An End-to-End Workflow using Topic Segmentation and Text Summarisation Methods for Improved Podcast Comprehension

Andrew Aquilina[1], Sean Diacono[1], Panagiotis Papapetrou[1], and Maria Movin[1,2]

[1] Department of Computer and Systems Sciences, Stockholm University, Sweden
{anaq3720,sedi5808}@student.su.se, {panagiotis,maria.movin}@dsv.su.se
[2] Spotify AB, Sweden

Abstract. The consumption of podcast media has been increasing rapidly. Due to the lengthy nature of podcast episodes, users often carefully select which ones to listen to. Although episode descriptions aid users by providing a summary of the entire podcast, they do not provide a topic-by-topic breakdown. This study explores the combined application of topic segmentation and text summarisation methods to investigate how podcast episode comprehension can be improved. We have sampled 10 episodes from Spotify's English-Language Podcast Dataset and employed TextTiling and TextSplit to segment them. Moreover, three text summarisation models, namely T5, BART, and Pegasus, were applied to provide a very short title for each segment. The segmentation part was evaluated using our annotated sample with the P_k and WindowDiff (WD) metrics. A survey was also rolled out ($N = 25$) to assess the quality of the generated summaries. The TextSplit algorithm achieved the lowest mean for both evaluation metrics ($\bar{P}_k = 0.41$ and $\bar{WD} = 0.41$), while the T5 model produced the best summaries, achieving a relevancy score only 8% less to the one achieved by the human-written titles.

Keywords: Topic Segmentation · Text Summarisation · Podcasts

1 Introduction

The consumption of podcast media has been increasing rapidly. In 2020, an estimate of 155 million users were listening to podcasts every week, while the total of monthly podcast US consumers has grown by 16% year-over-year [1]. This has encouraged media service providers, such as Spotify, to reconsider how users could be provided with such content. An example of how this could be achieved is by providing a deeper context of the 'topics' discussed in a podcast episode. In podcasts, a topic generally refers to what is being conveyed in the discourse. More often than not, podcast episodes shift from one topic to another, each revolving around keywords of interest. These are essential in understanding the flow of information, but they are simply not enough for listeners to grasp further context. For example, assume a piece of dialogue explaining the healthy

benefits of fruit. The keyword 'fruit' explains less than the summary of 'why fruits are good for you'. Therefore, summarising segments into short titles revolving around specific topics opens the door for improved and concise comprehension.

One of the challenges pertaining podcasts is their lengthy nature. This incentivises users to carefully select which episode to commit to before listening, ensuring that the provided content would be relevant or interesting to them. In fact, surveys reveal that listeners pay considerable attention to the text description of a podcast before deciding whether to listen to it[3]. Unfortunately, the utility of descriptions falls short if users are looking to consume specific parts of an episode. For example, if one wants to listen to a particular story or discussion, the listener is unable to do so without going over the entire episode. However, manually dividing transcripts may require skimming the full dialog, making it a very laborious and time-consuming task. It may also prove itself as a non-obvious task for human annotators. A widely used method to address such a problem is topic segmentation, which aims to automatically split a single document into shorter, topically coherent segments by an objective algorithm.

To combat such challenge, the newly proposed TREC 2020 Podcast Track [2] encouraged research into two isolated tasks, namely segment retrieval and summarisation. Text summarisation systems focus on generating a short excerpt describing the contents of an entire podcast episode. While these excerpts provide the user with podcasts pertinent to a given search query, they do not deliver a deeper context by labelling the topically-distinctive parts of the episode. As a result, information revealing the beginning or end to a topic is lost. Overcoming this can be achieved through segmented summaries, enabling listeners to not only navigate podcasts with ease, but also improve comprehension, searching capabilities, and information access. All in all, the lack of comparative studies for the combination of both tasks becomes evident. By studying the effects of summarising topically segmented podcasts into short titles and reporting whether this improves episode comprehension contributes to filling the outlined knowledge gap. Providing potential solutions could also shed light on how information access and users' comprehension of podcast content can be improved.

