them into the data module.

4.6. Training:

At this point, the model was ready for training. I loaded the training data into the data module and initialized the model using the Pytorch Lightning class with a learning rate of $2e^{-5}$. Initially, I used a batch size of 16 but upon realizing that this would often deplete the available RAM, I reduced the batch size to 8.

Initially, I attempted to train the model on the entire Big Patent data set but this led to inordinate training times and resulted in Google Colab crashing due to insufficient RAM. It also did not make

much sense from an evaluative standpoint since the Big Patent data set was around 10 times larger than previous data sets from the news domain. Therefore, I decided to first train BART on the 'a' configuration of the data set which corresponded to the 'human necessities' category and contained 174,134 training pairs, 9,675 testing pairs and 9,674 validation pairs, and then to train it on the 'g' configuration of the data set which corresponded to the 'physics' category and contained 258,935 training pairs, 14,386 testing pairs, and 14,385 validation pairs.

Another problem I encountered was that the training loss was not minimized sufficiently with a single epoch, so I increased the number of epochs to 5 and retrained the model. The improvement in performance achieved with this modification made up for the increase in training time.

4.7. Prediction:

After training the model, I adapted code from a Hugging Face blog post [6] to generate summary predictions for the input documents in the test sets.

Initially I attempted to generate predictions using the Hugging Face Trainer.predict method, but faced great difficulty in decoding my predictions for evaluation as the method returned encoded output tokens instead of regular text predictions. Therefore, I resorted to using the text_predictions method from the aforementioned blog post which made use of the BART tokenizer decode method to automatically return text sentences as predictions instead of encoded tokens.

4.8. Evaluation:

Once I had generated my predictions, all that was left to do was to evaluate the results of the fine-tuned BART model by calculating ROUGE scores. To accomplish this, I imported the rouge-score metric from the Hugging Face datasets library. Next, I sliced the reference summaries from the test sets so they could be used as labels, and used the NLTK library to tokenize the predictions and the labels and add a newline between each entry (since this is what the rouge-score method required). It

[6] https://github.com/huggingface/transformers/issues/3853

	ROUGE-1	ROUGE-2	ROUGE-L
Precision	45.52	11.51	39.20
Recall	13.66	3.38	11.59
F-1 Score	20.03	4.97	17.06

Figure 5: Big Patent category A results

was challenging to format the data correctly, but upon preparing the predictions and the labels I was able to calculate ROUGE-1, ROUGE-2 and ROUGE-L scores for both the category 'a' and 'g' subsets of the Big Patent data.

5. Results:

The fine-tuning and evaluation process was completed a total of three times, resulting in significant results which in some cases exceed BART's performance on both the CNN/Daily mail and XSum data sets.

Firstly, the model was fine-tuned and trained on the 'a' subset of the Big Patent data set for a single epoch. The training time for this round was approximately 2 hours. The results were promising, but they did not exceed the scores achieved on the CNN/Daily Mail and XSum data sets which are shown in Figure 6. The most promising result was the ROUGE-1 precision score which was 43.65 but this was still less than the 44.16 and 45.14 reported on the CNN/Daily Mail and XSum data sets respectively.

Next, I trained the model on the 'a' subset again but this time for five epochs. The training time increased to approximately 5 hours, but this was compensated for by the significant increase in performance observed. The most significant results achieved were a 45.52 ROUGE-1 precision score which exceeded both the 44.16 and 45.14 achieved on the CNN/Daily mail and XSum data sets respectively. Additionally, the ROUGE-L precision score achieved exceeded the score reported on XSum by approximately 2 ROUGE points and came within 2 points of the CNN/Daily Mail data

set.

After obtaining positive results on the category 'a' split of the Big Patent data set, I fine-tuned and trained the model on the category 'g' subset of the data. The 'g' subset corresponded to patents under the 'physics' category and contained 258,935 training pairs compared to the 174,134 in the category 'a' subset. I decided to omit training on a single epoch due to the improved performance obtained with 5 epochs on the 'a' subset. Training time was approximately 9 hours.

The results achieved on the 'g' subset were even more promising. The model achieved a 46.50 ROUGE-1 precision score which beat BART on the CNN data set by 2 ROUGE points and by 1 ROUGE point on XSum. The second significant result was a ROUGE-L score of 39.99 which beat the XSum data set by more than 2 ROUGE points and came within 1 ROUGE point of the CNN/Daily Mail data set.

The state-of-the-art results achieved with only $10-20\%$ of the Big Patent data set suggest that my initial hypothesis was correct. The structural features of a data set have a significant impact on abstractive summarization performance and suggest that the improvement of data can be viewed as a means to making text summarization more accessible to organizations with limited access to funding.

