Record Once, Post Everywhere: Automatic Shortening of Audio Stories for Social Media

Bryan Wang
University of Toronto
Toronto, ON, Canada
bryanw@dgp.toronto.edu

Zeyu Jin
Adobe Research
San Francisco, CA, USA
zejin@adobe.com

Gautham J. Mysore
Adobe Research
San Francisco, CA, USA
gmysore@adobe.com

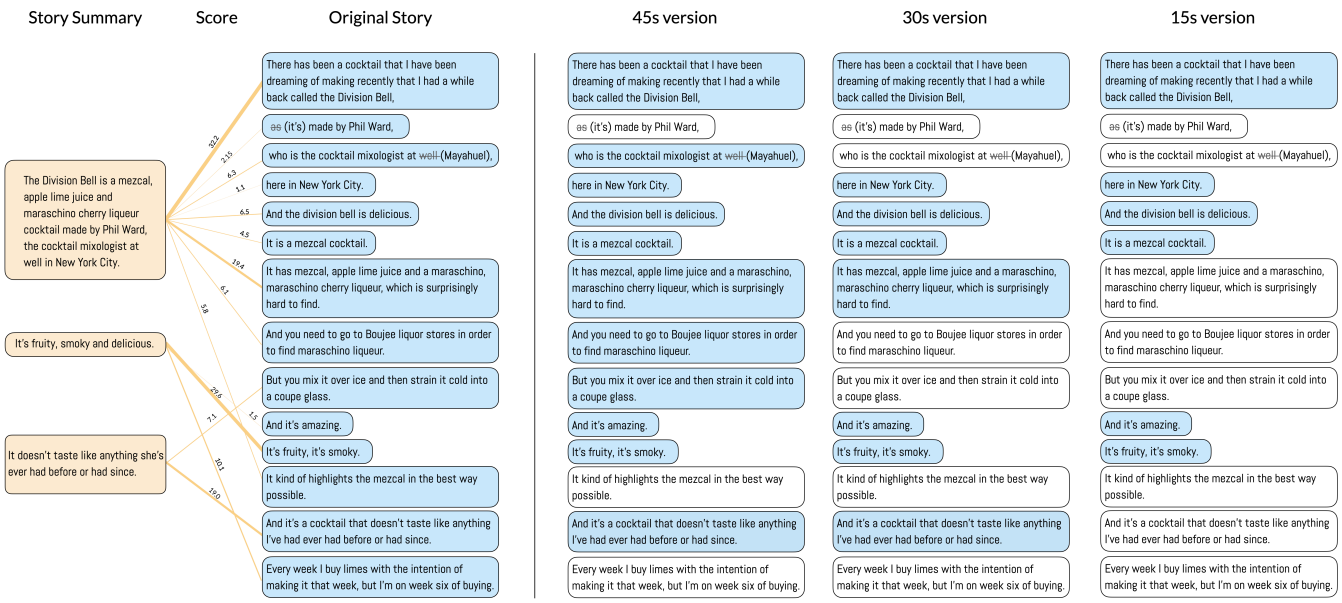

Figure 1: Users record audio stories once, and ROPE will automatically shorten them to create the 45 seconds, 30 seconds, and 15 seconds versions. ROPE can shorten audio to other lengths as well. It does so by selecting subsets of sentences from the original story. It first performs speech-to-text and text summarization to capture topics in the audio story. It then calculates each sentence's score, indicating how relevant it is to these topics. Finally, it performs a combinatorial optimization to select an optimal sentence combination that maximizes the total sentence score while complying with the length limit.

ABSTRACT

Following the prevalence of short-form video, short-form voice content has emerged on social media platforms like Twitter and Facebook. A challenge that creators face is hard constraints on the content length. If the initial recording is not short enough, they need to re-record or edit their content. Both are time-consuming, and the latter, if supported, can have a learning curve. Moreover, creators need to manually create multiple versions to publish content on platforms with different length constraints. To simplify this process, we present ROPE[1] (Record Once, Post Everywhere). Creators can record voice content once, and our system will automatically shorten it to all length limits by removing parts of the recording for each target. We formulate this as a combinatorial optimization problem and propose a novel algorithm that automatically selects optimal sentence combinations from the original content to comply with each length constraint. Creators can customize the algorithmically shortened content by specifying sentences to include or exclude. Our system can also use the user-specified constraints to recompute and provides a new version. We conducted a user study comparing ROPE with a sentence-based manual editing baseline. The results show that ROPE can generate high-quality edits, alleviating the cognitive loads of creators for shortening content. While our system and user study address short-form voice content specifically, we believe that the same concept can also be applied to other media such as video with narration and dialog.

UIST '22, October 29-November 2, 2022, Bend, OR, USA

ACM ISBN 978-1-4503-9320-1/22/10…$15.00
https://doi.org/10.1145/3526113.3545680

[1]Project page available at https://www.dgp.toronto.edu/~bryanw/rope

CCS CONCEPTS

• **Human-centered computing → Human computer interaction (HCI)**.

KEYWORDS

Audio editing, storytelling, social media, short-form audio

ACM Reference Format:
Bryan Wang, Zeyu Jin, and Gautham J. Mysore. 2022. Record Once, Post Everywhere: Automatic Shortening of Audio Stories for Social Media. In *The 35th Annual ACM Symposium on User Interface Software and Technology (UIST '22), October 29-November 2, 2022, Bend, OR, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3526113.3545680

1 INTRODUCTION

Audio-based social platforms have gained tremendous growth since the beginning of 2021. In February 2021, Clubhouse's [4] global downloads increased from over 3.5 million to 8.1 million within two weeks and accumulated up to more than 28 million at the end of the year [2]. Following Clubhouse's success, tech giants such as Twitter, Facebook, and Spotify had also joined the market by launching their versions of live social audio platforms [3]. In addition to live-streamed audio, platforms that allow users to record and post voice content have also emerged. Facebook Soundbites [5] enables creators to create and share bite-sized, short-form audio stories and conversations. Cappuccino [3] allows users to record short audio messages to share their thoughts with friends. While still in its infancy, short-form audio platforms are showing signs that social audio could follow a similar trend witnessed for videos on social media–they are becoming shorter [9].

Short-form content tends to be heavily-edited, concise media that effectively engages the audience. It offers rich information while requiring only seconds of a person's attention. Many platforms such as TikTok and Instagram (video), and Twitter (text) even deliberately impose hard constraints on length to encourage creators to curate content in concise formats. However, ensuring that content complies with length constraints can be time-consuming and sometimes requires expertise in content editing. Moreover, length limits can vary across platforms, and creators would therefore need to manually create multiple versions of the content if they wish to publish on various platforms. These difficulties can be challenging for creators with limited editing experience.

This paper focuses on the automatic shortening of audio stories to lower the barrier to short-form audio content creation. We present ROPE (Record Once, Post Everywhere), an automatic system that shortens a raw voice recording to any target length and generates high-quality audio output. ROPE contributes an editing paradigm that *understands* the audio story and semantically edits it, as opposed to existing methods that require users to edit raw waveforms [1, 2, 8] or text [11, 12, 16] without automatically reasoning about the overall story. To inform the design of our system, we first collected and analyzed recording samples of short audio stories from Mechanical Turk. We then conducted a listening study for audio time-stretching, an audio shortening technique commonly used in video/audio editing. The results indicate that speeding up speech audio would inevitably sacrifice its intelligibility and naturalness, and the subjective rating monotonically decreases as the speed-up factor increases.