The main aim of this paper is to therefore investigate the combined application of topic segmentation and text summarisation methods to improve podcast episode comprehension, sampling podcast episodes from Spotify's English-Language Podcast Dataset [3]. The structure of podcast data presents unique challenges in the pursuit of such an aim. For example, representing podcast as text can lead to potential inaccuracies caused by automatic speech recognition (ASR) methods. The structure of spoken and written language also vary significantly [4]. We therefore propose a workflow that includes the following contributions: (i) We evaluate the predictive performance of topic segmentation methods when compared to manually annotated segments, (ii) We carry out a preliminary survey using a small sample of episodes and users to determine whether titles generated by text summarisation methods encapsulate the topics of the identified segments, especially when compared to human-written titles.

[3] `https://www.thepodcasthost.com/promotion/podcast-discoverability/`

2 Background & Related Work

2.1 Topic Segmentation

The aim of topic segmentation is to divide blobs of text into semantically coherent segments, either to enhance human interpretation or to facilitate Natural Language Processing applications. As noted from the survey of Purver [5], related work on topic segmentation can be grouped into two categories: either through the specific detection of topic transitions (from the use of cue words or other prosodic features), or by noting changes in the text's vocabulary. Given the scope of this study, we shall focus on the latter, namely on discriminative and clustering-based methods.

Topic Segmentation efforts were founded on the intuition that topical shifts are characterised by vocabulary changes. Therefore, by detecting alterations in lexical use, topic boundaries can be detected. Making use of such an intuition is the TextTiling algorithm [6], one of the most well-known methods within such a domain. The algorithm is prominently used and built upon to this day. An example of this would be TopicTiling [7], which employs Latent Dirichlet Allocation (LDA) and assigns a topic to each word to aggregate topic-count for fixed-size windows. He *et al.* [8] also improve the TextTiling algorithm by introducing a curve-smoothing process, further highlighting the topic changes within segments. The algorithm has also been enhanced through the use of semantic word embeddings, improving on benchmarked approaches [9] .

Instead of detecting points of low cohesion as to note topic changes, clustering-based methods group highly cohesive sentences to reduce the text into a number of topics. The interest in unsupervised techniques is prominenet for clustering-based topic segmentation methods, and the work of Alemi and Ginsparg [10] is no exception, giving birth to Content Vector Segmentation (CVS). By employing GloVe word embeddings, CVS iteratively segments text through the generation of segment scores, splitting them into smaller groups until a threshold is reached. The authors demonstrate state-of-the-art performance. The TextSplit algorithm [11] was recently developed based on the aforementioned work. Nangi *et al.* [12] employed TextSplit to produce topic-specific video segments to index video lectures. Their work was evaluated through a questionnaire-based survey.

According to Jing *et al.* [13], there has been a scarcity of topic segmentation efforts specifically on spoken-word content and podcast data. Their contemporary work focused on identifying the introductory section from a manually-annotated dataset of 400 podcast episodes. The authors trained three Transformer models based on the pre-trained BERT model, providing a basis for introduction segmentation on podcast data. Earlier efforts by Fuller *et al.* [14] aimed to enable users skimming podcast episodes. The authors found that the TextTiling algorithm agreed quite well with the human-made segments. Their work was conducted on a corpus of 30 podcast episodes.

2.2 Text Summarisation

Text Summarisation is the process of taking a piece of text, selecting the most important information, and creating an abridged version. As manual text summarisation is arduous and time consuming, automatic summarisation systems have been developed. Such approaches can be classified into two main groups: extractive methods, where important sentences are directly taken from the source text and joined together to form a summary, and abstractive methods, where new sentences are generated to summarise the salient parts. For the scope of this study, we have focused on abstractive summarisation systems using deep-learning based methods.

Deep-learning based methods use Recurrent Neural Network (RNN) Sequence-to-Sequence (Seq2Seq) models to generate summaries [15]. One such architecture is BART [16], achieving state-of-the-art results in text generation tasks. Lewis *et al.* [16] fine-tuned BART on summarisation datasets and achieved Recall-Oriented Understudy for Gisting Evaluation 1 (ROGUE-1) scores of 44.16 on the CNN/DailyMail (CNN/DM) dataset [17]. Alternatively, Pegasus [18] is a model with a pre-training task aimed at abstractive text summarisation. Fine-tuned for text summarisation, Pegasus attained a ROGUE-1 score of 37.68 on the Annotated Enron Subject Line Corpus (AESLC) dataset [19]. Another deep-learning model used for text summarisation is the Text-To-Text Transfer Transformer (T5) model [20]. Fine-tuned on the CNN/DM dataset, the T5 architecture achieved a ROGUE-1 score of 43.52.

