May 2023

Compressing Presentation-Dominated Video Files

Ankur Goel

Compressing Presentation-Dominated Video Files

A video conferencing platform can enable video-based conferences between multiple participants via respective client devices that are connected over a network and share each other's audio (e.g., voice of a user recorded via a microphone of a client device) and/or video streams (e.g., a video captured by a camera of a client device) during a video conference. A video conferencing platform may maintain a recording (e.g., responsive to receiving a request from a client device) of a video conference for future reference (e.g., for an invitee who could not join the conference, for future attendees, etc.). The video conference platform may store (e.g., at a client device, a server, etc.) recorded video files associated with the video conference for later retrieval. Recorded video files may be maintained in a server and accumulate overtime to become quite large, resulting in a large storage cost for an organization. Accordingly, many video conferencing platforms utilize various compression algorithms to compress video files.

Some conventional compression algorithms, such as conventional lossy compression algorithms, may result in a lower quality video file than the original video because the compression process involves removing some information from the video in order to reduce its size. In some instances, the removed information may be redundant or less important to an overall visual quality of the video, but its removal can still lead to a loss of detail and a reduction in visual fidelity when uncompressed. Accordingly, viewers of the recreated video file may be unable to ascertain details of text and diagrams that may be essential to understanding content of the video file.

Other conventional compression algorithms, such as conventional lossless compression algorithms, may allow for a compressed video file to be recreated to its original, uncompressed

form without any loss of information or quality. However, as the compressed data still contains all of the original information and results from compressing each individual frame of a video file, lossless compression typically results in larger file sizes than lossy compression. As such, lossless compression may be impractical for video compression when storage overhead is a concern. Accordingly, these conventional methods do not solve the concern of maintaining a high quality, compressed video file with little memory overhead.

Therefore, a technique is proposed for compressing presentation-dominated video files that may be integrated into a video conferencing platform. A participant of a video conference can speak (e.g., present on a topic) to the other participants of the video conference. Some existing video conferencing platforms can provide a user interface (UI) to each client device connected to the video conference, where the UI displays the video streams shared over the network in a set of regions in the UI. For example, a participant can share a presentation to demonstrate a product or an activity presently being discussed by other participants of the video conference. In many instances, the presentation can be a set of displays that remain static for many frames (e.g., 1,000s of frames) of the video stream. For example, the video may include a slide show presentation where each slide is presented to participants of the video conference for thousands of frames. The presentation can include various types of documents that remain static for multiple frames, such as a slide presentation, a text document, a spreadsheet, or any other electronic document (e.g., an electronic document including text, tables, images, graphs, slides, charts, software programming code, designs, lists, plans, blueprints, maps, etc.).

The disclosed technique can be implemented as processing logic that may be integrated into a video conferencing platform. The processing logic may receive a video file including multiple frames, an audio, and multiple presentation slides associated with a video file. The

video file may be a video recording of a participant of a video conference presenting the presentation slides to other participants of the video conference. The presentation slides may include various types of documents that remain static over multiple frames of the video file, as described above.

The processing logic may generate a mapping between each of the multiple presentation slides to one or more of the multiple frames of the video file based on a similarity level between a given presentation slide and a given frame. For example, it may be determined that a first presentation slide of the multiple presentation slides is highly similar to frames 1 through 3,000 of the video file. Accordingly, the processing logic may determine that the first presentation slide corresponds to a presentation slide displayed from frame 1 to frame 3,000 of the video file and generate a mapping between the first presentation slide and frames 1 through 3,000. It may be further determined that a second presentation slide of the multiple presentation slides is highly similar to frames 3,001 through 5,500 of the video file. Accordingly, the processing logic may determine that the second presentation slide corresponds to a presentation slide displayed from frames 3,001 through 5,500 of the video file and generate a mapping between the second presentation slide and frames 3,001 through 5,500. In some embodiments, the level of similarity between presentation slides and frames may be predicted by a machine learning model that is trained to determine, based on a dataset of historical images that have been labeled with similarity scores, similarity between two or more images (e.g., slides of the presentation and frames of the video file), as described in detail below with respect to figure 1.

