Txt2Vid: Ultra-Low Bitrate Compression of Talking-Head Videos via Text

Pulkit Tandon, Shubham Chandak, Pat Pataranutaporn, Graduate Student Member, IEEE, Yimeng Liu, Anesu M. Mapuranga, Pattie Maes, Tsachy Weissman, Fellow, IEEE, and Misha Sra

Abstract—Video represents the majority of internet traffic today, driving a continual race between the generation of higher quality content, transmission of larger file sizes, and the development of network infrastructure. In addition, the recent COVID-19 pandemic fueled a surge in the use of video conferencing tools. Since videos take up considerable bandwidth (∼100 Kbps to a few Mbps), improved video compression can have a substantial impact on network performance for live and pre-recorded content, providing broader access to multimedia content worldwide. We present a novel video compression pipeline, called Txt2Vid, which dramatically reduces data transmission rates by compressing webcam videos (“talking-head videos”) to a text transcript. The text is transmitted and decoded into a realistic reconstruction of the original video using recent advances in deep learning based voice cloning and lip syncing models. Our generative pipeline achieves two to three orders of magnitude reduction in the bitrate as compared to the standard audio-video codecs (encoders-decoders), while maintaining equivalent Quality-of-Experience based on a subjective evaluation by users (n = 242) in an online study. The Txt2Vid framework opens up the potential for creating novel applications such as enabling audio-video communication during poor internet connectivity, or in remote terrains with limited bandwidth. The code for this work is available at https://github.com/tpulkit/txt2vid.git.

Index Terms—Video compression, perceptual compression, talking-head videos, generative models, voice cloning.

I. INTRODUCTION

VIDEO streaming represents the majority share of internet traffic today, with estimates as high as 80% [1]. With the COVID-19 outbreak, internet services have seen a surge in usage (∼50-100%), with video conferencing tools such as Zoom seeing ten times increase in usage [2]. A typical video conferencing call can consume anywhere from ∼100 Kbps to a few Mbps (bps refers to bits-per-second). Unfortunately, a vast majority of the world’s population does not have access to high bandwidth network connections [3] or faces intermittent connectivity issues. The ability to conduct an audio-video (AV) call at extremely low bitrates (∼100-1000 bps) could provide broader access to billions of people in developing countries or other locations with limited or unreliable broadband connectivity. Even with major improvements in bandwidth, advances in compression of data generated from video conferencing tools can provide broader access to this technology worldwide. Moreover, a reduction in required bandwidth can have a significant impact on global network performance by decreasing the network load [4], [5]. Thus, efficient compression of data generated using video conferencing tools is an important problem.

In this work, we propose and demonstrate a pipeline which can compress videos to text and reconstruct the videos from text using state-of-the-art deep learning based decoders. Given the recent extreme need and reliance on video conferencing, the focus of our work is on audio-video (AV) content transmitted from webcams during video conferencing or webinars. Current compression codecs (such as H.264 [6] or AV1 [7] for videos and AAC [8] for audio) lossily compress the input AV content by discarding details that have the least impact on user experience. However, the distortion measures targeted by these codecs are often low-level and attempt to penalize deviation from the original pixel values or audio samples. But in reality, what matters most is the final quality-of-experience (QoE) when a compressed media stream is shown to a human end-consumer [9]. It can be argued that the ultimate goal of AV transmission is to communicate “semantic information”. In fact, “semantic communication” [10], has seen a recent surge in research for various data modalities such as text [11], speech [12], Internet of Things (IoT) [13], and videos [14], to name a few (see Ref. [15] for a detailed review. Thus, in our proposed pipeline, instead of working with pixel-wise fidelity metrics we recreate the original content such that the QoE is maintained. Figure 1 contrasts the conventional compression approaches with our Txt2Vid approach. While conventional approaches typically require ∼10-100 Kbps of bandwidth, Txt2Vid achieves ultra-low bitrates of ∼100 bps by compressing videos to text. This can lead to multiple orders of magnitude compression advantage if we can achieve similar QoE as the conventional approaches in the low-bitrate regime.

There has been recent interest in the generative modeling community in reconstructing videos from lower bitrate alternatives such as text or low dimensional latent spaces [16], [17], [18], [19]. While we see significant progress in using
generative machine learning to model natural images from text [20], [21], [22], [23], [24], these approaches are currently unable to produce high-quality videos. To recreate webcam video data, 2D [25], [26], [27], [28], [29], [30], [31], [32] or 3D graphics based methods [33], [34], [35], [36] have been used successfully to generate realistic talking-head videos. Their success implies that machine learning methods have the potential to be used as decoders that can reconstruct a webcam video with high QoE, while requiring less data to be transmitted compared to standard codecs. A recent line of work on achieving lower bitrates for talking-head videos has focused on using a source image to encode facial data and a driving video consisting of facial keypoints to encode the dynamic components of the video [14], [37]. In this work, we take this approach one-step further by more aggressively compressing the talking-head videos into text, instead of a visual representation such as facial keypoints. We ask the question: “Can we compress AV content generated via webcams to just text and recover videos with similar QoE compared to standard codecs in a low bitrate regime?”

and using recent advances in deep learning based generative models we answer it in the affirmative. The contributions of our work are as follows:

- We propose a novel compression pipeline by compressing audio-video “talking-head” videos to just text. The pipeline uses a state-of-the-art voice cloning model [38] to convert text-to-speech (TTS), and a lip-syncing model [28] to convert audio to reconstructed video using a driving video at the decoder (Figure 2).
- We conducted a subjective study and our results demonstrate that at similar QoE in a low bitrate regime, the pipeline exhibits up to $100 - 1000 \times$ compression advantage over standard audio-video codecs.
- Information theoretically, our results can be viewed as establishing an empirical “achievability” result, showing that a rate of $\sim 100$ bps can yield reconstruction qualities, as assessed by humans, commensurate with what existing codecs would require orders of magnitude higher rates to achieve. In particular, what we achieve is up to two orders of magnitude lower bitrate compared to results reported using facial keypoints [14], [37].

Our pipeline can be used for storing the webcam AV content as a text file or for streaming this content on the fly. We envision the Txt2Vid framework can open up many novel application possibilities such as enabling audio-video communication during poor internet connectivity, or in remote terrains with limited bandwidth, including lunar or Martian space stations. It can be used for storing pedagogical content as text, and using the proposed decoder to learn from one’s favorite instructor or disseminating information in multiple languages through text translation without the need to re-record video content. The code for our pipeline along with some examples and demonstrations is available on GitHub,¹ and the generated dataset for the subjective study is available to download on Google Drive.²

II. COMPRESSION PIPELINE

The proposed pipeline is shown in Figure 2 as a block diagram with focus specifically on video content involving a single person speaking in front of their camera as described in

¹https://github.com/tpulkit/txt2vid.git
²https://tinyurl.com/33aadk6c
Section I. The decoder takes the text transcript of such a video as input and outputs a reconstructed video with audio. This is done by first converting the text component to audio using TTS synthesis resulting in reconstructed audio, followed by speech-to-video (STV) synthesis. STV is done by lip-syncing with the driving video to generate the final reconstructed video with audio output. (B) Expected compression advantage of transmitting text (or audio) over videos compressed using traditional methods.

![Diagram](image)

Fig. 2. Compression pipeline. (A) Block diagram of our codec. Text (or audio) is extracted from the original webcam video and transmitted. An additional driving video of the sender is transmitted once during the lifetime of communication between a particular sender and receiver pair, along with a “User ID”. A generative text-to-speech model is used to recover audio from the text, followed by lip-syncing with the driving video to generate the final reconstructed video with audio output. (B) Expected compression advantage of transmitting text (or audio) over videos compressed using traditional methods.

For performance evaluation (Section III), we modified a pre-trained lip syncing model, Wav2Lip [28] and used the Resemble API [38] for TTS. Our main focus was to demonstrate the capability of the proposed pipeline to drastically reduce the data transmission rate while maintaining QoE. Therefore, for performance evaluation we limited ourselves to the decoding pipeline which generates a video file from a text transcript. However, in practical systems, one may also be interested in streaming applications. To that end, we built an additional prototype to demonstrate how our approach can be used as part of a video streaming system. Details about how the various software tools were utilized and modified for this work are provided in the Appendix A, along with demonstrations of the streaming prototype on our Github page.

As shown in Figure 2B, the pipeline has the potential to achieve extremely low bitrates, up to $100-1000 \times$ smaller than videos compressed using standard codecs. In fact, text can be communicated at $\sim 100$ bps, whereas current codecs cannot achieve such high compression even at extreme settings. The current implementation can lead to loss or alteration of details like facial expressions, tone of voice, and prosody of speech. The QoE can be improved substantially in future versions by incorporating models which also account for these factors, at a cost of relatively small bandwidth increment required for transmitting the additional metadata.

III. PERFORMANCE EVALUATION

In this section, we evaluate the compression gains and reconstruction quality achieved by the proposed pipeline. The evaluation was performed through a subjective study with 242 participants. The study involved comparing videos reconstructed from a text transcript against the original videos encoded using different standard codec parameters. The evaluation focused on understanding the bitrate and QoE achievable...
A total of for A V1. Two different encoding properties of AAC resulted in 4 different encoding properties for H.264 and 3 different ones provided the best quality for the subjective study. We chose for video (audio). We chose the downsampling setting which (BR) along with downsampling by dataset, we tried achieving similar bitrate by varying CRF at the decoder. Therefore, for each reported bitrate in the transmitted and recovered at desired parameters by upsampling. This compressed and downsampled video (audio) can then be followed by using a better quality parameter for compressing. 

bitrates is achievable by first downsampling the video (audio), working with extremely low bitrates, a better quality at similar encoded audio and video. We observed that since we are 

by the pipeline, and hence it did not include the streaming mode or the STT-based encoder. We first discuss the dataset used for evaluation and present details on the subjective study for comparing standard codecs against the proposed approach, followed by the results.

A. Dataset

A dataset was created for the study by recording webcam talking-head videos. Six ∼30s videos were recorded by six different people (different ethnicities; four male, two female) under diverse natural indoor ambient lighting and speaking conditions to serve as the original AV content. Each video consisted of the speaker talking about a different technical topic. These videos were used to create several AVs using standard codecs (benchmark set) and our approach (Txt2Vid set). The training of generative machine learning models, viz. lip-syncing and TTS, used for generating the Txt2Vid set was done using data collected in the wild and did not involve training over the dataset used for the subjective study. The generated dataset is available on Google Drive (link in Section I), and exemplary encodes are shown in Figure 8 (Appendix C).