Several techniques have been applied to podcast data to explore their effectiveness. Rezapour *et al.* [21] used BART for abstractive summarisation and achieved an F-measure score of 18.42% compared to human-written descriptions. The models were pre-trained on the CNN/DM dataset and then fine-tuned on Spotify's English-Language Podcast Dataset. Their system outperformed humans according to a survey. Karlbom and Clifton [22] combined BART with Longformer Attention and achieved a higher F-measure score of 19.23%. Although these abstractive summarisation systems generate coherent summaries, they are usually still made up of multiple sentences, making them too lengthy to address our research problem.

Research has been done to apply text summarisation techniques for headline and title generation. Such approaches are able to create summaries made up of just a few words for a piece of text, making them ideal for tasks such as news headline generation and scientific paper title generation. The Gigaword dataset has enabled the development of such short summarisation systems by being made up of news articles and their respective headlines [23]. Aghajanyan *et al.* [24] proposed a language generation model based on BART. Their research focused on fine-tuning the model using different language generation datasets. The model achieved a ROGUE-1 score of 40.40 on the Gigaword dataset, surpassing the previous Pegasus model's score of 39.12. These models, although untested on podcast data, are promising candidates for our study's second objective.

2.3 Segmentation-based Summarisation

There has been prior work combining topic segmentation and text summarisation efforts. Cho *et al.* [25] investigated the utility of segmentation methods for extractive summarisation of lengthy scientific articles. Their approach learns representations using a Longformer model to conduct segmentation and summarisation simultaneously. Similarly, Liu *et al.* [26] build two text segmentation models to also improve the extractive summarisation task, noting that a segmentation-based approach enhances summarisation quality especially when information is not found at the beginning of a document. Their text segmentation model is based on a modified version of TextTiling using BERT embeddings. Experimenting with both extractive and abstractive summarisation systems, Miculicich and Han [27] propose two frameworks to summarise news articles. The first is the *Pipeline* framework, which first segments the text using a segmentation model, and then applies a title generation model to generate the headings. The other is the *Joint* approach, which tackles both tasks using a single encoder-decoder neural model. The authors note the beneficial nature of the latter as it allows the previously generated headings to be known when generating the concurrent title. While this may be advantageous for news articles due to their high level of topical cohesion, this may not be the case for podcast data.

3 Methodology

3.1 Problem Formulation

Let $\mathcal{P} = \{P_1, \ldots, P_n\}$ define a set of n podcasts, where each podcast is described by a transcribed piece of text, which we refer to as the *podcast transcript*. Moreover, for each podcast $P_i \in \mathcal{P}$, we can find a set of k_i indices $\mathcal{J}_i = \{\beta_1, \ldots, \beta_{k_i}\}$ that in turn defines a set of k_i segments $f = \{P_i[1 : \beta_1], P_i[\beta_1 + 1 : \beta_2], \ldots, P_i[\beta_{k_i-1} + 1 : \beta_{k_i}]\}$, with $\beta_{k_i} = |P_i|$, i.e., the last segment ends at the end of each podcast. Each podcast segment can also be described by some text summary, which encapsulates the main discussion point of said segment. In addition, for each podcast $P_i \in \mathcal{P}$, there exists $\mathcal{F}_i = \{f_1, \ldots, f_{m_i}\}$, which defines a set of m possible topic segmentations, and $\mathcal{G}_i = \{g_1, ..., g_{l_i}\}$ which defines a set of l possible text summaries given these segments.

For each podcast $P_i \in \mathcal{P}$, we want to find a set of segments f^* from $\mathcal{F}_i$ and a set of summaries g^* from $\mathcal{G}_i$, such that f^* yields the minimum error for topic segmentation and g^* yields the maximum score for average summary relevancy. Let $\mathcal{E}_1$ and $\mathcal{E}_2$ be defined as evaluation functions for f and g respectively[4]. The problem is denoted in Equation 1.

$$\forall P_i \in \mathcal{P}, \text{ such that } f^* = \underset{f \in \mathcal{F}_i}{\operatorname{argmin}} \mathcal{E}_1(f) \text{ and } g^* = \underset{g \in \mathcal{G}_i}{\operatorname{argmax}} \mathcal{E}_2(g) \tag{1}$$

[4] Instantiations for both functions can be found and further explained in Sections 4.2 and 4.3.