	CNN/DailyMail			XSum		
	R1	R2	RL	R1	R2	RL
Lead-3	40.42	17.62	36.67	16.30	1.60	11.95
PTGEN (See et al., 2017)	36.44	15.66	33.42	29.70	9.21	23.24
PTGEN+COV (See et al., 2017)	39.53	17.28	36.38	28.10	8.02	21.72
UniLM	43.33	20.21	40.51	-	-	-
BERTSUMABS (Liu & Lapata, 2019)	41.72	19.39	38.76	38.76	16.33	31.15
BERTSUMEXTABS (Liu & Lapata, 2019)	42.13	19.60	39.18	38.81	16.50	31.27
BART	**44.16**	**21.28**	**40.90**	**45.14**	**22.27**	**37.25**

Figure 6: BART results [4]

	ROUGE-1	ROUGE-2	ROUGE-L
Precision	46.50	10.55	39.99
Recall	11.89	2.56	10.09
F-1 Score	18.31	3.99	15.59

Figure 7: Big Patent category G results

However, there are also significant limitations to my results in the form of low ROUGE-2 scores as well as relatively low ROUGE-1 and ROUGE-L recall scores. The low recall scores also cause my F-1 scores to be low.

The low ROUGE-2 scores are a limitation to my work and can be improved upon with more optimized training methods. On the other hand, the discrepancy between precision and recall can be accounted for by the fact that my predicted summaries are much shorter on average than the reference summaries. Since $precision = \frac{TP}{TP+FP}$, it captures the proportion of n-grams shared by the predicted and the reference summaries over the total number of n-grams in the predicted summaries. Therefore, the high precision scores show that my model is capturing a high proportion of relevant n-grams in its predicted summaries. Recall can be trivially optimized for by producing the entire document verbatim since recall is equivalent to the overlap in n-grams divided by the total number of n-grams in the reference summary, so a very long summary including all n-grams in the reference summary along with a large number of useless n-grams would produce a perfect recall score. Hence we care more about precision than recall when evaluating model summaries since we want summaries to represent a high proportion of relevant information, and this explanation accounts for the focus on precision in my work.

In order to further remedy this limitation, I decided to do human evaluation of the predicted summaries by comparing them to the human-written reference summaries. Listed in Figure 8 are four example summaries which corroborate my hypothesis that the size of the predicted summaries is responsible for the low recall scores. In all cases, the predicted summaries closely mirror the

Reference Summary	Predicted Summary
a waste form for and a method of rendering hazardous materials less dangerous is disclosed that includes fixing the hazardous material in nanopores of a nanoporous material, reacting the trapped hazardous material to render it less volatile / soluble, and vitrifying the nanoporous material containing the less volatile / soluble hazardous material.	a method of making nanostructured glass - ceramic waste forms that can be used for disposition and disposal of hazardous material, such as radionuclides.
bone anchor implantation devices and methods for their use are disclosed . the bone anchor implantation devices have an ergonometric and / or rotatable handle. the bone anchor implantation devices and methods find particular application for implanting a bone anchor for maintaining or improving urinary continence by suspending or stabilizing the bladder neck.	a device for implanting bone anchor into bones includes an elongated member having first and second ends, the end being configured to be inserted in tissue
an external bone fixing member includes an external reconstruction plate to replace the conventional large external support frame to reduce the volume of the external fixing member and the external bone fixing member is cooperated with a screw which is angle adjustable so as to meet requirements of fixation of the fractured bones at different positions or comminuted fractures. the use of the screw is further cooperated with the reconstruction plate to reinforce the stability of the connection between the reconstruction plate and the screw to prevent the screw from withdrawing, moving, angle changing or movement of the reconstruction plate.	an external bone fixing member for use in reconstructing a fractured or broken femur includes at least one reconstruction plate, and preferably two connection members.
an orthodontic bracket, a method of manufacture and method of installing the bracket. the bracket is provided with visually enhanced reference edges for assisting in alignment of the bracket with respect to the tooth.	an orthodontic bracket includes a base member having first and second ends, the upper end being configured to be mounted on one of two opposing teeth.

Figure 8: Predicted vs Reference summaries

content of the reference summaries but miss out some information due to the restrictions on their size. However, overall the predicted summaries capture the same meaning as the reference summaries, proving that they are valuable in spite of the low recall scores. Another improvement of my work could also be to consider leveraging greater computational resources to increase the size of the predicted summaries in the hope of improving recall scores.

6. Conclusion:

6.1. Strengths:

In this project, I set out to fine-tune and train BART on the Big Patent data set in order to improve abstractive summarization performance. The improved performance was expected due to the structural improvements inherent in the Big Patent data set, which improves upon the weaknesses of existing text summarization data sets from the news domain by (i) ensuring that important information is dispersed throughout the document and (ii) reducing the amount of summary material repeated verbatim in predictions. I utilized the Pytorch Lightning library to fine-tune the model via transfer learning and evaluated the results by comparing the ROUGE scores achieved in my project to the ROUGE scores achieved by the authors of BART on previous news data sets such as the CNN/Daily Mail and XSumm data sets.