To address the limitations of existing shortening techniques, we present an algorithm that can shorten a voice recording to fit any length limit. We formulate automatic shortening as a combinatorial optimization problem to select optimal sentence combinations complying with length constraints. Our algorithm first transcribes the recording and segments it into sentences. It then selects optimal sentence subsets from the original recording by maximizing the total sentence score given the length constraints. We designed the sentence score function to consider both a sentence's duration and its relevance to the summary of the audio story, obtained with neural abstractive summarization, in the sentence embedding space. We use dynamic programming for efficient optimization, which runs in real-time. Once the optimal selection is obtained, ROPE synthesizes the final audio output by cropping and concatenating the selected sentences. We also apply an audio enhancement technology to increase sound quality. A key challenge of automatic shortening of audio stories is their subjective nature, i.e., there can be more than one "good story" that fits the length limit. Therefore, we further investigate user-in-the-loop approaches to refine ROPE's sentence selections based on user preferences. We conducted a user study comparing ROPE with a manual editing baseline to solicit user feedback. The results showed that using ROPE achieves higher outcome satisfaction while requiring lower cognitive loads for creators compared to fully manual editing. To the best of our knowledge, this work presents the first investigation on the length-constrained shortening of already short-form audio stories. In summary, our paper makes the following contributions:

- Data collection of short-form audio stories from Mechanical Turk and an analysis of the emerging media.
- A listening study that demonstrates the capabilities and limitations of audio time-stretching, a shortening technique commonly used in social media.
- The formulation of audio story shortening as a combinatorial optimization problem and a novel algorithm that can shorten audio to comply with any length limit.
- A full-stack system allows users to interactively refine the algorithmic output, which was shown to be effective for creating shortened audio content.

2 RELATED WORK

ROPE automatically shortens an audio story leveraging text summarization techniques. In this section, we review the literature in 1) automatic summarization for speech, and 2) speech editing tools.

2.1 Speech Summarization

Automatic summarization produces an condensed and informative version of its input sources and has been explored on various modalities such video [33, 46], audio [17, 30], and text [31, 36, 45]. Speech summarization [27, 41, 48] often summarizes speech content by transcribing them into text and leverages text summarization techniques that can capture the underlying semantics. Text summarization can be broadly categorized as extractive summarization

[2] https://www.highfidelity.com/blog/most-popular-social-audio-apps
[3] https://citrusbits.com/the-rise-of-social-audio-facebook-twitter-spotify-reddit-to-add-clubhouse-like-features

[13, 20, 45, 47, 49] and abstractive summarization [18, 21, 28, 37]. The former summarizes texts by extracting essential information sentences; the latter "re-writes" the summary, i.e., generate summaries that contain new phrases and sentences that may not appear in the source text. Our work also leverages text summarization to shorten audio stories. However, we did not directly apply existing extractive summarization models as they are usually trained to summarize longer-form content such as news articles [35] and do not consider the length of texts in audio. Instead, we build upon abstractive summarization and propose a novel speech shortening algorithm that considers both the language semantics and the length of text content in audio.

2.2 Speech Editing Tools

Speech content is traditionally edited using a waveform-based editor such Audition [2], Audacity [1], and Logic Pro [8]. However, navigating and editing raw speech recordings using a waveform editor can be difficult as it does not present the language content within the audio. Therefore, there has been a body of work investigating text-based editing of speech [11, 12, 16, 22, 40, 42]. Text-based editing leverages time-aligned transcriptions, enabling editors to focus on the text content while editing. While effective, text-based editing tools typically do not provide content-level edit suggestions. As a result, users still have to decide which content to edit manually. ROPE differs from prior work in that our algorithm can automatically generate edit suggestions using our natural language processing pipeline. ROPE also provides sentence-based manual selection tools for users to refine the algorithm output.

3 FORMATIVE STUDIES

We conducted formative studies to characterize the content people would record for short-form audio stories and to understand the limitation of audio time-stretching, a commonly used shortening technique for audio/video content on social media platforms. The findings motivated the design of our system's shortening pipeline.

3.1 Study 1: What are short-form audio stories?

Despite the abundance of edited short-form audio content on the internet, original unedited recordings are typically unavailable. To kickstart research in automatic editing and retargeting of short-form audio content, we collected a dataset of unedited audio story recordings on Amazon Mechanical Turk using a web app hosted on Google Cloud Platform.

3.1.1 Study Procedure. Firstly, we explained the task and solicited participants' consent to use their audio samples for research. We ensured the recordings were disassociated with their worker ID so that the speaker could not be identified. Then, we provided an example audio story one of the authors recorded to help participants understand the type of content we wished to collect. We then asked the participant to test their microphone and English fluency by reading out a sentence presented on the screen. Our system transcribed their speech in real-time and checked if it matched the assigned sentence. After that, the main task starts. We first showed a text prompt from a corpus of six prompts that encouraged the participant to talk about their personal experience on sufficiently general topics. These topics include describing a vacation, a project they worked on, a food/drink they like, a song they like, a place they would like to live in, or a recent event for which they feel grateful. We instructed the participants to record at least one minute and not to record personal information or excessive profanity in the recordings. Participants could switch to the last topic in the corpus if they found recording stories for the assigned topic challenging. We provide a text box for the participant to optionally sketch out their story. Once the participant clicked on the "start" button, the recording began. The participants could not stop the recording until at least 1 minute of audio was recorded. Once the recording was done, they could review it and decide if they wanted to re-record it. If they were satisfied with the recording and clicked to submit it, the website sent the audio to a back-end server hosted on a Google Cloud virtual machine anonymized with the worker ID.

3.1.2 Findings. We collected 35 audio recordings (25 male, 10 female). We discarded one audio clip in which the participant only recorded silences. The average length of the remaining recordings is 73.1 seconds (Std=14.1), and the average length of each audio story is 160.8 words (Std=64.7). The average word per minute in the story is 129.7 (Std=42.1). We observed a common topic-based structure of the audio stories we collected, as shown in figure 2 with sentences with different functionalities colored differently.

An audio story usually consists of the main topic, multiple relevant sub-topics, and an ending:

- **Main Topic**: The exposition, or the hook. The story's first sentence typically establishes the theme/context by reiterating or answering the prompts (teal).
- **Sub-topics**: Following the exposition, there may be several sub-topics that are relevant to the main theme. In addition to the topic sentence (blue), supporting sentences (yellow) were sometimes used to provide further information.
- **Ending**: To conclude the audio story, the speakers usually echo the exposition to provide a high-level summary of the whole story (teal). Some may end the story in more creative ways, such as making a joke or posing an intriguing question.

Based on the common structure we observed, we aim to design an algorithm that captures the underlying topics of short audio stories and shortens them by removing topics and their supporting sentences.

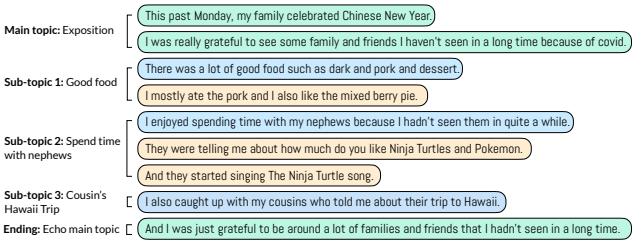

Figure 2: Example story from our data collection study. A story typically consists of the main topic (exposition), several related sub-topics, and an ending that echos the exposition. Sentences colored in teal are relevant to the exposition, while the blue ones are the topic sentences, and the yellow ones are sentences that support an established topic.