3.2 Workflow overview

The main steps of the proposed workflow addressing our research problem are depicted in Figure 1. We have taken cues from the *Pipeline* approach, as outlined by Miculicich and Han [27], to develop our methodology. Our workflow mostly consists of two steps: (1) topic segmentation, and (2) text summarisation. Sampled podcast transcripts are segmented using the topic segmentation algorithms, namely TextTiling and TextSplit, followed by an evaluation using human-made segments to report how well the podcasts' topics are captured. The segments stemming from the best performing topic segmenter are then summarised using the text summarisation models, namely T5, BART, and Pegasus, to provide a title to each segment. A survey is then deployed to a number of human participants to evaluate the relevancy of the generated summaries. To our knowledge, we are the first to utilise and provide a comparative evaluation of the aforementioned topic segmentation algorithms in conjuction with such text summarisation methods to improve podcast episode comprehension.

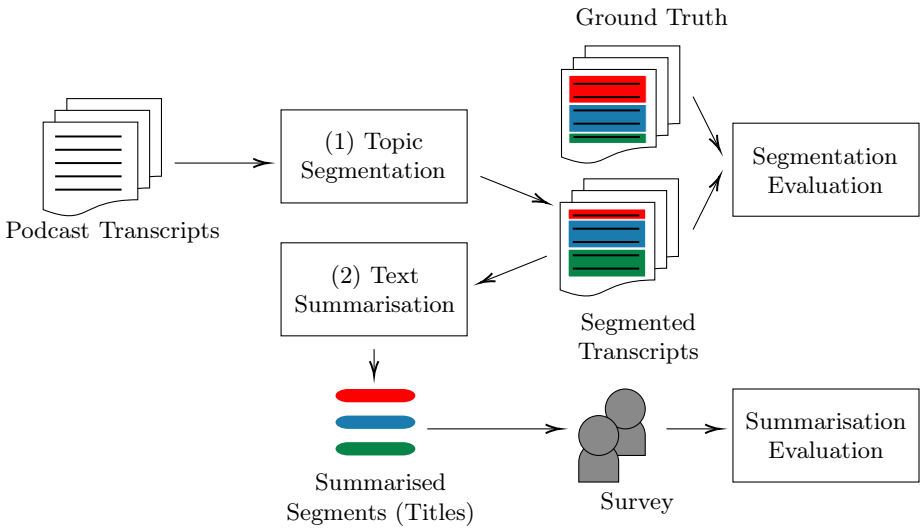

Fig. 1. An overview of our proposed workflow.

3.3 Topic Segmentation

Building on the majority of prior research [28, 29], we utilised the implementation of the TextTiling algorithm as provided by the `NLTK` module. The TextTiling algorithm [6] divides the given text into equally sized windows and represents each with a lexical frequency vector. Cosine similarities of adjacent windows

are then calculated to hypothesise topic boundaries. The algorithm's parameters consist of the pseudosentence length w, block size k, as well as the chosen approach to threshold the depth score plot f. While the author of the algorithm suggests optimal values for such parameters [6], namely $w = 20$ and $k = 10$, we conduct a grid-search to tune the algorithm's parameters. Our intention is to explore the parameter space, tuning the discussed parameters within the context of the podcast transcripts.

We also compare the TextTiling algorithm with TextSplit, an implementation based on the clustering-based segmentation technique of Alemi and Ginsparg [10]. By representing sentences in vector space through the employment of word embeddings, sentences are clustered together to form a number of topics. The algorithm terminates when the number of topics reaches the predefined penalty threshold p or when the reached gain from performing a split is below the required amount. There exist two variants of this algorithm, greedy or dynamic. The penalty hyper-parameter p defines the granularity of the split. While the greedy approach iteratively splits the text until it is below the given penalty threshold, the dynamic alternative finds the optimal p value given a maximum number of sentences that belong to a segment. We conduct a tuning experiment to note which segment-length l value yields the best p value.

