Podcast Summarization with NPR Transcripts

Kaavya Shah
University of California, Berkeley
School of Information
kaavyashah@berkeley.edu

Joyce Li
University of California, Berkeley
School of Information
joyceml@berkeley.edu

Abstract

Podcasts have become a widespread form of entertainment and news media, and present unique problems to solve in the NLP space. This paper details the process of summarizing NPR and Spotify podcast transcripts by using TextRank extractive summarization to prune transcript lengths, followed by a fine-tuned BART-CNN model for abstractive summarization. Our final model improved significantly from our baseline and achieved results close to state-of-the-art summarization models.

1 Introduction

Podcasts are increasingly popular in mainstream media as a medium for entertainment and information. Created by amateur and professional hosts, podcasts cover every possible topic. Many mainstream podcasts are posted with transcripts, and all creators are required to have a summary or description of the podcast episode to publish their episodes. Naturally, the breadth of transcript and summary data coming from podcasts is ideal for summarization tasks. However, summarizing podcasts is a challenging task due to the conversational nature and length of podcasts.

The two types of summarization tasks are extractive summarization and abstractive summarization. In extractive summarization, summaries are created by selecting important sentences only from the input text. In abstractive summarization, summaries are generated and not restricted to phrases and text from the input text. Summaries are analyzed based on the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) (Lin, 2004) scores, which automatically compares a generated summary to reference summary. The ROUGE metrics we use are ROUGE-1, ROUGE-2, and ROUGE-L F-measure scores.

In this paper, we combine extractive and abstractive summarization techniques to tackle the challenges of long documents and conversational language in podcasts. We use a dataset of NPR official podcast transcripts and transcribed Spotify podcast text for training and testing purposes. We found that combining the extractive TextRank and abstractive fine-tuned BART-CNN achieved successful results with room for improvement.

2 Background

2.1 Extractive Summarization Literature Review

Much research has been done on extractive summarization tasks. TextRank (Mihalcea and Tarau, 2004) is an unsupervised algorithm that creates a graphical representation of sentences and computes highest importance sentences using PageRank. This algorithm performs well for extractive summarization because it does not rely solely on local context but on the entire context of the document. TextRank is able to identify connections between various entities in a text, and does not require training on any domain or linguistic data.

A readily available extractive summarizer for Python is the bert-extractive-summarizer [1]. This model uses inference from pre-trained BERT libraries to produce embeddings for topic clusters that then produce summaries of lecture transcripts (Miller, 2019). Issues that this model faces are that there are no golden truth summaries of lectures, human evaluation is required, and the small ratio of sentences chosen to summarize a long lecture does not fully explain the topic. However, this model is still relevant to our task because it was trained on lecture content, which is spoken material just like podcasts.

BERTSUM (Liu, 2019) is another modification of BERT for extractive summarization. BERTSUM inserts [CLS] tokens at the beginning of each sentence and [SEP] tokens at the end, and assigns odd

[1] https://github.com/dmmiller612/bert-extractive-summarizer

or even segment embeddings to each input sentence. Finally, simple classifier layers, inter-sentence transformer layers, or Recurrent Neural Net layers are stacked on top of the BERT outputs. The BERTSUM model with inter-sentence Transformer layers performs best, returning a state of the art ROUGE-L score of 39.63 on the CNN/Dailymail dataset (Nallapati et al., 2016).

2.2 Abstractive Summarization Literature Review

Abstractive summarization has proven to be the more difficult summarization task. T5 (Raffel et al., 2019) is a prominent model that uses transformer architecture, built on self-attention. It treats all text processing tasks as text-to-text models, which allows us to apply the same model, objective, training procedure, and decoding process to various tasks. T5 has a maximum sequence input of 512 tokens, which raises difficulties given the length of podcasts.

BART-CNN (Lewis et al., 2019), a Bidirectional and Auto-Regressive Transformer, is trained on the CNN/Dailymail dataset[2], which is one of the largest datasets with articles and corresponding summaries. It uses a standard transformer-based neural machine translation architecture, and is a sequence-to-sequence model with a bidirectional encoder over corrupted text and a left-to-right autoregressive decoder. We chose to use BART because it has a consistently strong performance; BART achieves a ROUGE-1 score of 44.16, ROUGE-2 score of 21.28, and ROUGE-L score of 40.90 on CNN/Dailymail dataset. It takes in 1024 tokens as an input, which is greater than alternatives such as T5 which only take in 512 tokens, making it better for longer document summarization.