3.2 Study 2: Voice naturalness vs. speed-up rate

Audio time stretching alters an audio signal's playback speed and duration. It is commonly used for video and audio editing when editors want to conform longer material to a designated time slot [6]. Resampling-based time stretching, though widely used on short-form platforms like TikTok and Youtube Shorts, would alter the speech's pitch when changing its playback speed. In contrast, pitch-preserving time stretching, such as phase vocoder[26] and WSOLA[44] can preserve the harmonic structure of speech signal while reducing/increasing the number of periods to alter the duration. However, pitch-preserving methods could still introduce artifacts. The artifacts could go unnoticeable for a minimal speed-up but become more prominent as the degree of stretching increases–the words become less intelligible as phonemes are clustered together. Though intuitively understandable, the relationship between speeding up and the naturalness/intelligibility of speech is unclear, let alone in the context of social media audio stories. Therefore, we conducted a listening study to investigate how much speed-up, when exceeded, would result in perceivable degradation of naturalness and intelligibility.

3.2.1 Experiment Setup. We applied WSOLA to recordings collected in Section 3.1 with different speed-up rates and conducted a listening test on Amazon Mechanical Turk (AMT) [14] to obtain subject ratings for time-stretched audio stories. Workers were required to be in the United States and use English as the primary language. We selected 20 recordings that do not contain content that may be controversial or indicative of personal information. The recordings equally consisted of 10 male voices and 10 female voices. For each recording, we first split the audio based on silences to ensure we extracted excerpts that contain speech and that each excerpt was followed by a silence to avoid cutting in between words. We then randomly selected four excerpts per recording and sped up them with stretching factors ranging from 100% (original speed) to 200% (2x speed) with a common difference of 5% from 100% to 150% and a common difference of 10% from 150% to 200%. We also provided a 300% speed-up sample for validation, which was not used in the final analysis. We expected the turkers to consider the original recording more natural than average and the 300% speed-up less natural than average. Otherwise, we considered they ignored the instructions and thus invalidated the answers. In total, we stretched each audio to 17 different speeds, and 1360 audio recordings were generated for turkers to provide subjective ratings.

3.2.2 Study Procedure. In each HIT (Human Intelligence Task), we asked the turkers to perform one screening test and 20 individual tests, of which 16 were the actual rating tests and 4 were validation tests. For the screening test, we played back five audio samples and a reference sample and asked the turkers to pick out which of the five was identical to the reference sample. All five samples contain minimal artifacts that cannot be heard by people with hearing issues or who used a device that cuts off high or low frequencies. The turkers could proceed with the task only if they passed the screening test. In each rating test, we played one sped-up excerpt and asked the turkers to rate its naturalness on a 5-point scale. We instructed the turkers that 1 means unintelligible and 5 means sounding like a natural and unaltered recording. A rating of 2-4 means the audio is intelligible but not perfectly natural, with audible artifacts. Because the audio samples were also recorded by mechanical turkers in a prior study, the audio quality is non-professional. Therefore, we instructed the turkers to ignore audio quality issues such as background noise and echoes and focus on the naturalness of the speech. To avoid potential biases, the turkers rated 16 different utterances with 16 different speeds up for each HIT. All audio samples had to be played entirely (6 seconds long) before the turkers could enter the rating. The four validation tests also followed the same format, but the audio samples were either the original or the 300% sped-up version.

3.2.3 Findings. We collected 415 valid HITs from 221 unique turkers in 2 days. We received a total of 6640 answers or 415 ratings per speed-up. We then plot the average rating and the standard deviation of the average rating in Figure 3. We observed a steady monotonic decreasing naturalness rating as the speed-up factor increased. There is a slight but noticeable dip at 110% speed up, indicating that the turkers start to perceive a reduction of naturalness compared to unedited samples. There is a slight but statistically significant dip at 120% speed-up (p-value < 0.001, compared with 125%) that may be considered a soft limit for speed-up without significantly impacting the naturalness of the recording. It is also worth mentioning that the original audio samples (100%) were not considered perfectly natural by the turkers. This might have been because the turkers were not concurrently given unedited samples as a reference, thus providing the ratings conservatively. We did not want them to hear the original audio as it might bias their judgment, rating any sped-up sample as unnatural to game the task. As a take-away from this experiment, we consider speed-up up to 110% as no impact on naturalness and 120% as the upper limit. Moreover, since audio naturalness will be inevitably compromised as the speed-up rate increases beyond 120% threshold, we aim to design algorithms that can shorten audio further without harming its sound quality.

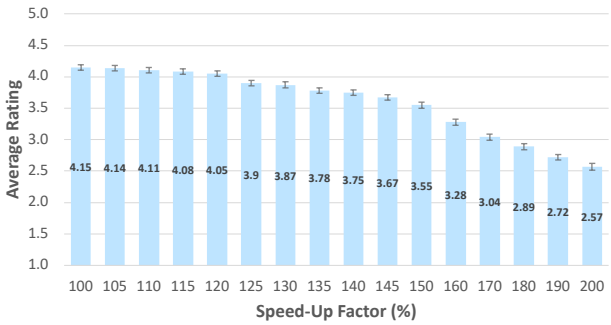

Figure 3: Average rating of naturalness as a function of speed-up rate. The error bars represent the standard deviation of the average. The naturalness rating monotonically decreases as the speed-up factor increases.

4 ROPE SYSTEM

4.1 Creating Audio Stories with ROPE

There are three steps in creating audio stories with ROPE: record, retarget, and refine. Users first **record** an audio story using ROPE's web interface with which they can playback or re-record the story until they are satisfied (figure 4A). ROPE will then automatically process the recording and **retarget** it to a user-specified length limit. The result will be sent back to the interface for users to preview (figure 4B). Since there is no "single correct answer" due to the subjective nature of audio stories, ROPE allows users to **refine** the algorithm output. Users can click on the *show editor* button to open ROPE's sentence editor (figure 4C and 4D) to review and finetune the edits made by ROPE. The sentence editor visualizes the selected sentences in green and the unselected ones in grey. It also presents each sentence's length to users. Whenever a new selection is made, the length of the animated bar below the editor will update to reflect the new audio length.

For the refine step, we investigated two designs of user-in-the-loop interaction paradigms. 1) *Toggle On/Off*: Based on the algorithm's suggestion, users click on sentences to select and deselect. The user must ensure that the selected sentences comply with the total length limit. 2) *Recompute with User Constraints*: Instead of toggling sentences on or off, users can click to specify sentences that must be included (blue) or must be excluded (red) in the final edit. By clicking on the *recompute* button, ROPE will run the optimization algorithm with the user-specified constraints to generate a new optimal sentence selection that fits the length limit. This paradigm frees users from the mental load in optimizing w.r.t. the length constraint, allowing them to focus on choosing the most/least important sentences. For both paradigms, users can click on *listen to the edit* button to listen to the audio of the current selection. Once users are satisfied with the content, they can click on the *finish edit* button to complete the refine step. Note that users can choose not to refine ROPE's outputs if they find them already satisfying.

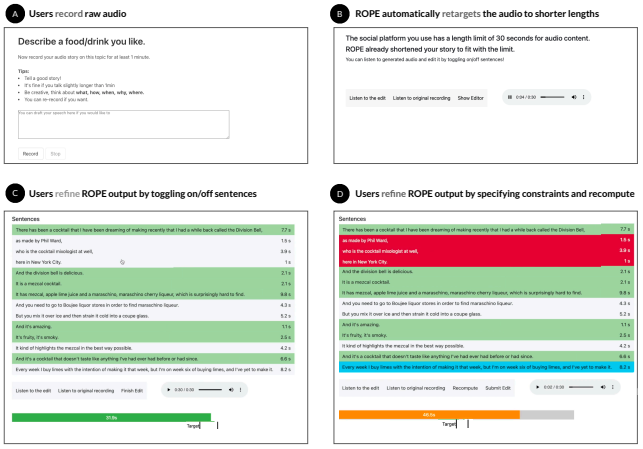

Figure 4: The user workflow of creating audio stories using ROPE's frontend interface. (A) Record: The user records a short-form audio story. (B) Retarget: ROPE automatically shortens the raw recording to the specified length limit. (C) and (D) Refine: The user can open ROPE's sentence editor to review the edits made by ROPE. Sentences included in ROPE's edit are highlighted in green. Users can refine the automatic edits by clicking on each sentence to select/deselect it. ROPE also enables users to identify sentences that must be included (blue) or excluded (red) by clicking a sentence multiple times to switch between each color. It then recomputes a new optimal sentence selection using our algorithm. ROPE visualizes the total length of currently selected sentences as a bar, whose color (green or orange) indicates whether the selection complies with the time limit or not. The target zone indicates the acceptable limit range discussed in section 4.5.2.