3.4 Text Summarisation

We made use of several text summarisation models to give short titles to each segment extracted from the podcast transcripts. Through a Python package, the HuggingFace Transformer's open-source library makes available several summarisation models, such as T5, BART, and Pegasus. This open-source library allows for inference with models fine-tuned on text summarisation datasets, hence why it was used in our research. The T5 architecture was implemented due to its flexibility and potential for good performance on our text summarisation task. The T5 model chosen from the HuggingFace library was fine-tuned on a WikiNews dataset made up of 500 thousand articles and their headlines. Based on preliminary qualitative testing using the HuggingFace interface, this model was found to produce appropriate titles for podcast segments. For this reason, we included this model in our research for further evaluation. The second model incorporated in this study is a Pegasus model fine-tuned on the AESLC dataset. The AESLC dataset is made up of email bodies and their respective subject lines [19], making it ideal for the text summarisation task. As discussed in Section 2, the Pegasus architecture achieved state-of-the-art results on this dataset [18] and it is included in the HuggingFace library, hence why it has been included. The BART architecture is the final text summarisation technique investigated in this research. The BART model used from the HuggingFace library was fine-tuned on the X-Sum dataset [30]. This dataset is comprised of news articles and summaries made up of just one sentence. Therefore, this version of BART generates extracts that also fit the requirements of our text summarisation task. We include Table 2 in the Appendix to present a collection of generated titles as examples.

4 Evaluation

4.1 Dataset

The dataset used in this study is the English-Language Podcast Dataset from Spotify [3] that includes both audio and text data. The corpus comprises of 100,000 podcast episodes varying in production quality, topics, and structural formats. For each episode, a transcript captured through the means of ASR is provided. For the scope of this study, we have randomly sampled a subset of 10 episodes and focused solely on such transcripts. Metadata, such as the show's name and publisher's area, are also included for each episode, but topical segments are not. As we require such segmentations to answer our research question, the sampled dataset has been annotated with segment boundaries. This was done by listening to the episodes themselves and marking down topic shifts.

4.2 Topic Segmentation

Adopted Evaluation As described by Purver [5], the P_k and WD metrics were used to evaluate the algorithms' performance when compared to the human reference points. P_k is calculated as follows. Let $\delta_S(i,j)$ indicate whether sentences i and j are in distinct segments, evaluating to 1 if so and 0 in the opposite case. The type S refers to whether the segmentation is derived from reference R or hypothesis H. Therefore $\delta_H(i,j) \oplus \delta_R(i,j)$ denotes whether H and R disagree about the separation of i and j, where $\oplus$ is the XOR operator. P_k is obtained by moving a sliding window across the entire segments, aggregating this score, and dividing by the number of windows $N-k$, shown in Equation 2.

$$P_k = \frac{\sum_{i=1}^{N-k} \delta_H(i,i+k) \oplus \delta_R(i,i+k)}{(N-k)} \tag{2}$$

The WD metric is calculated in a similar fashion, shown in Equation 3. While a fixed-width window is also moved across the data, windows are classified as "correct" if and only if the same number of boundaries are assigned between their start and end. This is defined with the variable $b_S(i,j)$.

$$WD = \frac{\sum_{i=1}^{N-k} [|b_H(i,i+k) - b_R(i,i+k)| > 0]}{(N-k)} \tag{3}$$

We also evaluate their performance to a random baseline, starting a new segment with probability $\frac{1}{k}$, where k is the number of average segments in the annotated set. For this, we used the `SegEval` Python package [31]. The best segmentation technique according to these metrics was utilised within the second objective.

Results Based on the smallest produced P_k and WD, we tune the parameters of the TextTiling and TextSplit algorithms using grid and linear search respectively. The optimal parameters were found to be $w = 30$, $k = 5$, and $f = 0$ for the TextTiling algorithm and $l = 10$ for the TextSplit algorithm. The effect of the utilised algorithms is illustrated in Figure 2, depicting their improvement over the baseline in terms of the P_k and WD error metrics. As it can be observed from the aforementioned figure, we conclude that the TextSplit algorithm (with $l = 10$) achieved the lowest mean for both error metrics: $\bar{P}_k = 0.41$ and $\bar{WD} = 0.41$. We therefore employed its use for the second objective.