The benchmark set was generated through ffmpeg using the following steps: 1) convert original video to 720p resolution, 25 fps and yuv420p, 2) encode the video using H.264 or AV1 codec, at a particular encoding parameter, viz. CRF (constant rate factor) and downsampling ratio, 3) convert original audio to sampling rate of 16kHz, 4) encode audio at a constrained bitrate (BR) using AAC codec, 5) merge encoded audio and video. We observed that since we are working with extremely low bitrates, a better quality at similar bitrates is achievable by first downsampling the video (audio), followed by using a better quality parameter for compressing. This compressed and downsampled video (audio) can then be transmitted and recovered at desired parameters by upsampling at the decoder. Therefore, for each reported bitrate in the dataset, we tried achieving similar bitrate by varying CRF (BR) along with downsampling by 1×, 2× and 4× (1×, 2×) for video (audio). We chose the downsampling setting which provided the best quality for the subjective study. We chose 4 different encoding properties for H.264 and 3 different ones for AV1. Two different encoding properties of AAC resulted in a total of 14 different encodings per video content, and a total

of 84 benchmark videos. Obtained audio-video bitrates are shown in Figure 3 and details of the codec parameters used are given in Appendix B. The choice of the encoding parameters was also informed by a small pilot study we conducted before the main study to gauge the appropriate range of bitrates to work with. Diversity in video bitrates at similar CRF across contents as seen in Figure 3 highlights the diversity present in the dataset.

The Txt2Vid set was generated by utilizing a bzip2 compressed text transcript file from the original content [40]. This text transcript was first converted to audio using the voice clone from the Resemble API for each individual in the dataset, followed by using Wav2Lip for lip-syncing with an independent driving video as described in Section II. The driving video was speaker specific but agnostic to the spoken content. The driving video was encoded using H.264 at CRF of 20, 720p resolution, 25 fps and yuv420p. TheTxt2Vid set also contained videos generated using Wav2Lip by directly passing audio through the pipeline instead of the reconstructed audio from Resemble. The audio used was encoded using AAC at ∼10kbps and at a sampling rate of 16kHz. Overall, the Txt2Vid set contains 2 videos per video content resulting in total of 12 Txt2Vid videos.

B. Subjective Study

The subjective study is generated using Qualtrics Platform and conducted using Amazon MTurk. MTurk workers were required to have “masters” qualification and lifetime approval rate greater than 98% to be qualified for our study. For each video content (six recordings in our dataset), we did a pairwise comparison test of all videos in the benchmark set with each video in theTxt2Vid set resulting in 14 × 2 = 28 comparisons per content. Thus, a total of 28 × 6 = 168 pairwise video comparisons were subjectively studied. Since watching many video pairs of 30s each can be time consuming and tiring for a study participant, we decided to show only 28 comparisons to each viewer belonging to one video content (and two additional comparisons for sanity check with obvious audio or video degradation respectively). Each pair was compared by ∼40 viewers leading to a total of 252 participants. Each video in the pair was shown at a resolution of 480p, and the viewers were asked to choose which video in each pair they preferred. We asked a general preference question instead of a specific quality question such as “which video has a higher quality” to account for varying personal preferences amongst viewers across video and audio qualities. We manually verified responses from the participants for ensuring quality of the responses and removed 10 participants from the study who either completed the study too fast (<10min) or failed both the sanity checks.

![Fig. 3. Benchmark Dataset Video Bitrates. Left and right panels show video bitrates for the six contents encoded using H.264 and AV1 as the video codecs. For each content, we encode at 4 different H.264 and 3 different AV1 configurations. Audio for each of these configurations was encoded at two different bitrates using AAC. This results in total of 4×2+3×2 = 14 different benchmark AV per content. Details of the codec parameters used to generate the benchmark dataset are available in Appendix B.](image)

**Table I.**

| Content     | 1    | 2    | 3    | 4    | 5    | 6    |
|-------------|------|------|------|------|------|------|
| Txt2Vid Bitrate bps | 79.20 | 81.33 | 92.27 | 83.47 | 82.07 | 89.6 |

8IRB Number: E-3152.
Our results in next section are reported over the remaining 242 participants. Screenshots from the subjective study test are also available on the Google Drive link provided above.

C. Results and Analysis

1) Achievability of Talking-Head Video Communication at Bitrates as Low as \(\sim 100\) bps: In comparison to the benchmark dataset with bitrates ranging from \(\sim 10-100\) kbps, the proposed Ttxt2Vid method requires only \(\sim 85\) bps on average across all contents evaluated in the study (Table I). This is relatively close to \(\sim 39\) bps information rate empirically estimated for spoken communication in many languages [43] and serves as an empirical lower-bound on the bitrate required for meaningful talking-head video communication. The difference between Ttxt2Vid and spoken language encoding rates can be partly explained by the shorter text segments in the study which are harder to compress using bzip2.