The processing logic may organize the mapping between presentation slides and frames in a data structure (also referred to as a “frame-slide mapping data structure” herein). For example, the mapping may be organized in an array where each index of the array corresponds to

a frame of the video file and each value stored at indices of the array corresponds to an associated presentation slide of the respective index. The processing logic may utilize one or more compression algorithms to compress the presentation slides, the audio, and the frame-slide mapping data structure. For example, the presentation slides, the audio, and the frame-slide mapping data structure may be compressed and stored as a single file using a file compression system. The file compression system, for example, may use a lossless compression algorithm (e.g., deflate, WavPack, PPMd, etc.) to generate the file. By compressing static elements (e.g., slides) of a presentation within a video file rather than compressing each frame of the video file, the disclosed technique can generate a compressed version of the video file with little storage cost (e.g., 12 MegaBytes (MBs)) compared to conventional techniques (e.g., 33.5MBs).

The description uses a video conferencing platform as an example platform in which the disclosed compression techniques can be applied. However, it is appreciated that the proposed technique can be applied to various other types of platforms such as a content sharing platform, a social network platform, etc. The description also uses presentation slides as an example video file in which the disclosed compression techniques can be applied. However, it is appreciated that proposed compression techniques can be applied to any video file that includes various types of documents that remain static for multiple frames, such as a slide presentation, a text document, a spreadsheet, or any suitable electronic document (e.g., an electronic document including text, tables, images, graphs, slides, charts, software programming code, designs, lists, plans, blueprints, maps, etc.).

Figure 1 illustrates a flow diagram of a method for compressing presentation-dominated video files of a video conferencing platform. At block 100, processing logic may receive, from the video conferencing platform, a video file including multiple frames and audio (e.g., an audio

channel). In some instances, the processing logic may separate the audio (e.g., the audio channel) from the video. At block 110, the processing logic may identify multiple presentation slides associated with the video file. In one example, the video file may be a recording of a participant of a video conference presenting the presentation slides to other participants of the video conference. The presentation slides may be provided (e.g., via a user interface (UI)) by the presenter to the video conference platform. The presentation slides may be prepared by the presenter to aid the presenter in delivering content and running the meeting. For example, for a first portion of the video the presenter may present a first slide to participants of the video conference, and for a second portion of the video the presenter may present a second slide to the participants, and so forth. Accordingly, each presentation slide may be presented within the video file for one or more frames corresponding to when and how long the respective presentation slide was presented to participants of the video conference.

At block 110, the processing logic may generate a mapping of each of the multiple presentation slides to one or more of the multiple frames based on a similarity level between a given presentation slide and a given frame. In one example, the similarity level between a given presentation slide and a given frame may be determined based on one or more outputs of a machine learning model. In some instances, a training engine can train the machine learning model using training data that includes training inputs and corresponding target outputs (correct answers for respective training inputs). For example, the training data may include historical data (e.g., previous slides, historical images, etc.) to predict a similarity level between images. The training engine may find patterns in the training data that map the training input to the target output (the similarity level to be predicted), and train the machine learning model according to these patterns. The machine learning model can be composed of, e.g., multiple levels of linear

and non-linear operations (e.g., a convolution neural network (CNN) deep network). An example of a deep network is a neural network with one or more hidden layers, and such a machine learning model can be trained by, for example, adjusting weights of a neural network in accordance with a backpropagation learning algorithm or the like. Once the model is trained, it can be used to compare new images (e.g., frames of the video file, presentation slides, etc.) to images in the training dataset to determine similarity. A similarity level, for example, can be determined by passing the frames and presentations slides through the CNN, extracting visual features, and comparing the visual features using a similarity metric such as a cosine similarity or a Euclidean distance. Based on the output of the machine learning model, the processing logic can generate a mapping of each of the multiple presentation slides to one or more of the multiple frames. It is appreciated that similarity between presentation slides and frames may be determined based on other methods. For example, the processing logic may determine similarity by utilizing a mean squared error (MSE) function to calculate the average square distance between pixels of two images or the like.