4.2 Algorithm Pipeline Overview

We now outline the algorithm design that enables ROPE's editing pipeline. Our algorithm takes an audio story and its transcription as input, extracts each sentence's audio and removes excessive silences, runs a length-constrained optimization, and generates a shortened audio output with sound quality enhancement (Figure 5). Ideally, shortened versions of an audio story should preserve as much important information as possible while fitting to the length limits. One plausible solution to shorten the content is using extractive summarization [20, 45, 47] which extracts representative sentences from the input audio story. Extractive summarization usually consists of two sequential phases [49]: 1) computing an affinity score to each sentence using a neural network and 2) selecting sentences based on the assigned scores. However, our early experimentation showed that directly applying extractive summarization to short-form audio stories does not generate satisfactory results for the following reasons. First, existing summarization models do not consider the audio length of each sentence *in audio* when extracting sentences. Secondly, most extractive summarization models were trained to shorten longer-form text data such as news articles [35] or live streaming [13], which may not generalize to our task of shortening an already short-form content.

To address these limitations, we designed our shortening algorithm to leverage abstractive summarization and each sentence's audio length to calculate sentence scores. We then used combinatorial optimization to select sentence combinations that achieve the highest total scores while complying with the length constraint. We designed the algorithm to operate at sentence level instead of smaller units as cropping words or phrases will likely lead to unpleasant artifacts containing choppy sounds. ROPE's algorithm pipeline consists of three components of processing: 1) Audio Preprocessing, 2) Sentence Score Calculation, and 3) Combinatorial Optimization. This pipeline is used in both retarget and refine steps in users' content creation process. In the following sections, we present each component in detail.

4.3 Audio Preprocessing

Once an audio story is recorded, ROPE transcribes it using a cloud-based speech-to-text API provided by Speechmatics. The transcriptions come with punctuation and timestamps for each word. Our experiment found that the timestamp accuracy is decent enough for sentence segmentation and word cutting. We also utilize the

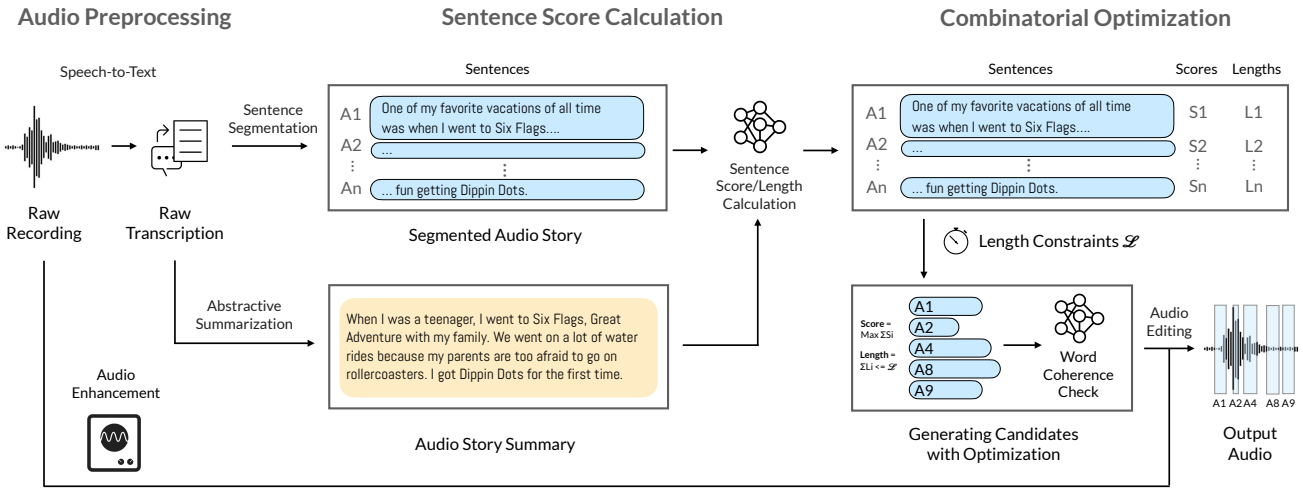

Figure 5: The ROPE pipeline consists of audio preprocessing, sentence score calculation, and combinatorial optimization.

timestamps to find silences and pauses, defined by a time gap between the ending of a word and the starting of the next word. Since we expect that average social content creators would not own professional microphones and an anechoic studio for recording, we apply HiFiGAN [43], an audio enhancement technology that can turn recordings from consumer devices into studio-level quality audio using denoising and dereverberation. The enhanced audio would be used to produce the shortened audio story output.

4.4 Sentence Score Calculation

ROPE segments the transcribed audio stories into sentences and applies natural language processing techniques to obtain a score for each sentence. Using each sentence's score and length in audio, combinatorial optimization is performed to decide which sentences to include so that the score is maximized and the time limit is satisfied.

4.4.1 Speech Sentence Segmentation. Speech sentence segmentation separates a full recording into segments. We leverage the punctuation and word timestamps obtained from the speech transcription service. Due to transcription errors and the nature of spoken language, speech may not be grammatically correct. Therefore, the punctuation is often unsatisfactory. Hence, we designed a rule-based method to refine the segmentation. We first use the period, question, and exclamation marks to crop the transcript into segments as they are clear indicators of a sentence's end. After that, we check sentences with commas as they might be lengthy sentences consisting of multiple sub-sentences which require further splitting. However, naively splitting sentences with commas might result in short sentences with only one word. Therefore, we split a sentence with commas only when specific criteria are met. If the words before and after a comma are separated with a long pause (1 second), we split the sentence with the comma. For sentences longer than 10 seconds with multiple clauses connected by commas, we split the clauses longer than five words. The thresholds were set empirically, and our method achieved decent performance. However, we recognize that rule-based methods may not always perform as speaking styles vary between users, and transcription errors, which happen more often for non-native speakers, would affect our sentence segmentation's performance. We will further discuss the limitations of our sentence segmentation in the discussions.

4.4.2 Sentence Score Function. ROPE shortens the audio stories by selecting a set of sentences that covers the main topic and are the most relevant to the topic while complying with the time limit. To achieve so, ROPE first identifies the topics/theme of the audio story using abstractive summarization [21, 28, 37], which generates concise summaries that may contain phrases and words that were not in the original text. We use a BART-based summarization model with a max sequence length of 1024 words (equals ~8 mins of speech according to the words per minute found in our data collection study). We treat each sentence in summary as a topic sentence and define the sentence score as how relevant a sentence in the story is to the topics. We proposed a distance-based score function, as shown in equation 1. We first embed each sentence in the audio story and the summary using pre-trained Sentence-BERT embedding[29]. For each sentence A_i in the audio story, we calculate the cosine distances between the embeddings of it and every summary sentences S_j. This gives us how "close" A_i is to each topic, in the embedding space. Inspired by inverse distance weighting methods [10, 32], we inversed the cosine distances to obtain likelihood measures. The smaller the distance, the higher the likelihood of selecting it. Since we treat every topic in summary equally, we use the maximum of the inversed distances that indicate the highest relevance to any summary sentence. We then multiply the max inversed distance with the length of the audio story sentence L_i. This additional length factor is introduced to encourage the combinatorial optimization to favor longer sentences over short sentences to make the narratives more coherent.

$$Score(Ai) = \max_{S_j \in sum} \frac{1}{D(Emb(A_i), Emb(S_j))} * L_i \quad (1)$$

4.5 Selecting Sentence using Combinatorial Optimization

The objective of automatic audio shortening is to satisfy length constraints while maximizing the amount of important information in the shortened content, determined by the sentence scores. Therefore, the shortening task can naturally be formulated as a length-constrained combinatorial optimization problem, equivalent to solving a classic 0/1 knapsack problem [15, 39]. In this case, each sentence is an "item" whose weight is the audio length and value is the sentence score. The goal is to fill the knapsack (i.e., the social media post length limit) while maximizing the total sentence score.