2) Ttxt2Vid Achieves Two-to-Three Orders of Magnitude Compression at Similar QoE Compared to Standard Codecs: Figure 4 shows the results for the subjective study comparing the Ttxt2Vid generated reconstructions (from text) with the standard codecs (AVC and AV1 video codecs shown separately) for all six contents. Recall that we used two audio bitrates (AAC codec) for each video codec setting, both of which are shown in the plots. Note that only the set of videos generated from text using Resemble API are shown here, while the results for the set of videos generated by directly using the original audio are provided in Section III-C.3 (Ablation Study). The plots show the percentage of users preferring the Ttxt2Vid method over a given standard codec against the ratio of bitrates for the standard codec and Ttxt2Vid. As this ratio increases, the reconstruction quality for the standard codec improves and we expect fewer people to prefer Ttxt2Vid. We observe this monotonicity in the plots, along with some outliers to the trend. These outliers can be explained by statistical noise and the fact that we can get different overall qualities at the same bitrate depending on how the total bitrate is allocated to video and audio. The error bars show the 95% Confidence Interval using standard normal distribution assuming each choice can be modeled using a binomial distribution. Focusing on the 50% preference level (dashed line in plot), we observe that for most contents Ttxt2Vid achieves similar QoE with up to 1000\times\) smaller bitrates than the widely used H.264 + AAC codec (Figure 4a). Even when using the state-of-the-art AV1 video codec, we still see close to 200\times lower bitrates using Ttxt2Vid at similar user preference (Figure 4b). These results illustrate the promise of our pipeline in reducing videos to a minimal text-based representation followed by generative reconstruction. Figure 8 in Appendix C shows frames belonging to exemplary Ttxt2Vid and baseline encodes found at similar-QoE using our subjective study for a few different contents in the dataset.

3) Ablation Study: To discern the advantages coming from the whole pipeline against just the lip-syncing, we conducted an ablation study. Instead of using TTS audio, we passed the original audio into the lip-syncing decoder. Figure 5 shows the results for the subjective study comparing the Ttxt2Vid videos generated from original audio (at 10kbps) against the standard codecs. The overall trend remains similar to Figure 4 but the reduction in bitrate for Ttxt2Vid over standard codecs at the 50% preference level is much smaller. We achieve \(\sim 5\times\) reduction over AVC and \(\sim 2x\) reduction over AV1. The audio bitrates are comparable to video bitrates in the low bitrate regime, and hence we do not save as much by just transmitting the audio. An attempt to reduce the audio bitrate further in our study resulted in discernible artifacts in the encoded audio, resulting in much worse QoE.

4) Inference of Quality Scores From Subjective Study: To further elucidate our results, we modeled the obtained QoE results with the widely-used Bradley-Terry-Luce (BTL)
Fig. 5. Ablation Study: subjective study results comparingTxt2Vid videos generated from original audio to standard codecs. Each plot shows the percentage of users preferring theTxt2Vid video over the standard codec vs. the ratio of the bitrate between the standard codec andTxt2Vid. The dashed horizontal line represents the 50% preference levels where the two methods are equally preferred by the participants. The error bars show 95% confidence interval.

model [42], [44], [45]. BTL converts the obtained paired comparison results to an intrinsic quality score for each video on an interval-scale. BTL uses a logistic regression model: assume $Q_i$ is the intrinsic quality score of video $i$ and $P_{i,j}$ is the probability assigned by model that a user would prefer video $i$ over video $j$, then

$$P_{i,j} = \frac{e^{Q_i - Q_j}}{1 + e^{Q_i - Q_j}}$$  \hspace{1cm} (1)$$

Let $\text{frac}(i > j)$ be the fraction of users preferring video $i$ over video $j$ in the observed data, then the quality scores $Q_i$ for each video can be inferred by maximizing the likelihood of $\text{frac}(i > j)$ given model probabilities $P_{i,j}$ through an optimization procedure such as the Newton-Raphson method.

Figure 6 shows the obtained Quality Scores using BTL model against the ratio of bitrates for the standard codec andTxt2Vid, for each content separately. These quality scores were extracted by doing MLE over all paired comparisons across 16 videos used in our subjective study (per content): [(4 AVC + 3 AV1) × (2 AAC)] + [Txt2Vid (from Text) +Txt2Vid (from Original Audio)]. Since the logistic model stays the same under addition of a constant over the quality scores, we report all quality scores with respect toTxt2Vid generated video (shown as a red star in the figure). Thus, a positive quality score implies a higher inferred quality of the video compared toTxtVid, and vice-versa. As the ratio of standard codec toTxt2Vid bitrate increases, we expect the reconstruction quality and obtained quality scores to be higher and we see that in the plots. Figure 6 shows that ourTxt2Vid approach provides two-to-three order of magnitude compression at similar quality scores.

Surprisingly, we found that for most of the contents, Txt2Vid videos reconstructed from text using generated audio had higher quality scores than Txt2Vid videos reconstructed using original audio. This shows that at extremely low bitrates, audio quality likely starts dominating the user’s experience. However, even at a few kbps, compressed audio quality for spoken content is relatively poor. TTS alleviates this issue by regenerating high quality audio samples at the decoder, without requiring transmission of original audio. Thus, the orders of magnitude advantage we report in this work is coming from generative modeling of both video and audio together.