In an illustrative example, it may be determined (e.g., using the machine learning model) that a first presentation slide of the multiple presentation slides is highly similar to frames 1 through 3,000 of the video file. Accordingly, the processing logic may determine that the first presentation slide corresponds to a presentation slide displayed from frame 1 to frame 3,000 of the video file and generate a mapping between the first presentation slide and frames 1 through 3,000. It may be further determined that a second presentation slide of the multiple presentation slides is highly similar to frames 3,001 through 5,500 of the video file. Accordingly, the processing logic may determine that the second presentation slide corresponds to a presentation

slide displayed from frames 3,001 through 5,500 of the video file and generate a mapping between the second presentation slide and frames 3,001 through 5,500.

At block 130, the processing logic may organize the mapping of presentation slides to frames in a data structure (also referred to as a "frame-slide mapping data structure" herein). For example, the mapping may be organized in an array where each index of the array corresponds to a frame of the video file and each value stored at indices of the array corresponds to an associated presentation slide of the respective index. In an illustrative example, indices 0-2999 of the array may correspond to frames 1-3000 of the video file. A first presentation slide may be displayed from frames 1-3,000 of the video file. Accordingly, indices 0-2,999 of the array may be updated with a value of "1" to indicate indices 0-2,999 are mapped to the first presentation slide. In another example, indices 3,000-5,499 of the array may correspond to frames 3,001-5,500 of the video file. A second presentation slide may be displayed from frames 3,001-5,500 of the video file. Accordingly, indices 3,000-5,499 of the array may be updated with a value of "2" to indicate indices 3,000-5,499 are mapped to the first presentation slide.

In some instances, the processing logic may additionally perform run-length encoding (RLE) on the above-described array. RLE can be used to compress a string of data (e.g., numbers, characters, etc.). In RLE, consecutive identical values are replaced with a single value and a count of the number of times consecutive identical values occur. RLE can be applied to the array, where runs of values corresponding to the same frame are replaced with a single number and a count, where the single number corresponds to a given frame and the count corresponds to a number of indices corresponding to the given frame. For example, the array with 3000 indices corresponding to the first frame and a subsequent 2,500 indices corresponding to the second frame can be encoded to "(1,3000)(2,2500)" using RLE.

At block 140, the processing logic may compress the multiple presentation slides, the audio (e.g., the audio), and the data structure to obtain a compressed file. In some instances, the processing logic may compress the run-length encoding of the data structure (e.g., the above-described array) in lieu of the data structure. The processing logic may utilize one or more compression algorithms to compress the presentation slides, the audio, and the data structure containing the mapping between presentation slides and frames of the video file. For example, the presentation slides, the audio, and the data structure may be compressed and stored as a single file using a file compression system. The file compression system may use a lossless compression algorithm (e.g., deflate, WavPack, PPMd, etc.) to generate the file. In some instances, the audio can be compressed prior to being stored as a single file with the mapping and the audio. For example, the audio may be compressed using lossy audio compression formats such as MP3, AAC, WMA, and the like. In another example, the audio may be compressed using lossless audio compression formats such as FLAC, ALAC, and the like. In some instances, the presentation slides may be compressed as an image, a Portable Document Format (PDF), or similar text and/or image format. The video file can later be recreated using the audio, the presentation slides, and the RLE of the presentation slide numbers corresponding to each frame of the video file.

By compressing static elements (e.g., slides) of a presentation within a video file rather than compressing each frame of the video file, the disclosed technique can generate a compressed version of the presentation with little storage cost compared to conventional techniques and improve quality for static contents (e.g. text, diagrams, sheets, etc.) of the recreated video.