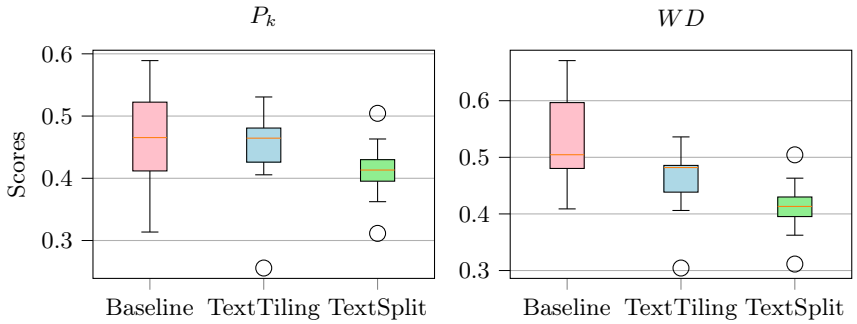

Fig. 2. The P_k and WD scores achieved by the various segmenters, compared with the baseline. The shown results are average values after 10 iterations.

4.3 Text Summarisation

Adopted Evaluation Due to the absence of ground truth data, evaluation was carried out through the use of a preliminary survey [21, 22], whereby participants were questioned on the summary's relevancy with respect to its corresponding segment. The survey comprised of a small set of 10 randomly sampled podcast segments (one from each episode) extracted by the best segmentation algorithm (TextSplit, in this case). The three summaries generated by T5, BART, and Pegasus, alongside a human-written title[5], were included for each segment. A human-written title was also included. After listening to the audio segment, the respondents were requested to score each title (on a 5-Point Liktert Scale) based on the relevancy of the summary. Given N participants, we define the average relevancy score R_x as outlined in Equation 4.

$$R_x = \frac{\sum_{i=1}^{\tilde{\mathcal{S}}_x} \text{score}(S_i)}{|\tilde{\mathcal{S}}_x| \cdot N} \tag{4}$$

[5] The human-written titles underwent scrutiny to ensure they accurately represented the content of the respective segment in a comprehensive manner.

This allows the quantitative comparison of the generated summaries' quality and is calculated with respect to some text summarisation model x. The total score obtained on some summary S_i is given by score(S_i). The set of the sampled summaries generated by model x is denoted by $\tilde{\mathcal{S}}_x$. A model that achieves a relevancy score equal to or higher than that of human-written titles is considered to be a favorable outcome.

Results For each of the 10 sampled segments, we generate a summary using the considered text summarisation models and surveyed 25 human participants to assess their relevancy. The results from the survey revealed that the T5 model produced the best summaries ($R_{\text{T5}} = 3.34$), achieving a relevancy score closest to that of the human-written summaries ($R_{\text{Human}} = 3.64$). On the contrary, BART, and Pegasus ($R_{\text{BART}} = 2.60$ and $R_{\text{Pegasus}} = 2.30$ respectively) performed the worst due to the scores of the irrelevant summaries they sometimes produced. We analyse the effect of variables towards the relevancy scores in Table 1. For example, we investigate the initial intuition that an episode with high segmentation error consists of segments with poor topic coverage, and therefore, highly irrelevant summaries. From such an analysis, we note that no significant correlation between the quality of the episodes' segmentation and the summaries' relevancy was found. While the addition of more samples may be necessary to substantiate this finding, we conjecture that a segment's summary can still be considered relevant even if the segment does not adhere to a segmentation baseline. Additionally, segment length was positively correlated with higher relevancy for Pegasus, but not for the other models. While investigating this may fall outside the scope of this study, it is worth noting that the Pegasus model underwent fine-tuning using the AESLC corpus, which, in comparison to the datasets employed for fine-tuning BART and T5, exhibits the smallest average document size [19, 32]. Interestingly, longer titles generated by T5 was also attributed to a higher relevancy score for the BART model, and shorter titles generated by BART attributed to a higher relevancy score for the T5 model.

	Relevancy			
	Human	T5	BART	Pegasus
Segment Length	.304	-.032	.242	.708*
Human Summary Length	.745*	.049	-.07	.016
T5 Summary Length	-.072	-.388	.738*	.355
BART Summary Length	.27	-.668*	.02	-.135
Pegasus Summary Length	-.224	.466	-.251	-0.08
Episode P_k	-.153	.205	.360	.026
Episode WD	-.1	.263	.288	.102

Table 1. Pearson's correlation coefficients between the relevancy scores and the outlined variables (* = significant at the $p < .05$ level).

