MLPEG: Using Machine Learning for Data Compression

Statement of Problem

How can machine learning be used to compress video data?

Introduction

Video streaming accounts for over 80 percent of all online traffic, therefore an effective video compression algorithm is paramount. Reducing the file sizes of videos can broaden the availability of video on devices ranging from desktops to embedded systems and lower the overhead cost of long-term video storage. MLPEG is a compression algorithm and video codec—powered my machine learning—that aims to have a smaller footprint that H.264, the most common algorithm for video compression.

MLPEG is a lossy compression algorithm, meaning that some data is irreversibly discarded when compressing. This data is regenerated to some degree by the machine learning model. Despite some discrepancies in decompressed videos, MLPEG was also able to maintain visual quality and high resolution. MLPEG shows that machine learning is a useful tool in the field of data compression.

Materials

- Python 3.9
- TensorFlow 2.11, Numpy, OpenCV, Pandas
- Inter4K: A Dataset for Video Interpolation and Super-Resolution in 4K

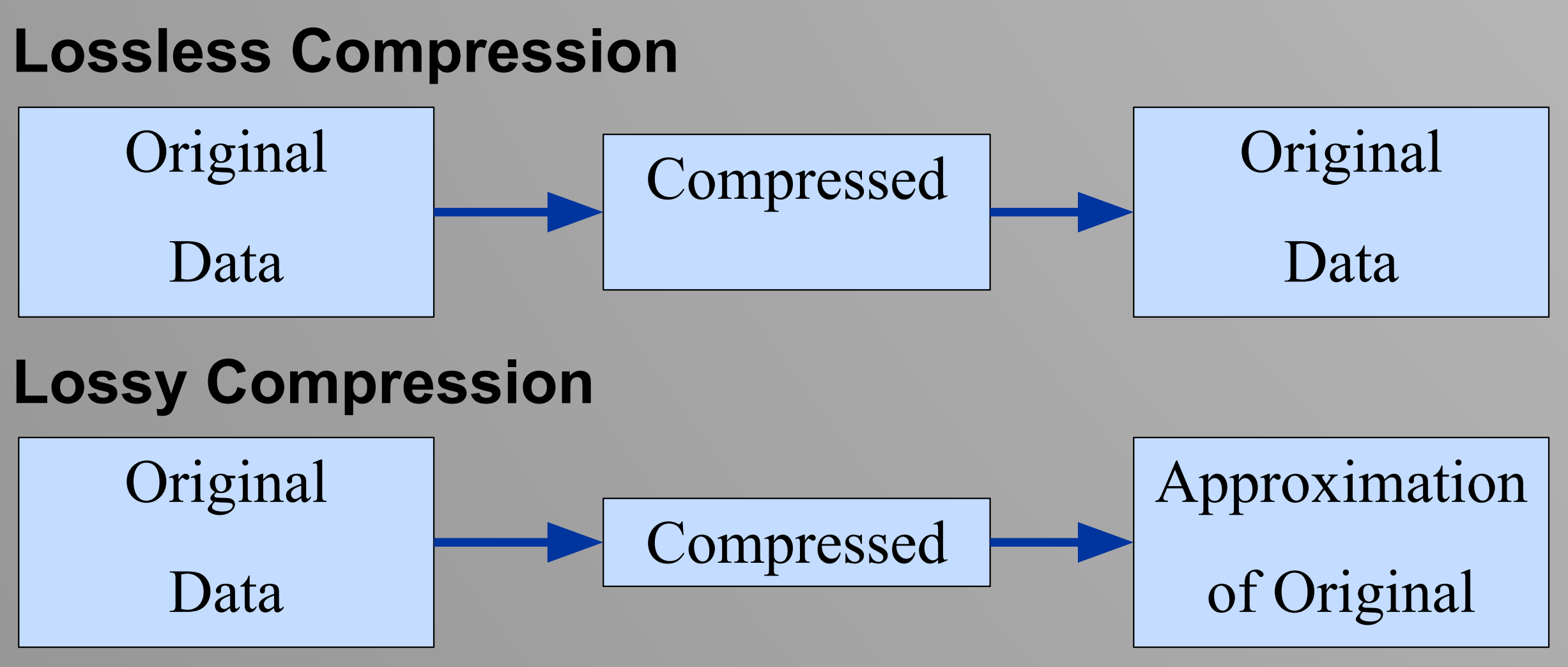


Figure 1: Visual representation of the distinction between lossy and lossless data compression.

Method

For compression, MLPEG removes all frames except every Nth, based on the sampling rate.

For decompression, MLPEG uses an interpolation model and upscale model to create a visually similar version of the original.