4.5.1 Length-Constrained Score Optimization. Given a set of sentences $A_1, A_2, ...A_n$, each with a length L_i and a score $Score(Ai)$, our goal is to find a subset of sentences so that the total length is less than or equal to a given limit L_m and the total score is as large as possible. Therefore, our algorithm maximizes

$$\sum_{i=1}^{n} Score(Ai)X_i \tag{2}$$

subject to $\sum_{i=1}^{n} L_i X_i <= L_m$. Here $Xi \in \{0, 1\}$ and $\forall i = 1, ..., n$. We used dynamic programming to efficiently solve the optimization problem, which runs in $O(nm)$, where n is the number of sentences and m is length limit, as opposed to brute-force search which runs in exponential time $O(2^n)$. Since dynamic programming requires the lengths to be integers for tabulation, we quantize each sentence's length with 0.1s and multiply them by 10. For example, a 6.7s sentence becomes 67 units after quantization.

4.5.2 Optimizing with Acceptable Limit Range. During our pilot tests with the algorithm, we found that sometimes the length of the shortened content was shorter than the limit because adding any other new sentence would break the constraint. However, we also noticed that in many cases, including a new sentence could introduce significant content while only slightly exceeding the limit. Such a phenomenon becomes more salient when the length limit gets shorter. Considering the following scenario: the best sentence selection strictly complying with a length limit of 15s might have only a length of 11s. Selecting another 4.5s sentence only exceeds 0.5s, yet it could convey approximately 30% of new content, which is preferred. To accommodate such scenarios, we allow our algorithm's selections to exceed the length limits slightly based on our study findings that a small speed-up has minimal effects on sound naturalness and quality. We allow our algorithm to select sentence sets whose lengths are 10% or less above the limits. We then use audio time stretching to speed up the audio outputs so they still strictly comply with the time constraints.

However, having the additional flexibility in length also means there could be multiple optimal sentence selections (optimal w.r.t. different length limits within the 10% range, respectively). Therefore, we enumerate all the possible sentence selections within the acceptable limit range using the calculated dynamic programming table. We then choose the best selection among the candidates by examining the word coherence using an approach similar to Rescribe [38]. We input the concatenation of each sentence selection to the GPT-2 language model and use the output log loss, which estimates the probability of the word sequence, as the word coherence metric. We choose the candidate with the smallest GPT-2 loss as the final selection. Finally, ROPE crops the selected sentences from the enhanced audio recordings and concatenates them to generate the final audio outputs.

4.6 Example Results

Figure 1 shows how ROPE shortens an audio story recorded by one of our user study participants (Section 5). Figure 1 left illustrates ROPE's sentence score calculation process. Each sentence in the original audio story is linked to the summary sentence with which it generates the highest score. The numbers associated with the links are the sentence scores. Based on the visualization, we can see that each summary sentence represents a distinct topic within the story and that adjacent sentences in the audio story tend to map to the same topic. It also shows the limitation that the summary may not capture every topic in the story. For example, the last sentence *"Every week I buy limes with the intention of making it that week, but I'm on week six of buying"* was not captured by any summary sentence. Figure 1 right shows the results of ROPE's shortening the audio story (71s) to three different lengths: 45s, 30s, and 15s. We show the result of 15's multiples as they are standard length limits for social platforms such as Instagram and Tiktok. ROPE can shorten the input to other lengths as well. Comparing the sentence selections between different length limits, we observe that the shorter versions' sentence selections are always subsets of the longer versions. Such behavior is expected for combinatorial optimization, and we believe it also mimics many's mental processes when attempting to shorten a piece of content to multiple lengths.

4.7 Implementation Details

We implemented the ROPE pipeline in Python. For sentence similarity, we used the `all-MiniLM-L6-v2` pre-trained model from the open-sourced implementation of sentence-BERT, which offers good quality while being 5x faster than the full model, suitable for interactive systems. We used the `philschmid/bart-large-cnn-samsum` model from `huggingface.co` for abstractive summarization. The model performed better than other BART-based variants we tried as it was finetuned on the SAMSum dataset [19] which consists of dialogue data relatively similar to spoken language. We used an GPT-2 model implementation from `huggingface.co` as well. We used python libraries `PyTSMod` to stretch audio and `pydub` to edit the final audio output. We implemented ROPE as a Flask web application running on a Linux machine with a V100 GPU.

5 USER STUDY

We aim to design a system that facilitates users to create shorter versions of recorded content quickly and easily. To evaluate the effectiveness of our algorithm and design choices, we conducted a user study to understand 1) how satisfied users are with the shortened content automatically generated by ROPE and 2) how useful is ROPE's output as suggestions for users when shortening audio stories? We also seek to solicit users' subjective feedback on our proposed sentence-based editing paradigms that allow user-in-the-loop refinements of automatic outputs.

5.1 Participants

We recruited 6 participants (4 male and 2 female) within our organization. All participants had prior experience creating social media content, with P5 self-reported as "very frequent"; P2, P3, P6 as "sometimes"; and P1, P4 "have done it for a few times." The distribution of proficiency matches our envisioned user group. We seek to understand how typical users with at least some prior social media experience would use ROPE and how ROPE can make authoring short-form audio content easier.

5.2 Methodology

The study follows a within-subject design. We compared system conditions with and without ROPE. We tested both user-in-the-loop refinement designs presented in section 4.1. The study has three editing conditions:

- ***ROPE-toggle*** tests the *Toggle On/Of* paradigm, which generates an automatic result and then allows users to refine the result by togging on and off sentence selections.
- ***ROPE-recompute*** tests the *Recompute with User Constraints* design where users specify sentences constraints(must-be-included and must-be-excluded) and ROPE will recompute based on the constraints. When no constraints are made, the automatic result is presented.
- ***Manual*** condition asks users to shorten an audio story using the sentence selection interface similar to *ROPE-toggle* but all sentences were selected at the start without using our algorithm. The design mimics the mental process of shortening audio by removing its parts. We also provided a "deselect all" button and explicitly instructed the users that they could start from an empty set if desired.

For each study condition, the user recorded an audio story and shortened it to three length limits 45s, 30s, and 15, simulating three social media constraints. Therefore, the study has nine shortening tasks in total (3 audio, each needs to be shortened to 3 length limits). The three conditions were counterbalanced with a 3x3 Latin square. For the two conditions with ROPE, the user could choose not to make any refinement if they feel satisfied with the algorithm output.