2.3 Related Work

Minimal research has been conducted on podcast summarization. The most successful prior work on podcast summarization was completed by Clifton et al. (2020). 100,000 Spotify-owned podcasts that were posted between January 1, 2019 and March 1, 2020 were randomly sampled, and then transcribed using the Google Cloud Speech to Text API. Creator generated descriptions are used as reference summaries; however, not all descriptions are informative summaries. This dataset was summarized using 4 models: First Minute, which returns all the text from the first minute of transcripts, TextRank, BART-CNN [3], and BART-PODCASTS, a BART-CNN model that is fine-tuned on the Spotify dataset.

3 Methods

3.1 Data

We chose to use the Spotify Podcast Dataset and NPR transcripts as our datasets.

The Spotify Podcast Dataset [4] contains 100,000 podcasts from the following genres: Comedy, Sports, Health Fitness, Society Culture, Education, Science, News Politics, Government Organization, and Fiction (Clifton et al., 2020). It contains only English podcasts with cloud-generated transcripts and creator descriptions. After obtaining data from the Spotify Podcasts team, we parsed through transcript files to obtain a portion of the Spotify gold dataset, which was used by Spotify as a test set that was evaluated by humans. We decided to use this as a test set for our own model to understand how well a fine-tuned model on NPR would perform on a different set of transcripts.

NPR is an independent, nonprofit media organization that shares local stories to over 25.1 million weekly listeners around the world [5]. With the company's permission to use transcripts from their website, we scraped 4180 English podcast transcripts and creator-generated descriptions from 18 different shows from 5 different genres: news politics, business, society culture, entertainment, and science tech. Rather than manually creating reference summaries after reading the podcast transcripts, we were able to use the creator descriptions as target summaries. The average podcast duration was 22 minutes, and the average transcript and description sentence lengths were 3706 words and 62 words respectively. See Figure 1 for a histogram of the distribution of original transcript lengths.

Because the NPR podcast transcripts and descriptions from the website were written by NPR contractors, we decided that this dataset was more accurate and reliable than cloud-generated transcripts. We used 90% of the NPR dataset to train the extractive model and fine-tune the abstractive model, and the remaining 10% to test the performance of

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

[3] https://huggingface.co/facebook/bart-large-cnn

[4] https://podcastsdataset.byspotify.com

[5] https://www.npr.org/about/

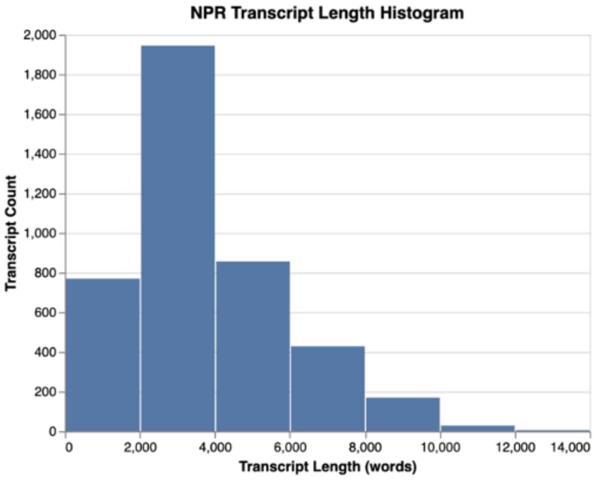

Figure 1: A histogram of original transcript word counts from the NPR dataset

our combined models.

After obtaining our data, we cleaned the messy transcript dialogue format. Using RegEx, we removed HTML tags, narrator names, soundbite tags, new lines, podcast outros, and new lines from transcripts and similarly cleaned up podcast descriptions.

3.2 Modeling

We used both extractive and abstractive summarization to produce podcast summaries. As proposed by Hsu et al. (2018), combining extractive and abstractive summarization methods returns very high ROUGE-F1 scores when evaluated on the CNN/DailyMail test set, as it generates the most informative and readable summaries. Models we were considering took in either 512 or 1024 tokens as an input, while the transcripts were on average over 3500 words long. Because the transcripts were written in spoken dialogue format, they contained many useless filler sentences and phrases. Our strategy was to use extractive summarization to prune unnecessary sentences from the dataset first and then run the data through a fine-tuned abstractive summarization model.

3.3 Baseline

As a baseline, we used the Bert Extractive Summarizer for lecture summarization (Miller, 2019) as the extractive model, and the Facebook BART-CNN (Lewis et al., 2019) as the abstractive base model. We used the Bert Extractive Summarizer as our first extractive summarizer because it was easily implemented in Python, trained on spoken text, performed well on lecture summarization, and used inference from language model BERT to produce a summary. After running our cleaned training data through the extractive summarizer and obtaining shortened transcripts, we ran this data through the BART-CNN large base model without fine-tuning to obtain final baseline summaries. We did not expect this model to perform extremely well because we did not fine-tune BART-CNN, and used the most readily-available but not necessarily the best extractive summarizer. We hoped to improve our baseline scores with our final model.