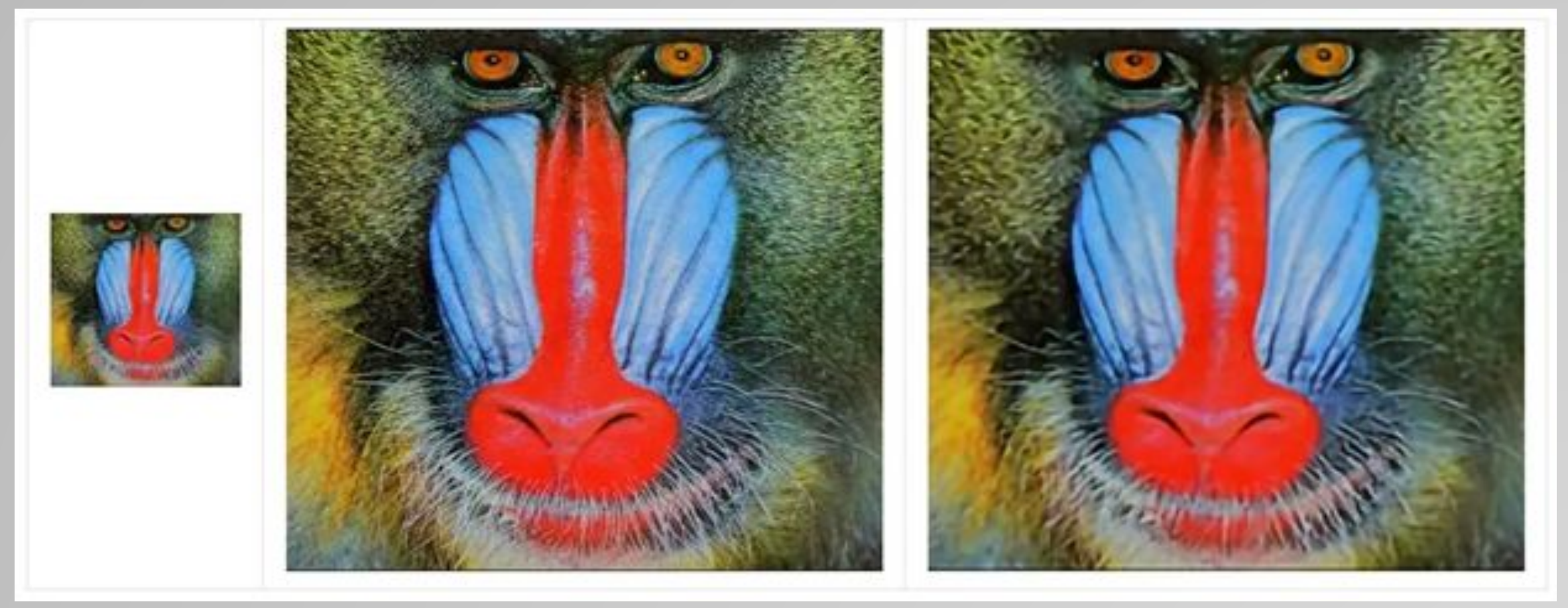


Figure 2: A super-resolution example by ESPCN; Left: low-resolution image, Middle: original Image, Right: super-resolution result, Upscaling factor: 3