5) Analysis Across Content: We note that there is some variation between the results for the different contents which is expected due to the diversity in the background and lighting conditions, e.g. for some contents we observed that the lip synced region was visible as a rectangular artifact making the video appear unnatural, as well as voice cloning quality. In particular, the preference forTxt2Vid is lower for content 2.  

Fig. 6. Quality Scores using BTL model. Each plot shows the inferred quality scores on an interval-scale inferred by modeling complete paired comparison data using probabilistic Bradley-Terry-Luce model [42], against the ratio of the bitrate between the standard codec andTxt2Vid. Txt2Vid videos (red star) serve as reference with quality score of 0, and a positive score implies a higher inferred quality for the corresponding video. This demonstrates potential of the proposedTxt2Vid approach to achieve two-to-three order of compression over standard codecs. The color bar for the benchmark videos shows the fraction of total AV bitrate spent on the video component.
IV. DISCUSSION

A. Applications

We believe that our setup can enable several applications with positive societal impact as shown in Figure 7. Due to the extremely low bandwidth requirements, this can allow people in areas with poor internet availability, high costs and limited access, including those living in remote areas and in the developing world, to be better connected and get a good audio-video experience. This pipeline is particularly suited for transmission of pedagogical content, which is topical given the growth in remote learning and online instructional videos. Since the pipeline only transmits a text transcript, it also opens up the possibility of using different voices and faces to help students feel more engaged with the content. One can imagine Albert Einstein teaching relativity or a child’s favorite movie character teaching them math. Recent studies have shown that people retain the information better when they learn from their favorite instructor [46]. Given advances in machine translation, this system can also be easily modified to display videos in multiple languages without the need to create and store multiple original versions.

Even for a normal video calling application, this work opens up many possibilities. A user can speak as usual and the communication can occur via the transcribed text or the original audio. The reconstructed video seen by the other users can potentially use any face or voice, no longer subject to the various implicit biases. A user caught up in some other task can simply type in what they wish to say, without affecting the audio-video experience for the other users. We plan to further investigate these applications.

B. Limitations and Future Work

1) Computational Complexity: The proposed pipeline is currently a prototype to demonstrate the advantages of a generative approach. More work is needed to enable widespread use in daily life especially in the context of streaming. The high computational complexity of lip syncing and the requirement for GPUs is a bottleneck for using this system on low-end devices which are more likely to be found in regions with poor connectivity. In addition, the current reliance on cloud-based APIs for TTS and STT is not well-aligned with the broader aim of reducing bandwidth usage. But, with existing high investment in hardware to accelerate deep learning models, reduction in GPU costs, significant research in edge-computing, and more efficient open-source models, we envision that these limitations will be considerably alleviated in the near future. As part of future work, we hope to build a standalone application on desktop and mobile devices to make this system widely accessible.

We will take several complementary approaches to reduce the model complexity and size. Our first approach would be to reduce the frame rate of the generated videos as perceptually required for communicating information in the talking-head videos. All prior works focus on video generation at 25 frames per second (fps) since it is a standard in the video community. But in our experience, a far lower frame rate of 10 fps is sufficient to capture most of the talking-head AV experience. Reducing the number of frames needed by the decoder to generate output in real-time has the potential to significantly reduce the decoder computational complexity.

(and content 5 for the case of Txt2Vid videos generated from original audio). Note that a non-negligible fraction of participants still prefer Txt2Vid over the standard codecs which require orders of magnitude higher bitrate. To better understand the performance variation, we analyzed the comments from participants, and found that some of them listed audio quality as the primary determinant for their preferences. For most contents, the audio produced by Resemble has much better quality as compared to the benchmark set (which had audio with relatively low bitrates). However, in certain cases, the Resemble audio sounded unrealistic, robotic and/or lacked clarity. This was also observed in a comparison between Txt2Vid with Resemble audio and Txt2Vid with original audio (at 10 kbps bitrate), where only 30% participants preferred the Resemble audio for content 2 (in contrast to more than 50% preference for most other contents). We believe that this is an artifact of the quality of training data and the Resemble training process, and we expect things to improve as voice cloning technology progresses. For the other contents, this highlights a major advantage of the proposed pipeline - the ability to obtain high quality audio at extremely low bitrates using voice cloning and TTS.
A second approach will focus on model compression via model pruning [47] and teacher-student learning [48] or knowledge distillation [49], which will enable faster inference time by reducing the number of parameters in the model, allowing us to run the pipeline on smartphones. To benchmark the trade-off between model complexity and quality-bitrate, we will conduct ablation studies to identify the complexity stress points in our approach.

2) Latency: The latency of the streaming pipeline needs to be reduced to enable real-time interactive applications such as video calling. As described in Section A-C, the streaming system currently is a proof-of-concept rather than a production-level system. The current end-to-end latency is close to 4 seconds, largely due to the buffering at ffmpeg. We believe that this can be reduced by using a real-time protocol, and we plan to explore this in future.