3.4 Final

We first attempted to use BERTSUM as our final extractive model because it achieves state-of-the-art results for extractive summarization on the CNN/DailyMail dataset. We noticed that our baseline summaries contained speaker tags learned from the lecture extractive summarization model, which was text that we wanted removed from our final model. We thought that using a model that is trained on the same dataset that BART-CNN is optimized on might bring more coherence to our model. However, BERTSUM required PyTorch's CUDA packages for parallel computing, which is not supported on MacOS. Due to this resource constraint, we looked into other options as an alternative extractive model.

We found a recently-published paper by Yadav et al. (2022), who detailed that TextRank actually performs as the best extractive summarization mode for longer texts compared to TF-IDF, BERTSUM, and LexRank. The paper based findings on news articles from the MultiNews dataset [6] and long texts from the Reddit-TIFU dataset [7], which are of similar length to podcast transcripts, and thus we decided that results could be generalized and applied to our project. Because TextRank is an unsupervised extractive model, we also did not have to worry about inconsistent lecture-specific speaker tags.

After running cleaned train data through TextRank [8] and deciding to limit extractive summaries to 20 sentences, we performed a sanity check to ensure that summary length was shorter than before.

[6] https://www.tensorflow.org/datasets/catalog/multi_news

[7] https://www.tensorflow.org/datasets/catalog/reddit_tifu

[8] https://derwen.ai/docs/ptr/explain_summ/

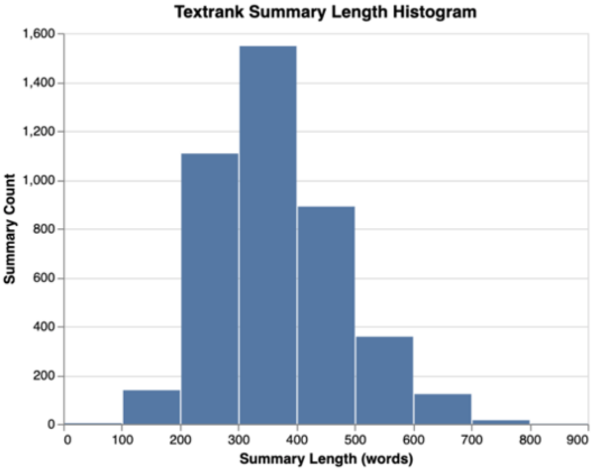

Figure 2: A histogram of transcript word counts after running TextRank on the NPR dataset

We found that summaries averaged 364 words, compared to the 3706 average words before extractive summarization, and all had lengths that were short enough to be entirely passed through BART-CNN's 1024 token limit. See Figure 2 to see the distribution of TextRank summary lengths.

We then fine-tuned BART-CNN on our entire training set of 3782 NPR transcripts on one epoch and produced the same number of final summaries for the training set. We used the fast.ai library [9] to optimize and streamline the model training process and the blurr package [10] to integrate the Huggingface BART-CNN model with fast.ai [11]. These libraries allowed us to fine-tune our model with our limited GPU resources as efficiently as possible. With the fine-tuned model, we generated summaries on both our cleaned NPR transcript test set and Spotify gold test set and evaluated ROUGE-1, ROUGE-2, and ROUGE-L F1 scores.

Lastly, rather than only looking at ROUGE scores, we also decided to manually evaluate a random sample of 10 summaries ourselves. We reviewed the podcast transcript, the creator generated description, the TextRank output, and then our final output summaries from the fine-tuned BART-CNN to determine whether summaries were concise and accurate to their respective transcripts.

4 Results and Discussion

Our final model's performance surpassed all of our baseline model's ROUGE measures. This makes sense because we were able to trim down all summaries to be under the 1024 token limit, and then pass them through a fine-tuned model. However, our model performed slightly worse than the BART-PODCASTS model from Clifton et al. (2020). Although our final model evaluated on our NPR testset had a higher ROUGE-1 score than the BART-PODCASTS model, the ROUGE-2 and ROUGE-L scores were slightly lower. Our model's evaluations on a small subset of Spotify Gold transcripts performed worse than BART-PODCASTS in all ROUGE scores. Refer to Table 1 for a comparison of ROUGE scores.