5.3 Procedure

The study was conducted remotely. Participants used their computers and communicated with the experimenter using a videoconferencing tool. The participants were asked to share their screen and computer sound so the experimenter could record the study. To record their audio, three participants (P2, P3, and P6) used their laptop's built-in microphones, two used external microphones (P4 and P5), and one used Airpods (P1). Before the study, the experimenter first demonstrated each of the three conditions. The participant received a URL link to the study website serving on a cloud machine. Similar to the formative data collection, participants will listen to a short audio example and test their microphone with speech recognition before starting the tasks. At the beginning of each experiment condition, participants chose a topic prompt from a provided list (11 topics in total) to record their short audio story. They were instructed to record audio between 1 to 1.5 minutes and could re-record if needed. The recordings would then be uploaded to the server, and the participants were required to listen to their recordings again to familiarize themselves with the content while waiting for the back-end processing. Once the processing was done, participants would be redirected to an editing page, following one of the three conditions mentioned above.

5.4 Subjective Ratings

We solicited subjective ratings through post-stimulus questionnaires. Ratings were given after a task/condition was finished.

5.4.1 Outcome Satisfaction. For each of the nine tasks, we asked participants to rate how satisfied they were with their shortened outcome to feel comfortable sharing on social platforms. The ratings are on a 5-point Likert scale (1–very unsatisfied, 5–very satisfied). For tasks where algorithm suggestion was provided, we solicited additional ratings regarding the automatic algorithm output.

5.4.2 Algorithm Usefulness. For system conditions with ROPE suggestion, we asked participants how useful the automatic result was when performing audio shortening tasks on a 5-point Likert scale, with 1 being not useful at all and 5 being very useful.

5.4.3 Mental Demand, Effort, and Performance. To understand the task workload of each condition, we asked the participants to rate on *Mental Demand*: How mentally demanding is the task? *Effort*: How much effort did they have to make to accomplish the task? *Overall Performance*: How successful were they in performing the task? All 3 questions were rated on a 10-point Likert scale, with 10 being very demanding, requires very high effort, and very successful, respectively.

5.5 Results

5.5.1 Outcome Satisfaction. As shown in figure 6, the output of automatic methods already achieves similar satisfaction scores with manual editing. Our results indicate that shortening around 1-minute content to 30 seconds yields the highest average satisfaction for both manual and automatic algorithm outputs (Mean=3.89, STD=1.18). Moreover, using the algorithm without user interaction yields nearly the same satisfactory score as manual editing for all length limits, suggesting that our algorithm could possibly replace manual effort in retargeting audio recordings to different length limits. Moreover, using either *ROPE-toggle* or *ROPE-recompute* interface to refine the automatically generated output significantly boosted the satisfaction scores. For *ROPE-toggle* and *ROPE-recompute*, the user refinements based on automatic suggestion on average respectively increase 36.6% and 35.8% satisfaction scores with the 15 seconds content receiving most significant boosts of 52.9% and 46.7%. Combining results from *ROPE-toggle* and *ROPE-recompute*, user-in-the-loop refinements for our algorithm increases the mean score of automatic algorithm output from 3.25 (STD=1.23) to 4.35 (STD=0.87). A further paired T-test shows that *ROPE-toggle* significantly outperforms manual editing with a p-value of 0.001, so does *ROPE-recompute* with a p-value of 0.004. The results indicate that with algorithm suggestions, users could create more satisfying content than doing the shortening task alone.

5.5.2 Usefulness. Echoing the results of the satisfaction rating, the average usefulness rating is 4.17 (STD=0.72), showing that the participants found ROPE's algorithm useful for creating shortened

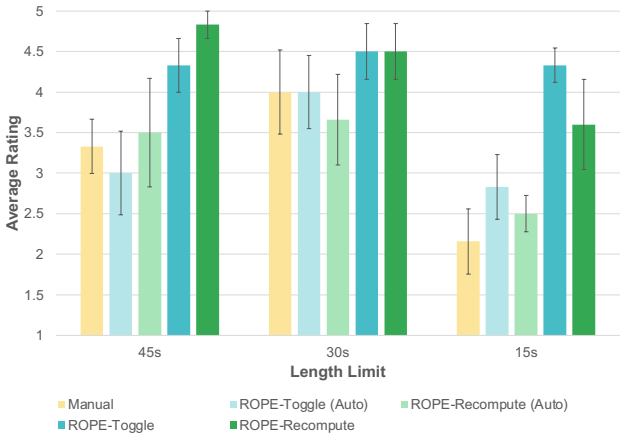

Figure 6: Average ratings of outcome satisfaction on a 5-point Likert scale. The ratings for *Manual, ROPE-Toggle, ROPE-Recompute* indicate user satisfaction for the final outcome of each study condition. The *ROPE-Toggle* (Auto) and *ROPE-Recompute* (Auto) ratings represent satisfaction for the intermediate results generated by algorithm without user finetuning. Error bars were drawn with standard error.

content. P2 pointed out how the algorithm improves the content *"I was very impressed, and I liked how it was smart about, you know, making things snappier."* P3 commented *" It was very interesting for me to realize that without the actual suggestions, it was really hard for me to just go over it and try to do it manually."* These comments highlight the usefulness of ROPE's automatic output. Participants also pointed out limitations where ROPE could improve. For example, P1 and P2 mentioned that it would have been nice for ROPE to further remove the filler and repetitive words that commonly appear in spoken language. When asked about users' preferences over the three tested conditions for shortening audio, the participants unanimously preferred ROPE over manual editing. However, participants favored different ROPE conditions, with two participants (P1 and P3) preferring *ROPE-recompute* and the other four liked *ROPE-toggle* more. P4 commented *"(ROPE-toggle) provides the most straightforward suggestions for me"* while P3 commented *"the overall experience (of ROPE-recompute) is really good."* In some cases, we did observe that users found the *ROPE-recompute* condition less intuitive than *ROPE-toggle*'s direct manipulation [23], since users may not be able to predict what recomputed suggestions would be. However, P3 pointed out that *"I liked to be surprised by what the algorithm thought that would go"* and P3 commented *"sometimes the suggestion will give me something that I did not expect, but still makes sense for content creation purposes...That also gives some new inputs on how to make it more concise or highlights some other points (of the story)."*. The user feedback signals that being able to mode-switch between *ROPE-toggle* and *ROPE-recompute* may lead to a better editing experience.

5.5.3 Mental Demand, Effort, and Overall Performance. As shown in figure 7, both *ROPE-toggle* and *ROPE-recompute* achieves lower average ratings for mental demand and effort, compared to the *Manual* condition. The participants were also more successful in performing the shortening tasks using ROPE-based approaches. P4 commented, *"One of my most favorite things (about ROPE) was just submitting and getting the automatic response. Um, simply because this is the least effort."* while P4 thought it was *"super easy"* to use ROPE for shortening audio stories. However, manual editing can sometimes also requires minimal effort only. For example, P3 commented using *Manual* condition was also easy as she already had a rough idea of key points she wanted to put in the content.

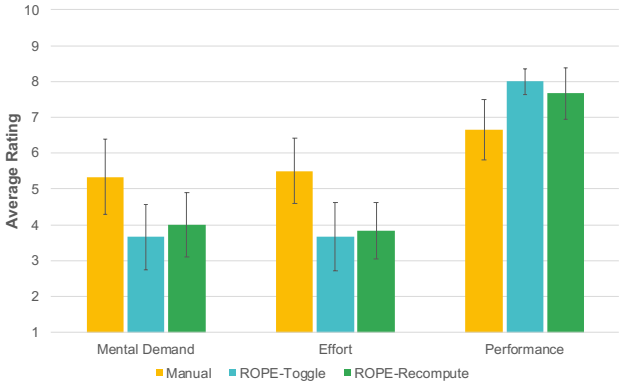

Figure 7: Average ratings of *Mental Demand, Effort,* and *Overall Performance* for each study condition on a 10-point Likert scale. Error bars were drawn with standard error.