3) Quality-of-Experience: In addition to the above technical limitations, there is scope for improvement in the encoding and reconstruction quality. We found that the STT occasionally incorrectly transcribes certain words, especially technical words, and requires manual proofreading. Furthermore, the system by design is restricted to communicating the transcript, and thus can alter verbal content (e.g., tone of voice, prosody of speech) or miss non-verbal cues (e.g., head nods, eyebrow raises), non-speech sounds (e.g., laughter), and facial expressions. We note that this is not a fundamental limitation and one can envision a system that transmits additional metadata along with the transcript to capture these aspects, coupled with a decoder capable of incorporating them into the reconstruction. Specifically, one can learn personalized metadata corresponding to speech rhythm, head pose and body language using small pre-recorded video clips of individuals (e.g. see Refs. [50], [51]), and emotions from analyzing the text transcript. This metadata will be aligned with the target identity at the decoder to generate high quality personalized talking videos. This would involve training newer models that integrate these non-verbal communication features and evaluating the approach on more datasets, which is left as future work.

4) Ethical and Privacy Considerations: Finally, we note that a pipeline like ours raises some privacy concerns and has a potential for misuse typically associated with Deepfakes [52]. For example, transmitting video as text allows for any face or voice to be used at the decoder potentially allowing misportrayal of an individual’s identity. A mechanism to limit the usage of generative models to the duration of the call could potentially abate the misuse. One way to do so would be to integrate a security mechanism (such as an encrypted key) at the receiver which only allows generating content using the sender’s identity when explicitly allowed by the sender, or limited to the duration of the call. New ways to distinguish actual audio/video from the generated audio/video, both at a human and computational level are required so that users at all times are aware they are interacting with generated video. In summary, widespread usage of this technology requires cooperation between governments, industry and academia to develop safeguarding mechanisms and address these challenges at legal, technical and societal levels [53].

V. Conclusion

In this work, we presented a novel video compression pipeline Tvx2Vid, for extreme compression of talking-head videos as seen in video conferencing and webinars. Tvx2Vid minimizes the data transmission rate by reducing the video to a text transcript, followed by a realistic reconstruction of the video using recent advances in deep learning based voice cloning and lip syncing models. We implemented a prototype using Resemble voice cloning and Wav2Lip lip-syncing frameworks. We also demonstrated a proof-of-concept streaming pipeline that can potentially enable real-time applications in the future. We evaluated our pipeline using a subjective study on Amazon Mturk to compare user preferences between Tvx2Vid generated videos and videos compressed with standard codecs at varying levels of compression. In the study, performed on multiple video contents, our proposed pipeline achieved two to three orders of magnitude lower bitrates than state-of-the-art audio-video codecs at comparable Quality-of-Experience. The proposed framework can enable several applications with great potential for social good, expanding the reach of video communication technology. While we used specific tools in our pipeline to demonstrate its capabilities, we envision significant progress in the components used over the coming years leading to even better reconstruction quality.

Appendix A

Software Tools

A. Lip-Syncing Model

For STV synthesis, we focus on models for generating realistic talking-head videos from audio. These models are built on the observation that realistic talking-heads can be generated from audio by mostly capturing the mouth movement in the video. These models can be categorized into, (a) audio-driven 3D facial animation [54], [55], [56], [57], [58], [59], or (b) lip-synced talking-head videos [28], [29], [34], [60], [61], [62], [63], [64]. 3D models generate the whole frame by first generating an audio-driven 3D model of an individual followed by a 2D projection into a frame. Lip-syncing talking-head videos simplify the problem further by not generating the whole frame from just the audio. They utilize supplementary information from a driving frame or video. This driving frame (or video) is used as a prior for the model generating the synced lip-movements with the audio. This generation method reduces the computational complexity by only generating the region around the lip, and provides the best output quality as a lot of visual information is already contained in the driving frame (or video). We work with a pre-trained lip-syncing model – Wav2Lip [28].

Wav2Lip takes as input the driving video (or frame) and an audio clip, and outputs a lip-synced video or frame with the audio. In our case, the driving video is a short silent clip (~30s) of the person facing the webcam with natural head/eye movements. Wav2Lip relies on a generative adversarial network (GAN) trained with the help of a pre-trained discriminator that can detect lip-syncing errors. During inference,
Wav2Lip first identifies the face and lip region in the driving video frames, and then uses the trained generator model to reconstruct this region. It has been shown to work well with dynamic and unconstrained talking-head videos. However, the existing implementation requires the complete audio for generating a lip-synced video which does not work for our pipeline as the decompressor needs to decode the stored or streamed text files on the fly. We modified the code from Wav2Lip GitHub repository\(^4\) to enable the generation of lip-synced videos in a streaming manner. This was done by chunking the audio stream used to generate videos and ensuring a batch size of one frame during inference for frame-by-frame generation. Some caveats of using the existing Wav2Lip pre-trained model include, (a) the audio chunk size has to be greater than 200 ms because of model architecture leading to a fixed minimum latency, and (b) the model works well only with single person driving videos with frame resolution of 720p and the person directly facing the camera. These drawbacks are a consequence of using the Wav2Lip model which is trained on a specific dataset. They can be overcome by training a new lip-sync model from scratch on different datasets depending on the application use cases.