5 Conclusion

In this research, we have explored the combined application of topic segmentation and text summarisation methods to investigate how podcast episode comprehension can be improved. Using the segments generated by the best performing topic segmenter (TextSplit with $\bar{P_k} = 0.41$ and $\bar{WD} = 0.41$), we employed the considered text summarisation models (T5, BART, and Pegasus) to produce respective summaries. A survey was rolled out to 25 human participants to assess the relevancy of the generated summaries. From the reported results, we deem the T5 model as the most promising text summarisation model out of the three. Having an average relevancy score of 3.34, the T5 model was off by 8% when compared to the human-written titles. In conclusion, by investigating the effectiveness and efficiency of topic segmentation alongside text summarisation techniques, we show that such combination can indeed improve podcast episode comprehension. A limitation of our approach is the small samples taken to evaluate the second objective. Future work may expand the total number of surveyed users and sampled episodes. For example, sampling multiple segments within an episode may provide further insights. Additionally, other algorithms that were not explored in our experiments can also be considered. The code used in this study is freely available on GitHub [33].

References

[1] Edison Research and Triton Digital, *The infinite dial 2020*, 2020.
[2] R. Jones, B. Carterette, A. Clifton, M. Eskevich, G. J. F. Jones, J. Karlgren, A. Pappu, S. Reddy, and Y. Yu, "Trec 2020 podcasts track overview," 2021.
[3] A. Clifton, S. Reddy, Y. Yu, A. Pappu, R. Rezapour, H. Bonab, M. Eskevich, G. Jones, J. Karlgren, B. Carterette, and R. Jones, "100,000 podcasts: A spoken English document corpus," in *Proceedings of the 28th International Conference on Computational Linguistics*, Barcelona, Spain (Online): International Committee on Computational Linguistics, Dec. 2020, pp. 5903–5917.
[4] R. Power, D. Scott, and N. Bouayad-Agha, "Document structure," *Computational Linguistics*, vol. 29, no. 2, pp. 211–260, 2003.
[5] M. Purver, "Topic segmentation," *Spoken language understanding: systems for extracting semantic information from speech*, pp. 291–317, 2011.
[6] M. A. Hearst, "Text tiling: Segmenting text into multi-paragraph subtopic passages," *Computational linguistics*, vol. 23, no. 1, pp. 33–64, 1997.
[7] M. Riedl and C. Biemann, "Topictiling: A text segmentation algorithm based on lda," in *Proceedings of ACL 2012 Student Research Workshop*, 2012, pp. 37–42.
[8] X. He, J. Wang, Q. Zhang, and X. Ju, "Improvement of text segmentation texttiling algorithm," in *Journal of Physics: Conference Series*, IOP Publishing, vol. 1453, 2020, p. 012 008.

[9] V. Gupta, G. Zhu, A. Yu, and D. E. Brown, "A comparative study of the performance of unsupervised text segmentation techniques on dialogue transcripts," in *2020 Systems and Information Engineering Design Symposium (SIEDS)*, IEEE, 2020, pp. 1–6.

[10] A. A. Alemi and P. Ginsparg, "Text segmentation based on semantic word embeddings," *arXiv preprint arXiv:1503.05543*, 2015.

[11] C. Schock, *textsplit*, May 2020. [Online]. Available: `https://github.com/chschock/textsplit`.

[12] S. R. Nangi, Y. Kanchugantla, P. G. Rayapati, and P. K. Bhowmik, "Offvid: A system for linking off-topic concepts to topically relevant video lecture segments," in *2019 IEEE 19th International Conference on Advanced Learning Technologies (ICALT)*, IEEE, vol. 2161, 2019, pp. 37–41.

[13] E. Jing, K. Schneck, D. Egan, and S. A. Waterman, "Identifying introductions in podcast episodes from automatically generated transcripts," *arXiv preprint arXiv:2110.07096*, 2021.

[14] M. Fuller, M. Tsagkias, E. Newman, J. Besser, M. Larson, G. J. Jones, and M. de Rijke, "Using term clouds to represent segment-level semantic content of podcasts," *CIP GEGEVENS KONINKLIJKE BIBLIOTHEEK, DEN HAAG*, p. 12, 2008.

[15] S. Gupta and S. K. Gupta, "Abstractive summarization: An overview of the state of the art," en, *Expert Systems with Applications*, vol. 121, pp. 49–65, May 2019.

[16] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 7871–7880.

[17] R. Nallapati, B. Zhou, C. Gulcehre, B. Xiang, *et al.*, "Abstractive text summarization using sequence-to-sequence rnns and beyond," *arXiv preprint arXiv:1602.06023*, 2016.