Promising results were achieved on both the category 'a' and category 'g' subsets of the Big Patent data set, with ROUGE-1 precision scores beating the BART model's performance on both the CNN/Daily Mail and XSum data sets by 1-2 points and ROUGE-L scores beating the XSum data set by 2 points. The significant improvements in performance achieved in this project have proven my initial hypothesis that the structure of a data set is relevant to the performance of state-of-the-art abstractive text summarization models. The results thus inform future strategies for improving text summarization performance by shifting the focus from improving models to improving the structure of the underlying data as an alternative solution.

6.2. Limitations and Further Work:

Despite promising ROUGE-1 and ROUGE-L scores, the model achieved relatively low ROUGE-2 scores. This could be improved by altering the hyper-parameters, changing the learning rate or increasing the number of epochs in future work. Leveraging greater computational power than was

available in this project could also help address this limitation.

Secondly, although my model achieved state-of-the-art ROUGE-1 precision scores, it exhibited relatively low recall scores. However, this can be accounted for by the fact that my model's summaries had to be restricted in size or else the RAM would max out in Google Colab. To confirm this, human evaluation of the results was undertaken by comparing the content of the predicted summaries to the reference summaries. In all cases, the predicted summaries closely mirrored the content of the reference summaries but missed out on some information included in the references due to their smaller size. Therefore, future work could leverage greater computational resources and funding to increase the maximum length of the predicted summaries in the hope of improving recall scores.

Finally, it is worth mentioning that the results presented in this project were achieved by fine-tuning BART on just $10-20\%$ of the Big Patent data set. Given the promising ROUGE-1 scores achieved by training BART on just a fraction of the overall data set, it would be fruitful to consider training the model on larger subsets of Big Patent with a view towards improving abstractive summarization performance even further in future work. Nevertheless, my project proves that the improvement of the structure of data sets can be thought of as a means to improve abstractive summarization performance. These improvements are of great value as they bring us closer to democratizing natural language processing by enabling smaller organizations with limited funding to utilize the latest and most advanced models without accruing inordinate costs.

7. Acknowledgements:

I would like to thank my independent work advisor, Ilia Sucholutsky, for his guidance in helping me understand the fine-tuning process, assisting me in defining project requirements and providing me with reading material and Python libraries to complete this research project.

8. Honor Code:

I pledge my honor that I have not violated the Princeton honor code in this research project.

References

[1] Liu, Yang. "Fine-tune BERT for extractive summarization." arXiv preprint arXiv:1903.10318 (2019).

[2] Lin, Chin-Yew. "Rouge: A package for automatic evaluation of summaries." Text summarization branches out. 2004.

[3] See, Abigail, Peter J. Liu, Christopher D. Manning, "Get to the point: Summarization with pointer-generator networks." arXiv preprint arXiv:1704.04368 (2017).

[4] Lewis, Mike, et al. "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension." arXiv preprint arXiv:1910.13461 (2019).

[5] Sharma, Eva, Chen Li, and Lu Wang. "BIGPATENT: A large-scale dataset for abstractive and coherent summarization." arXiv preprint arXiv:1906.03741 (2019).

[6] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

[7] Lin, Yuan-Pin; Jung, Tzyy-Ping "Improving EEG-Based Emotion Classification Using Conditional Transfer Learning". Frontiers in Human Neuroscience, 27 June 2017

[8] Huang, Dandan, Leyang Cui, Sen Yang, Guangsheng Bao, Kun Wang, Jun Xie, and Yue Zhang. "What Have We Achieved on Text Summarization?." arXiv preprint arXiv:2010.04529 (2020).

[9] Nallapati, Ramesh, Bowen Zhou, Caglar Gulcehre, and Bing Xiang. "Abstractive text summarization using sequence-to-sequence rnns and beyond." arXiv preprint arXiv:1602.06023 (2016).

[10] Chen, Yisong, 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), vol. 5, pp. 2005-2010. IEEE, 2021.

[11] Moratanch, N., and S. Chitrakala. "A survey on extractive text summarization." In 2017 international conference on computer, communication and signal processing (ICCCSP), pp. 1-6. IEEE, 2017

[12] Tas, O. , Kiyani, F. "A SURVEY AUTOMATIC TEXT SUMMARIZATION". PressAcademia Procedia 5 (2017): 205-213

[13] Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan et al. "Language models are few-shot learners." Advances in neural information processing systems 33 (2020): 1877-1901.