### B. Text-to-Speech

As compared to STV, TTS is a relatively mature technology with many commercial vendors providing an API such as Google [65], Microsoft [66], Resemble [38], and Descript [67]. TTS technologies use machine learning based generative modeling to create realistic audio samples from text. They allow for the generation of voice samples in an individual’s voice (voice cloning) to more closely represent the original person. Thus, using currently available TTS technology provides a timely and unique opportunity for the proposed compression pipeline. In this work, we use Resemble API [38] to generate natural voice clones of individuals and use them during inference for decoding. Training a voice clone on Resemble is fast and easy. It requires recording a minimum of 50 predefined voice samples on their website. Our current prototype relies on a web-based API for TTS leading to additional communication overhead for sending and receiving the text script and the generated audio, respectively. But this is a constraint due to the presently available TTS systems, and we do not include this as part of the transmission rate since an actual production-level system could perform TTS locally [68].

### C. Streaming Prototype

Our main focus in this work was to demonstrate the capability of the proposed pipeline to drastically reduce the transmission rate while maintaining good QoE. Therefore, for performance evaluation (Section III) we limited ourselves to the decoding pipeline which generates a video file from a text transcript. However, in practical systems, we may also be interested in streaming applications. These applications typically have a range of latency requirements depending on whether real-time interaction is desired. In addition, real-time video playback requires real-time decoding to prevent degrading the user’s experience. Therefore, we built an additional prototype to demonstrate how our approach can be used as part of a streaming pipeline.

At the encoder, we used Google STT streaming API [69] to convert the speech audio to text in real-time. The API returns finalized text for the preceding speech segment after it detects a pause in spoken content. This text is then transmitted to the receiver using websockets. One limitation of the current implementation is that the latency varies depending on the length of the sentence (≈1-10s), which can sometimes lead to unexpected silence on the receiver’s end. The variability in latency can be managed with a buffer on the receiver though at the cost of increased latency. At the decoder, we receive text from the websocket or read it from a file, and send it to Resemble using their API (one paragraph at a time). A separate callback server receives the generated audio and sends it to the processing script through a pipe. The voice cloned audio is split into 200ms chunks (see Appendix A) and passed to the Wav2Lip generator. Finally, the generated frames and the audio are combined using ffmpeg and sent to an HTTP port for video playback using fffmpeg or VLC. The receiver can also accept audio directly from the websocket or from an audio file, in which case the TTS part is no longer needed and is skipped. We used multithreading and pipes in Python to build this system.\(^5\)

### Appendix B

#### Dataset Generation

Table II enlists all the parameters used for the generation of 14 different encodes, referred to as benchmark set in Section III-A. Same parameters were used for all video contents. Codec-V (Codec-A), CRF, DS-V (DS-A), BR and Avg. Bitrate refer to the video (audio) codec, Constrained Rate Factor for video encoding, video (audio) downsampling, constrained bitrate for audio encoding, and average combined audio and video bitrate across all contents respectively. A higher CRF and higher video downsampling implies a lower

\(^1\)https://github.com/Rudrabha/Wav2Lip

\(^5\)Demonstration videos along with detailed instructions are available on GitHub.
video quality at lower bitrate. A lower BR and higher audio downsampling implies a lower audio quality at lower bitrate. The choice of parameters was driven by an attempt to get the best subjective quality at a given total bitrate.

APPENDIX C
EXEMPLARY ENCODES

Figure 8 shows frames belonging to few exemplary encodes for different contents in the dataset obtained at similar-QoE.

ACKNOWLEDGMENT

The authors would like to thank all the participants of their subjective user study.

REFERENCES

[1] Cisco. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2017–2022. Accessed: 2021. [Online]. Available: https://s3.amazonaws.com/media.mediapost.com/uploads/CiscoForecast.pdf

[2] R. De’, N. Pandey, and A. Pal, “Impact of digital surge during covid-19 pandemic: A viewpoint on research and practice,” Int. J. Inf. Manage., vol. 55, Dec. 2020, Art. no. 102171.

[3] Cisco. Cisco Annual Internet Report (2018–2023) White Paper. Accessed: 2021. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

[4] M. Candela, V. Luconi, and A. Vecchio, “Impact of the COVID-19 pandemic on the internet latency: A large-scale study,” Comput. Netw., vol. 182, Dec. 2020, Art. no. 107495.

[5] G. S. Ford, “COVID-19 and broadband speeds: A multi-country analysis,” 2020.

[6] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, “Overview of the H. 264/AVC video coding standard,” IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 7, pp. 560–576, 2003.

[7] Y. Chen et al., “An overview of core coding tools in the AV1 video codec,” in Proc. Picture Coding Symp. (PCS), Jun. 2018, pp. 41–45.

[8] K. Brandenburg, “MP3 and AAC explained,” in Proc. Audio Eng. Soc. Conf., 17th Int. Conf., High-Quality Audio Coding, 1999.

[9] Y. Chen, K. Wu, and Q. Zhang, “From QoS to QoE: A tutorial on video quality assessment,” IEEE Commun. Surveys Tuts., vol. 17, no. 2, pp. 1126–1165, 2nd Quart., 2014.

[10] J. Bao et al., “Towards a theory of semantic communication,” in Proc. IEEE Netw. Sci. Workshop, Jun. 2011, pp. 110–117.