5.5.4 Shortening Strategy. We interviewed the participants about their strategies to shorten the content using the provided sentence selection interface. The participants all pointed out similar strategies: they would first think about the main topics of the recordings and try to preserve sentences covering important topics when shortening the content. Such strategies resemble our summarizing-based algorithm, which also identifies the main topics in audio stories. Moreover, P2 pointed out he was storytelling-oriented, so he would try to make sure the shortened stories cover three elements of storytelling, 1) exposition: set the scene, context, 2) conflict: something happens, and 3) resolution: the "happily ever after." He achieved this in 2 out of the 3 stories he shortened.

6 DISCUSSION AND LIMITATIONS

We discuss the observations during the experiments, the limitations of our solution, and how future work could build upon ROPE.

6.1 Speech-to-Text

Though speech-to-text technology has improved significantly, we notice a few issues still present as limiting factors to ROPE's user experience. Firstly, incorrectly transcribed words may alter the meaning of a sentence and affect the accuracy of the abstractive summarization model we use in this work. The transcription accuracy is often lower for non-native speakers, which may affect usability to a broader audience. Moreover, the punctuations of the transcript are inferred based on grammar and pauses in the audio. However, audio story recordings may not follow rigorous grammar

and may have pauses and silences caused by disfluencies and emphasis. As speech-to-text technology improves, we believe ROPE's performance will be further boosted as well.

6.2 Text Summarization

A limitation of abstractive summarization models is that they were not trained to comprehensively cover all the content and topics in the original text. Therefore, some subtopics in the audio story might not be covered in summary and would therefore be less favored by our optimization algorithm. Future work could explore steerable summarization models that allow users to decide how many topics they would like to include in the shortened version and train dedicated models for shortening short-form audio stories.

6.3 Sentence-Based Editing

We limited the individual units to sentences, as they usually have silence on both ends leading to a higher chance of natural sounding cuts. However, our user study participants expressed that they wished to be able to break sentences into smaller segments to obtain more fine-grained editing options. This signals that our sentence segmentation's performance does not perfectly align with human expectations, likely because of the inaccurate punctuation provided by the transcription service and the limitations of our rule-based refinement method. However, enabling users to cut words/phrases without generating noticeable artifacts is challenging, and algorithms mitigating the resulting discontinuities remain an ongoing research problem in the audio processing community.

We have also experimented with filler-word detection to remove disfluencies such as repeated words and fillers like *'uh'* and *"ah"*. However, removing filler words suffer from the same challenges of yielding sound artifacts. Therefore, we decided not to support this feature in ROPE's pipeline to preserve the overall sound quality. Speech synthesis methods [24, 25, 34] could be used to refine the unpleasant results caused by word/phrase removal. However, they typically require training personalized models over at least 10 minutes of recordings, which is unavailable in our use case. Future work can explore speech synthesis techniques to preserve the naturalness while requiring less personal training data.

6.4 Alternative Algorithms

The objective of our study was to compare editing experience with and without ROPE's suggestions, so we have not exhaustively compared ROPE's core algorithm to alternative baselines. Simple algorithms such as selecting the first N sentences or choosing the longest K sentences until the length limit is met can also generate shortened stories. However, they do not consider the semantics of audio stories and may omit important information in the original recordings. For example, while selecting the first N sentences may cover the opening, it can lose sub-topics that appear later in the recording and may not select the ending sentences, resulting in an abrupt story ending. Speeding up can also shorten audio stories to any desired length. However, our study found that speeding up over 120% (i.e., shorten to less than 83.3%) would significantly impact the sound naturalness and intelligibility.

6.5 Editing vs. Re-recording

Another feasible way to author shorter versions of an audio story is to re-record, instead of editing audio with tools like ROPE. However, with re-recording, creators still have to track time by themselves during live performance to ensure the content fits within the time constraints, which is challenging. Moreover, re-recording extra versions for different platforms can be time-consuming. As a result, video creators often repurpose the same content across multiple platforms [7], which requires editing to ensure the new versions fit with platforms' format requirements, such as length limits and aspect ratio. We believe similar practices would apply to audio social media platforms as well. Nonetheless, with audio social media platforms starting to emerge, it is unclear whether audio styles on different platforms would vary significantly. Re-recording will be necessary to create versions with different styles as ROPE currently does not explicitly address style variances for different versions of audio stories.

6.6 Generalizing to Other Media

ROPE focuses on short-form audio content, but it can also be applied to the audio portion of short-form video and short-form text such as tweets. In our preliminary experiments, we found that ROPE can also effectively shorten short speech-heavy videos, such as talking-head videos and videos with narration. However, depending on the video styles, it might require additional mechanisms to ensure that the corresponding visual edits are fluent. ROPE's algorithm could also be generalized to shortening text content by optimizing the sentence scores based on the number of characters rather than the duration of sentences in audio. ROPE's current design may not directly scale to long-form audio as they often contains more topics that abstractive summarization may not fully capture, which could lead to missing information when shortened. Applying hierarchical summarization [27] may be helpful, but it requires further study, which we leave as future work.

7 CONCLUSION

We have presented ROPE, a system that automatically shortens voice content to target lengths required by social media platforms. ROPE's workflow has three steps - record, retarget, and refine. A user first records their content without worrying about the target length. Our system then retargets (shortens) the recording to comply with the target length by preserving the sentences that contribute the most to the overall story and automatically removing the rest. It can apply a slight speed up, within limits informed by our formative studies, to provide flexibility for shortening the audio. Finally, the user can refine the results using our sentence selection interface. Our user study indicates that voice content automatically shortened by our system produces results of comparable quality to manually edited content. Moreover, interactively refining content using our system produces significantly higher quality results. This suggests that ROPE is an effective tool for quickly creating high-quality short-form voice content.

REFERENCES

[1] 2022. Audacity. https://www.audacityteam.org/.
[2] 2022. Audition. https://www.adobe.com/ca/products/audition.html.

[3] 2022. Cappuccino. https://apps.apple.com/us/app/cappuccino-podcast-w-friends/id1506849927.

[4] 2022. Clubhouse). https://www.clubhouse.com/.

[5] 2022. Facebook Soundbite. https://www.facebook.com/business/help/407115354300204?id=391624202615570&ref=search_new_947.

[6] 2022. How to Change the Speed of Your Videos: Step-by-Step Tutorial. https://www.descript.com/blog/article/how-to-change-the-speed-of-your-videos-step-by-step-tutorial.

[7] 2022. How to Easily Repurpose Video Content Across Social Media Channels (YouTube, TikTok, Instagram, Pinterest and More). https://later.com/blog/repurpose-video-content/.

[8] 2022. Logic Pro. https://www.apple.com/logic-pro/.

[9] 2022. Videos are getting shorter...and here's why! https://www.azonetwork.com/marketing-science/blog/videos-are-getting-shorterand-heres-why.

[10] Patrick M Bartier and C Peter Keller. 1996. Multivariate interpolation to incorporate thematic surface data using inverse distance weighting (IDW). *Computers & Geosciences* 22, 7 (1996), 795–799.

[11] Floraine Berthouzoz, Wilmot Li, and Maneesh Agrawala. 2012. Tools for Placing Cuts and Transitions in Interview Video. *ACM Trans. Graph.* 31, 4, Article 67 (jul 2012), 8 pages. https://doi.org/10.1145/2185520.2185563

[12] Juan Casares, A Chris Long, Brad A Myers, Rishi Bhatnagar, Scott M Stevens, Laura Dabbish, Dan Yocum, and Albert Corbett. 2002. Simplifying video editing using metadata. In *Proceedings of the 4th conference on Designing interactive systems: processes, practices, methods, and techniques*. 157–166.