ABSTRACT

A technique is proposed for efficiently compressing presentation-dominated video files. Processing logic may receive, from a video conferencing platform, a video file including multiple frames and audio. The processing logic further may identify multiple presentation slides associated with the video file. In some instances, to identify the multiple presentation slides, the processing logic may receive the presentation slides from one of the video conferencing platform or a user of the video conferencing platform (e.g., via a client device). The processing logic may further generate a mapping of each of the multiple presentation slides to one or more of the multiple frames based on a similarity level between a given presentation slide and a given frame and organize the mapping in a data structure (frame-slide mapping data structure). This similarity level may be determined using one or more machine learning models. The processing logic may further compress the multiple presentation slides, the audio, and the frame-slide mapping data structure to obtain a compressed file. The video can be recreated/uncompressed by inserting the presentation slides into the video frames as determined by frame-slide mapping data structure and adding the audio (e.g., the audio channel) to the generated video. This results in less storage cost compared to conventional compression algorithms and better video quality for static contents (e.g. text, diagrams, sheets, etc.) of the recreated video.

Keywords: virtual meeting, video conference, video compression, video decompression, presentation

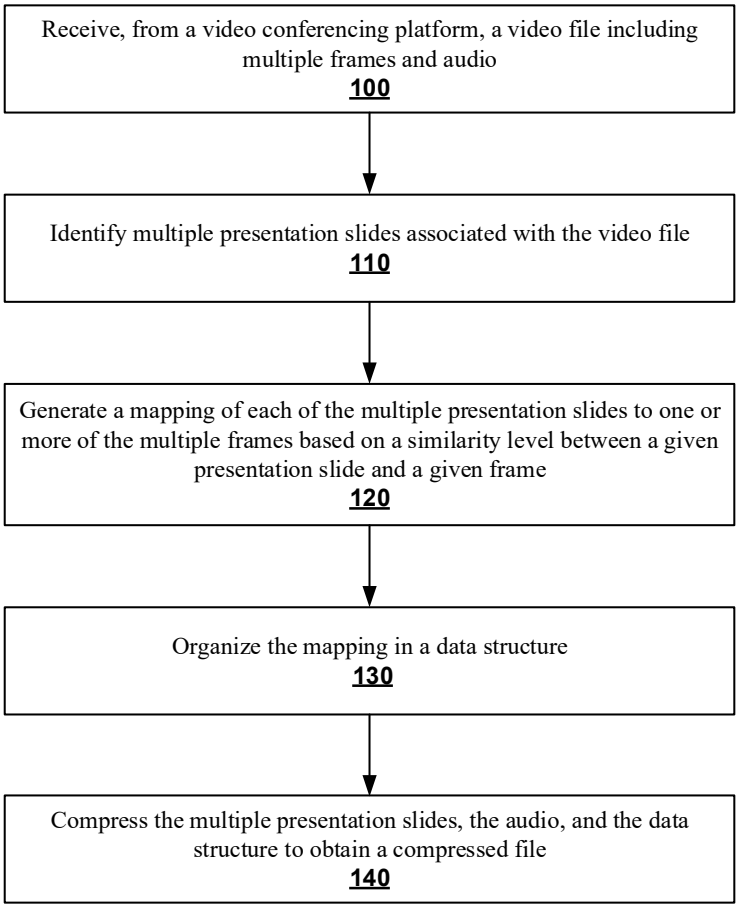
Receive, from a video conferencing platform, a video file including multiple frames and audio
100
Identify multiple presentation slides associated with the video file
110
Generate a mapping of each of the multiple presentation slides to one or more of the multiple frames based on a similarity level between a given presentation slide and a given frame
120
Organize the mapping in a data structure
130
Compress the multiple presentation slides, the audio, and the data structure to obtain a compressed file
140

FIG. 1