[11] H. Xie, Z. Qin, G. Y. Li, and B.-H. Juang, “Deep learning enabled semantic communication systems,” IEEE Trans. Signal Process., vol. 69, pp. 2663–2675, 2021.

[12] Z. Weng and Z. Qin, “Semantic communication systems for speech transmission,” IEEE J. Sel. Areas Commun., vol. 39, no. 8, pp. 2434–2444, Aug. 2021.

[13] H. Xie and Z. Qin, “A lite distributed semantic communication system for Internet of Things,” IEEE J. Sel. Areas Commun., vol. 39, no. 1, pp. 142–153, Jan. 2021.

[14] T.-C. Wang, A. Mallya, and M.-Y. Liu, “One-shot free-view neural talking-head synthesis for video conferencing,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2021, pp. 10039–10049.

[15] Z. Qin, X. Tao, J. Lu, W. Tong, and G. Ye Li, “Semantic communications: Principles and challenges,” 2021, arXiv:2201.01389.
Pulkit Tandon received the B.Tech. degree in electrical engineering from IIT Bombay and the master’s degree in electrical engineering from Stanford University, where he is currently pursuing the Ph.D. degree with the Electrical Engineering Department. He is also a Raminus Stanford Graduate Fellow with Stanford University. His research interests include the intersection of video compression, neuro-engineering, perceptual engineering and brain-machine-interfaces with components of algorithms, and hardware and human perception.

Shuhham Chandak received the B.Tech. degree in electrical engineering from IIT Bombay, India, and the Ph.D. degree in electrical engineering from Stanford University advised by Prof. Tsachy Weissman. He is currently an Applied Scientist at Amazon AWS S3. He has mainly worked on data compression for various domains, including genomic data and on error correction coding especially in the context of DNA-based data storage.

Pat Pataramutaporn (Graduate Student Member, IEEE) is currently pursuing the Ph.D. degree with the MIT Media Laboratory, Fluid Interfaces Research Group. He is also a Technologist and a Researcher at the Massachusetts Institute of Technology (MIT), where he explores the intersection of synthetic virtual humans and synthetic biology, specifically at the interface between biological and digital systems. His interdisciplinary research ranges from investigating AI-generated characters for learning and well-being, human-AI co-reasoning, wearable lab on the body with programmable bio-digital organ for space exploration, and mind-controlled 3D printer.

Yimeng Liu is currently pursuing the Ph.D. degree with the Computer Science Department, UC Santa Barbara. With her background in computer vision, machine learning, human-computer interaction, and brain-computer interface, she is enthusiastic about building machine learning enabled applications to enhance human abilities. She has worked on research topics, including 3D computer vision, human motion, and EEG, and has published papers in ACM UIST, ACM SUI, and Frontiers in Computational Neuroscience.

Anesu M. Mapuranga is currently pursuing the bachelor’s degree in electrical engineering with Stanford University. His research interests range from data compression and signal processing to medical device technology.

Pattie Maes received the bachelor’s degree in computer science and the Ph.D. degree in artificial intelligence from Vrije Universiteit Brussel, Belgium. She is currently a Professor at MIT’s Program in Media Arts and Sciences, where she runs the Media Laboratories Fluid Interfaces Research Group, which does research at the intersection of human-computer interaction and artificial intelligence with a focus on applications in health, wellbeing, and learning. She is the editor of three books and an editorial board member and a reviewer for numerous professional journals and conferences. She has received several awards and, in addition to her academic endeavors, she has been an active entrepreneur as the co-founder of several venture-backed companies. Prior to joining the Media Laboratory, she was a Visiting Professor and a Research Scientist at the MIT Artificial Intelligence Laboratory.

Tsachy Weissman (Fellow, IEEE) has been a Faculty Member of the Electrical Engineering Department, Stanford University, since 2003, researching and teaching the science of information, with applications spanning genomics, neuroscience, and technology. He has been serving on editorial boards for scientific journals, technical advisory boards in industry, and as the Founding Director of the Stanford Compression Forum. His recent projects include the STEM2SHTEM science and humanities summer internship program for high schoolers, the Starling initiative for data integrity and Stagecast, and a low-latency video platform allowing actors and singers to perform together in real-time while geographically distributed due to the pandemic. He has received multiple awards for his research and teaching, including best paper awards from the IEEE Information Theory and Communications Societies, while his students received best student authored paper awards at the top conferences of their areas of scholarship. He has prototyped Guardant Health’s first algorithms for early detection of cancer from blood tests, and has more recently co-founded and sold Compressable to Amazon, reducing humanity’s cloud storage carbon footprint. His favorite gig to date was advising the HBO show “Silicon Valley.”

Misha Sra received the Ph.D. degree from the MIT Media Laboratory in 2018. She is currently the John and Eileen Gengross Assistant Professor and directs the Perceptual Engineering Laboratory, Computer Science Department, UCSB. She is affiliated with the UCSB’s Center for Responsible Machine Learning (CRML), Mind & Machine Intelligence and the Cognitive Science Program. She has published at the most selective HCI and VR venues, such as CHI, UIST, VRST, and DIS, where she received multiple best paper awards and honorable mentions. Her laboratory works on designing and engineering new AR/VR, hardware, and AI technologies that can augment our physiological and perceptual capabilities.