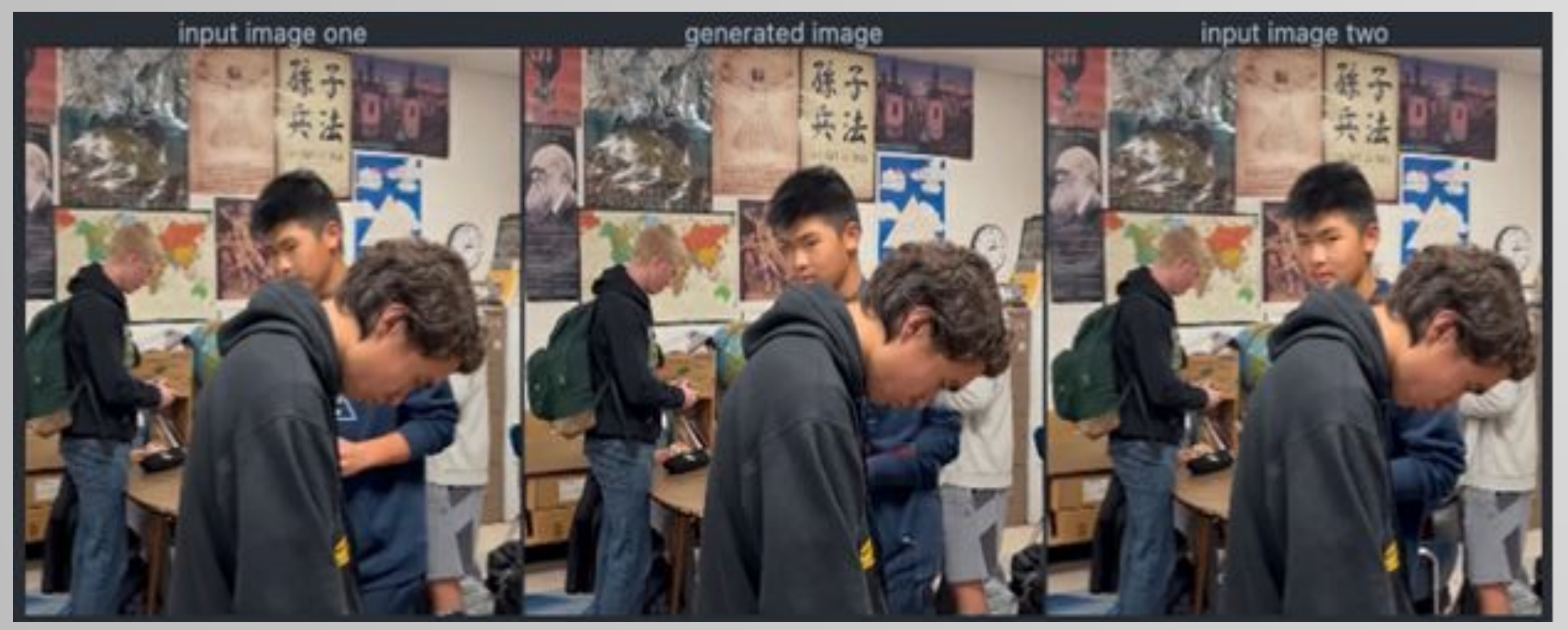


Figure 3: FILM Interpolation: Given two input images one is generated for in between