[13] Sangwoo Cho, Franck Dernoncourt, Tim Ganter, Trung Bui, Nedim Lipka, Walter Chang, Hailin Jin, Jonathan Brandt, Hassan Foroosh, and Fei Liu. 2021. StreamHover: Livestream Transcript Summarization and Annotation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 6457–6474. https://doi.org/10.18653/v1/2021.emnlp-main.520

[14] Kevin Crowston. 2012. Amazon mechanical turk: A research tool for organizations and information systems scholars. In *Shaping the future of ict research. methods and approaches*. Springer, 210–221.

[15] Arnaud Fréville. 2004. The multidimensional 0–1 knapsack problem: An overview. *European Journal of Operational Research* 155, 1 (2004), 1–21.

[16] Ohad Fried, Ayush Tewari, Michael Zollhöfer, Adam Finkelstein, Eli Shechtman, Dan B Goldman, Kyle Genova, Zeyu Jin, Christian Theobalt, and Maneesh Agrawala. 2019. Text-based Editing of Talking-head Video. https://doi.org/10.48550/ARXIV.1906.01524

[17] Sadaoki Furui, Tomonori Kikuchi, Yosuke Shinnaka, and Chiori Hori. 2004. Speech-to-text and speech-to-speech summarization of spontaneous speech. *IEEE Transactions on Speech and Audio Processing* 12, 4 (2004), 401–408.

[18] Sebastian Gehrmann, Yuntian Deng, and Alexander M Rush. 2018. Bottom-up abstractive summarization. *arXiv preprint arXiv:1808.10792* (2018).

[19] Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. SAMSum Corpus: A Human-annotated Dialogue Dataset for Abstractive Summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*. Association for Computational Linguistics. https://doi.org/10.18653/v1/d19-5409

[20] Nianlong Gu, Elliott Ash, and Richard H. R. Hahnloser. 2021. MemSum: Extractive Summarization of Long Documents Using Multi-Step Episodic Markov Decision Processes. https://doi.org/10.48550/ARXIV.2107.08929

[21] Som Gupta and S. K Gupta. 2019. Abstractive summarization: An overview of the state of the art. *Expert Systems with Applications* 121 (2019), 49–65. https://doi.org/10.1016/j.eswa.2018.12.011

[22] Bernd Huber, Hijung Valentina Shin, Bryan Russell, Oliver Wang, and Gautham J Mysore. 2019. B-Script: Transcript-based B-roll Video Editing with Recommendations. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–11.

[23] Edwin L Hutchins, James D Hollan, and Donald A Norman. 1985. Direct manipulation interfaces. *Human–computer interaction* 1, 4 (1985), 311–338.

[24] Zeyu Jin, Gautham J Mysore, Stephen Diverdi, Jingwan Lu, and Adam Finkelstein. 2017. Voco: Text-based insertion and replacement in audio narration. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–13.

[25] Kundan Kumar, Rithesh Kumar, Thibault de Boissiere, Lucas Gestin, Wei Zhen Teoh, Jose Sotelo, Alexandre de Brébisson, Yoshua Bengio, and Aaron C Courville. 2019. MelGAN: Generative Adversarial Networks for Conditional Waveform Synthesis. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2019/file/6804c9bca0a615bdb9374d00a9fcba59-Paper.pdf

[26] Jean Laroche and Mark Dolson. 1999. Improved phase vocoder time-scale modification of audio. *IEEE Transactions on Speech and Audio processing* 7, 3 (1999), 323–332.

[27] Daniel Li, Thomas Chen, Albert Tung, and Lydia B Chilton. 2021. Hierarchical Summarization for Longform Spoken Dialog. In *The 34th Annual ACM Symposium on User Interface Software and Technology*. 582–597.

[28] Hui Lin and Vincent Ng. 2019. Abstractive summarization: A survey of the state of the art. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 9815–9822.

[29] Hui Lin and Vincent Ng. 2019. Abstractive summarization: A survey of the state of the art. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 9815–9822.

[30] Yang Liu and Dilek Hakkani-Tür. 2011. Speech summarization. *Spoken language understanding: Systems for extracting semantic information from speech* (2011), 357–396.

[31] Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. *arXiv preprint arXiv:1908.08345* (2019).

[32] George Y Lu and David W Wong. 2008. An adaptive inverse-distance weighting spatial interpolation technique. *Computers & geosciences* 34, 9 (2008), 1044–1055.

[33] Yu-Fei Ma, Lie Lu, Hong-Jiang Zhang, and Mingjing Li. 2002. A user attention model for video summarization. In *Proceedings of the tenth ACM international conference on Multimedia*. 533–542.

[34] Max Morrison, Lucas Rencker, Zeyu Jin, Nicholas J. Bryan, Juan-Pablo Caceres, and Bryan Pardo. 2021. Context-Aware Prosody Correction for Text-Based Speech Editing. https://doi.org/10.48550/ARXIV.2102.08328

[35] Ramesh Nallapati, Bowen Zhou, Cicero Nogueira dos santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond. (2016). https://doi.org/10.48550/ARXIV.1602.06023

[36] Ani Nenkova and Kathleen McKeown. 2012. A survey of text summarization techniques. In *Mining text data*. Springer, 43–76.

[37] Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304* (2017).

[38] Amy Pavel, Gabriel Reyes, and Jeffrey P. Bigham. 2020. *Rescribe: Authoring and Automatically Editing Audio Descriptions*. Association for Computing Machinery, New York, NY, USA, 747–759. https://doi.org/10.1145/3379337.3415864

[39] David Pisinger. 1997. A minimal algorithm for the 0-1 knapsack problem. *Operations Research* 45, 5 (1997), 758–767.

[40] Steve Rubin, Floraine Berthouzoz, Gautham J. Mysore, Wilmot Li, and Maneesh Agrawala. 2013. Content-Based Tools for Editing Audio Stories. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology* (St. Andrews, Scotland, United Kingdom) *(UIST '13)*. Association for Computing Machinery, New York, NY, USA, 113–122. https://doi.org/10.1145/2501988.2501993

[41] Guokan Shang, Wensi Ding, Zekun Zhang, Antoine Jean-Pierre Tixier, Polykarpos Meladianos, Michalis Vazirgiannis, and Jean-Pierre Lorré. 2018. Unsupervised abstractive meeting summarization with multi-sentence compression and budgeted submodular maximization. *arXiv preprint arXiv:1805.05271* (2018).

[42] Hijung Valentina Shin, Wilmot Li, and Frédo Durand. 2016. Dynamic Authoring of Audio with Linked Scripts. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) *(UIST '16)*. Association for Computing Machinery, New York, NY, USA, 509–516. https://doi.org/10.1145/2984511.2984561

[43] Jiaqi Su, Zeyu Jin, and Adam Finkelstein. 2020. HiFi-GAN: High-fidelity denoising and dereverberation based on speech deep features in adversarial networks. *arXiv preprint arXiv:2006.05694* (2020).

[44] Werner Verhelst and Marc Roelands. 1993. An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech. In *1993 IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 2. IEEE, 554–557.

[45] Danqing Wang, Pengfei Liu, Ming Zhong, Jie Fu, Xipeng Qiu, and Xuanjing Huang. 2019. Exploring Domain Shift in Extractive Text Summarization. https://doi.org/10.48550/ARXIV.1908.11664

[46] Ke Zhang, Wei-Lun Chao, Fei Sha, and Kristen Grauman. 2016. Video summarization with long short-term memory. In *European conference on computer vision*. Springer, 766–782.

[47] Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive Summarization as Text Matching. https://doi.org/10.48550/ARXIV.2004.08795

[48] Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, et al. 2021. QMSum: A new benchmark for query-based multi-domain meeting summarization. *arXiv preprint arXiv:2104.05938* (2021).

[49] Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. 2018. Neural Document Summarization by Jointly Learning to Score and Select Sentences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, 654–663. https://doi.org/10.18653/v1/P18-1061