[18] J. Zhang, Y. Zhao, M. Saleh, and P. J. Liu, "Pegasus: Pre-training with extracted gap-sentences for abstractive summarization," in *Proceedings of the 37th International Conference on Machine Learning*, ser. ICML'20, JMLR.org, 2020.

[19] R. Zhang and J. Tetreault, "This Email Could Save Your Life: Introducing the Task of Email Subject Line Generation," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 446–456.

[20] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020.

[21] R. Rezapour, S. Reddy, A. Clifton, and R. Jones, "Spotify at TREC 2020: Genre-aware abstractive podcast summarization," in *Proceedings of the Twenty-Ninth Text REtrieval Conference, TREC 2020, Virtual Event*

[Gaithersburg, Maryland, USA], November 16-20, 2020, E. M. Voorhees and A. Ellis, Eds., ser. NIST Special Publication, vol. 1266, National Institute of Standards and Technology (NIST), 2020.

[22] H. Karlbom and A. Clifton, "Abstract podcast summarization using BART with longformer attention," in *Proceedings of the Twenty-Ninth Text REtrieval Conference, TREC 2020, Virtual Event [Gaithersburg, Maryland, USA], November 16-20, 2020*, E. M. Voorhees and A. Ellis, Eds., ser. NIST Special Publication, vol. 1266, National Institute of Standards and Technology (NIST), 2020.

[23] D. Graff and C. Cieri, *English Gigaword*, Jan. 2003.

[24] A. Aghajanyan, A. Gupta, A. Shrivastava, X. Chen, L. Zettlemoyer, and S. Gupta, "Muppet: Massive Multi-task Representations with Pre-Finetuning," *arXiv:2101.11038 [cs]*, Jan. 2021.

[25] S. Cho, K. Song, X. Wang, F. Liu, and D. Yu, "Toward unifying text segmentation and long document summarization," *arXiv preprint arXiv:2210.16422*, 2022.

[26] Y. Liu, C. Zhu, and M. Zeng, "End-to-end segmentation-based news summarization," *arXiv preprint arXiv:2110.07850*, 2021.

[27] L. Miculicich and B. Han, "Document summarization with text segmentation," *arXiv preprint arXiv:2301.08817*, 2023.

[28] Y. Song, L. Mou, R. Yan, L. Yi, Z. Zhu, X. Hu, and M. Zhang, "Dialogue session segmentation by embedding-enhanced texttiling," *arXiv preprint arXiv:1610.03955*, 2016.

[29] Y. Xu, H. Zhao, and Z. Zhang, "Topic-aware multi-turn dialogue modeling," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, 2021, pp. 14 176–14 184.

[30] S. Narayan, S. B. Cohen, and M. Lapata, "Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium: Association for Computational Linguistics, Oct. 2018, pp. 1797–1807.

[31] H. Nakayama, *Hiroki Nakayama*, 2020. [Online]. Available: `https://github.com/chakki-works/seqeval`

[32] R. Calizzano, M. Ostendorff, Q. Ruan, and G. Rehm, "Generating extended and multilingual summaries with pre-trained transformers," in *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, 2022, pp. 1640–1650.

[33] S. Diacono and A. Aquilina, *Spotify Text Segmentation and Summarisation*, May 2022. [Online]. Available: `https://github.com/seandiacono/Spotify-Topic-Segmentation`

Appendix A Generated Summaries

Segments	**Titles**			
	Human	T5	BART	Pegasus
Segment 1	Eat something before an argument	Getting Into An Argument - Eat Something	Had to work really well for us is if we start to get heated walk away	The first question one of us would ask is when was the last time you ate
Segment 2	Farmer eating dirty milk	A Cow Drop Some Milk Into A Field	I think I know what happened to a cow's milk	Milk
Segment 3	Igor's Theme	I Got My Eyes Open - I Got My Eyes Open	Igor is Tyler, the Creator's first album in four years, is a story about the Alps and the people who live there	Album Review
Segment 4	Target Audience of Human Library	Human Library - Is There a Target Audience?	Criterion Library, an open-air library in the Indian city of Bangalore, has been talking to the BBC's Geeta Pandey about its work and its target audience.	Human Library
Segment 5	Hating soda and candy	What Are Some Things You Loved As a Kid and Now Not So Much Or Hate?	What are some things you loved as a kid that you now hate?	Soda

Table 2: Example list of generated summaries.