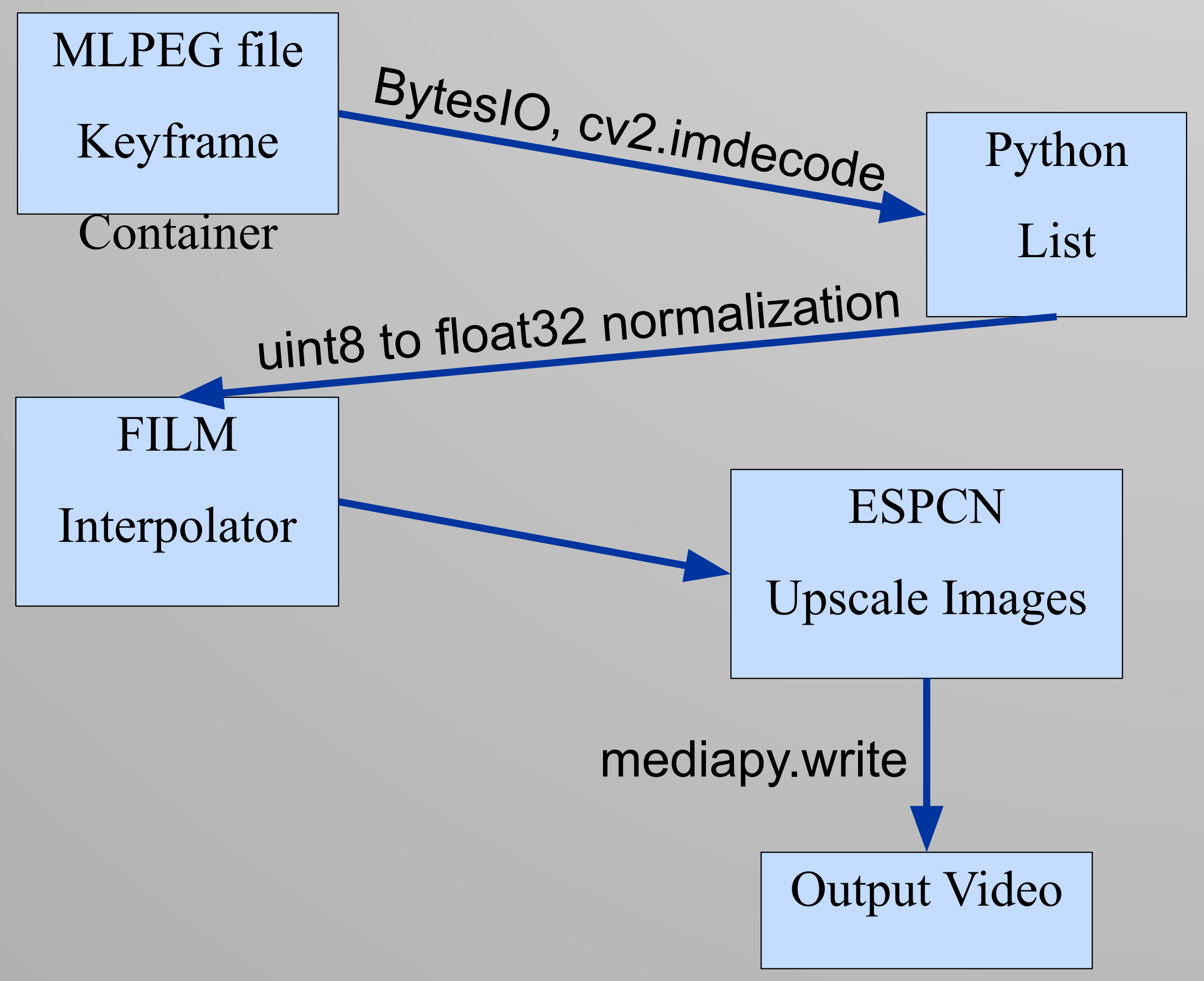
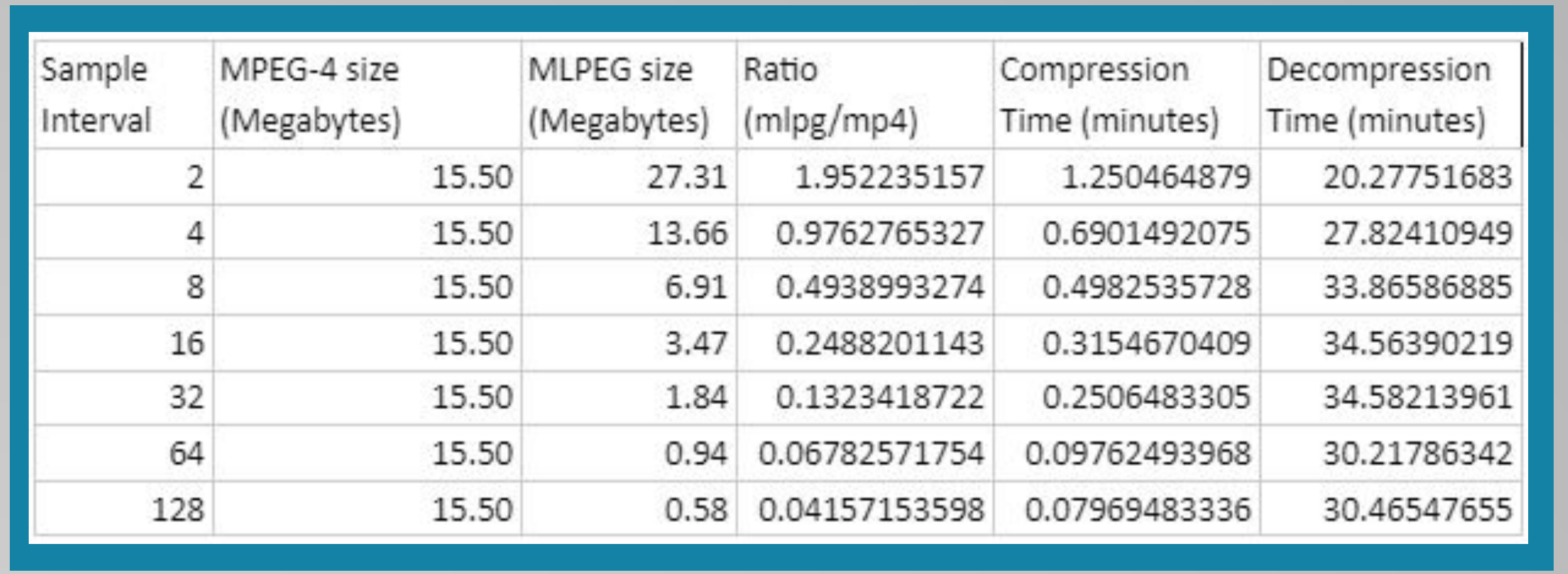


Figure 4: Transcribed from the author's research notebook, this diagram approximates the decompression process.