That being said, our final model's generated summaries were well-written upon human evaluation (refer to Table 2 in the Appendix for a few examples). We observed that the summaries gave a good general outline of podcast content, matched most main points from target summary, and were able to generalize details well. Despite not getting groundbreaking ROUGE scores, our summaries were representative of the transcript text.

4.1 Error Analysis

We attribute our model's lack of success compared to the BART-PODCAST model to our lack of training data from NPR. Although we scraped most of the possible transcripts from NPR, we only had 4180 transcripts, which were then split into a training and test set. The BART-PODCAST model was trained on almost 100,000 podcasts, which explains why ROUGE metrics were generally higher.

Upon inspecting our final summaries, we observed that they occasionally missed some main points and would generally contain an unrelated last sentence. We suspect that this may be caused by the lack of epochs that we trained on due to time and GPU constraints, since the model was not able to fully learn how to conclude summaries. Another reason why this may be the case is that we were not able to fully remove soundbites and other extraneous sounds in the podcast transcript. As a result, some unrelated sentences and phrases with similar key words but different topics may have remained in the TextRank and final summaries.

Another issue to note is that we did not always have perfect reference summaries, as some creator descriptions were not important or reflective of

[9] https://docs.fast.ai

[10] https://ohmeow.github.io/blurr/

[11] https://github.com/francoisstamant/Fine-tuning-for-text-summarization/blob/main/text_summarization.ipynb

Model	**R1-F**	**R2-F**	**Rl-F**
Baseline: Bert Extractive + BART-CNN on NPR test set	21.90	4.97	13.58
Final: TextRank + fine-tuned BART-CNN on NPR test set	29.93	10.36	19.39
Final: TextRank + fine-tuned BART-CNN on Spotify Gold	23.43	4.50	13.87
Clifton et al. (2020): BART-PODCASTS on Spotify Non-Brass	29.46	12.87	22.07

Table 1: ROUGE-F1 scores for our baseline, our final, and Clifton et al. (2020) Spotify BART-PODCASTS models.

the entire podcast. This caused very low ROUGE scores because some generated summaries were accurate to the transcript, but did not resemble the imperfect creator descriptions. The only workaround to this problem requires human generated summaries and human evaluation of all model generated summaries, which was infeasible. This led us to conclude that ROUGE scores may not necessarily be the best or most accurate measure of summary quality in our case, simply because similarity to creator-generated podcast summaries does not equate to having a high quality summary. Refer to Table 2 in Appendix for example summaries from the baseline and final model.

5 Conclusion

Through combining TextRank and a fine-tuned BART-CNN abstractive model on NPR transcripts, we were able to significantly improve our ROUGE scores from our preliminary baseline model. Our project is unique because it shows the advantage of using source transcripts, instead of transcriptions as proposed in the Spotify dataset. Additionally, we show that combining extractive and abstractive models can be effective when tasked with summarizing large text amounts.

Looking forward, there are many things we could improve in our model, given more time and resources. We suggest the following:

1. Include more official and accurate transcripts, from various publishers and creators, to train our model.

2. Include human evaluation of conciseness and accuracy of generated summaries as opposed to ROUGE scores.

3. Train the model on more epochs to improve summaries. Unfortunately, we were only able to run our model on one epoch due to GPU resource constraints.

4. Clean podcast transcripts more precisely. While we were able to remove most non-dialogue text in each transcript, we realized that we had still included soundbite speech and music in our final transcripts. This is a fundamental difficulty in working with dialogue text, because it is almost impossible to write generalizable code in regex to remove specific sentences with no clear patterns.

5. Get better reference summaries. This is a classic problem in many NLP tasks: getting quality target text values.

6. Combine extractive and abstractive models into one cohesive model, similar to the architecture proposed by Hsu et al. (2018), rather than keeping it in two separate parts.

However, even with room for improvement, our model obtained competitive results with state-of-the-art summarization models.

References

Ann Clifton, Sravana Reddy, Yongze Yu, Aasish Pappu, Rezvaneh Rezapour, Hamed Bonab, Maria Eskevich, Gareth Jones, Jussi Karlgren, Ben Carterette, and Rosie Jones. 2020. 100,000 podcasts: A spoken English document corpus. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5903–5917, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A unified model for extractive and abstractive summarization using inconsistency loss. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 132–141, Melbourne, Australia. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer.

2019. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yang Liu. 2019. Fine-tune BERT for extractive summarization. *CoRR*, abs/1903.10318.

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

Derek Miller. 2019. Leveraging BERT for extractive text summarization on lectures. *CoRR*, abs/1906.04165.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Divakar Yadav, Naman Lalit, Riya Kaushik, Yogendra Singh, Mohit, Dinesh, Arun Kr. Yadav, Kishor V. Bhadane, Adarsh Kumar, and Baseem Khan. 2022. Qualitative analysis of text summarization techniques and its applications in health domain. *Computational Intelligence and Neuroscience*, 2022.

A Appendix

See table on the next page.

	Text
1 - Creator Description	If you’ve ever posted a celebrity’s picture inside a locker, used a photo of them as your lock screen, or followed every moment of someone’s career, you know all about pop culture crushes. There are the ones you’re proud of, the ones you’re less proud of, and the ones that never quite go away. Today, we’re sharing some of our earliest crushes and how they informed our later adorations along the way.
1 - Baseline Summary	I love this person’s name all over my Trapper Keeper,” Aisha Harris says. ”One of the very first episodes actually was, I think, where I fell in love,” she adds. When weŕe young, we don’t always know enough to get our crushes from true comedy - like, from the true world of broad comedy. In the spirit of a good Glen Weldon taxonomy, I started to kind of scribble down the taxonomy of where a childhood crush comes from, like what it is rooted in. And it’s just a reminder of how basic I am now and have always been. ”I just felt like I should come and say hello because I did see you twice live at the Valley Forge Music Fair. And to see her go from ”Scary Movie” to these more layered performances that she’s had over the years
1 - Final Summary	In this episode, we’re talking about some of our first pop culture crushes on POP CULTURE HAPPY HOUR from NPR’s Culture Desk. There are the ones you’re proud of later, the ones that never quite go away, and the crushes that never go away.
2 - Creator Description	We know his rhetoric has been described as boundary breaking when it comes to race. But U.S. presidents have been enacting racist policies forever. So as President Trump wraps up his first (and maybe only) term in office, we’re asking: In terms of racism, how does he stack up to others when it comes to both words and deeds?
2 - Baseline Summary	Zelizer: I don’t think the kinds of things that we’re seeing with the president and the current administration are necessarily new or even necessarily unique. What’s different is that Donald Trump, again, says the ugly parts out loud. This week’s pod focuses on presidential administrations from the past 50 years or so. Julian Zelizer, a Princeton history professor, says, you just can’t make very valuable comparisons to presidents from, say, before the Civil War. Isabeth’s father was deported during the George W. Bush administration as part of a policy called Secure Communities. It’s now known as the Illegal Immigration Reform and Immigrant Responsibility Act of 1996. Trump is Teflon when it comes, I think, to white voters, even Latino voters to some extent and Asian American voters. Blue states’ legislatures and local governments are much more willing to pass laws that stop state cooperation with federal immigration enforcement.
2 - Final Summary	President Trump’s rhetoric on immigration and race got people riled up early in his presidential campaign. But the president’s policies and rhetoric are not what they used to be. In this episode, we look back at some of the ways in which President Trump used rhetoric and policies to marginalize people of color in the U.S. and how that rhetoric has been used by the Trump administration. — Subscribe to our weekly newsletter here. Email the show at nprpolitics@npr.org and follow us on Twitter @NPRItsBeenAMin and Facebook @nprpolitics.
3 - Creator Description	In ”Prison City,” Wisconsin, white elected officials are representing voting districts made up mostly of prisoners. Those prisoners are disproportionately black and brown. Oh, and they can’t actually vote.
3 - Baseline Summary	Prisoners are used to shift how political power is distributed. The state uses these numbers to determine election districts. In a lot of cases, this looks like a white elected official representing disproportionately black and Latinx prisoners. In one town, there are three state prisons with a total of more than 3,000 incarcerated people. Each of those districts is represented by a local elected official known as an alderperson. This is according to analysis by the research and advocacy group the Prison Policy Initiative. Waupun is home to the iconic ”End Of The Trail” sculpture, a fitting tribute to the native American Indian. Ryan Mielke, who represents prisoners at Dodge Correctional Institution, told us he has never visited that prison before.
3 - Final Summary	The U.S. census counts prisoners as residents of the town they’re incarcerated in. Not where they lived before they got convicted. The state uses these numbers to determine election districts. It can leave prison towns in an identity crisis. And now some states are trying to change that, in part because prison populations have gotten so big over the decades. NPR’s Shereen Marisol Meraji talks to Augustus Hill from HBO’s ”Oz” about how prisoners, the census and political power are all connected. In participating regions, you’ll also hear a local news segment that will help you make sense of what’s going on in your community. Email the show at samsanders@

Table 2: The creator descriptions, baseline summaries, and final summaries for three randomly selected podcast transcripts from the NPR test set