Results



Sample Interval	MPEG-4 size (Megabytes)	MLPEG size (Megabytes)	Ratio (mlpg/mp4)	Compression Time (minutes)	Decompression Time (minutes)
2	15.50	27.31	1.952235157	1.250464879	20.27751683
4	15.50	13.66	0.9762765327	0.6901492075	27.82410949
8	15.50	6.91	0.4938993274	0.4982535728	33.86586885
16	15.50	3.47	0.2488201143	0.3154670409	34.56390219
32	15.50	1.84	0.1323418722	0.2506483305	34.58213961
64	15.50	0.94	0.06782571754	0.09762493968	30.21786342
128	15.50	0.58	0.04157153598	0.07969483336	30.46547655

Figure 5: This data table shows the averages values of the recorded metrics for each sample interval. The same data is used to construct the following graphs.

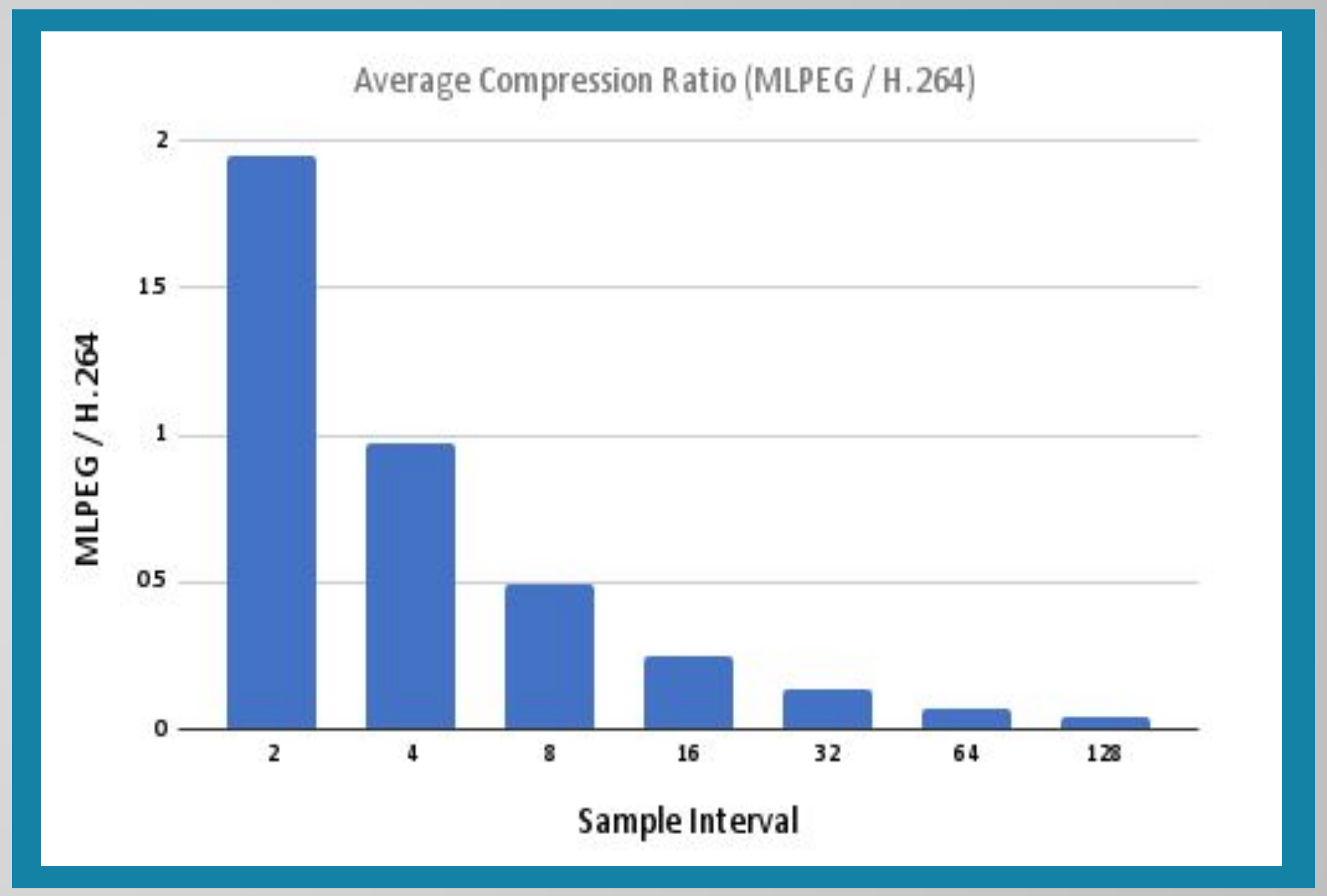


Figure 6: This graph shows the average compressed file sizes of the output ".mlpg" files. A very clear downward exponential slope is visible, as was intuitively expected by the doubling of the sample rate at each step.

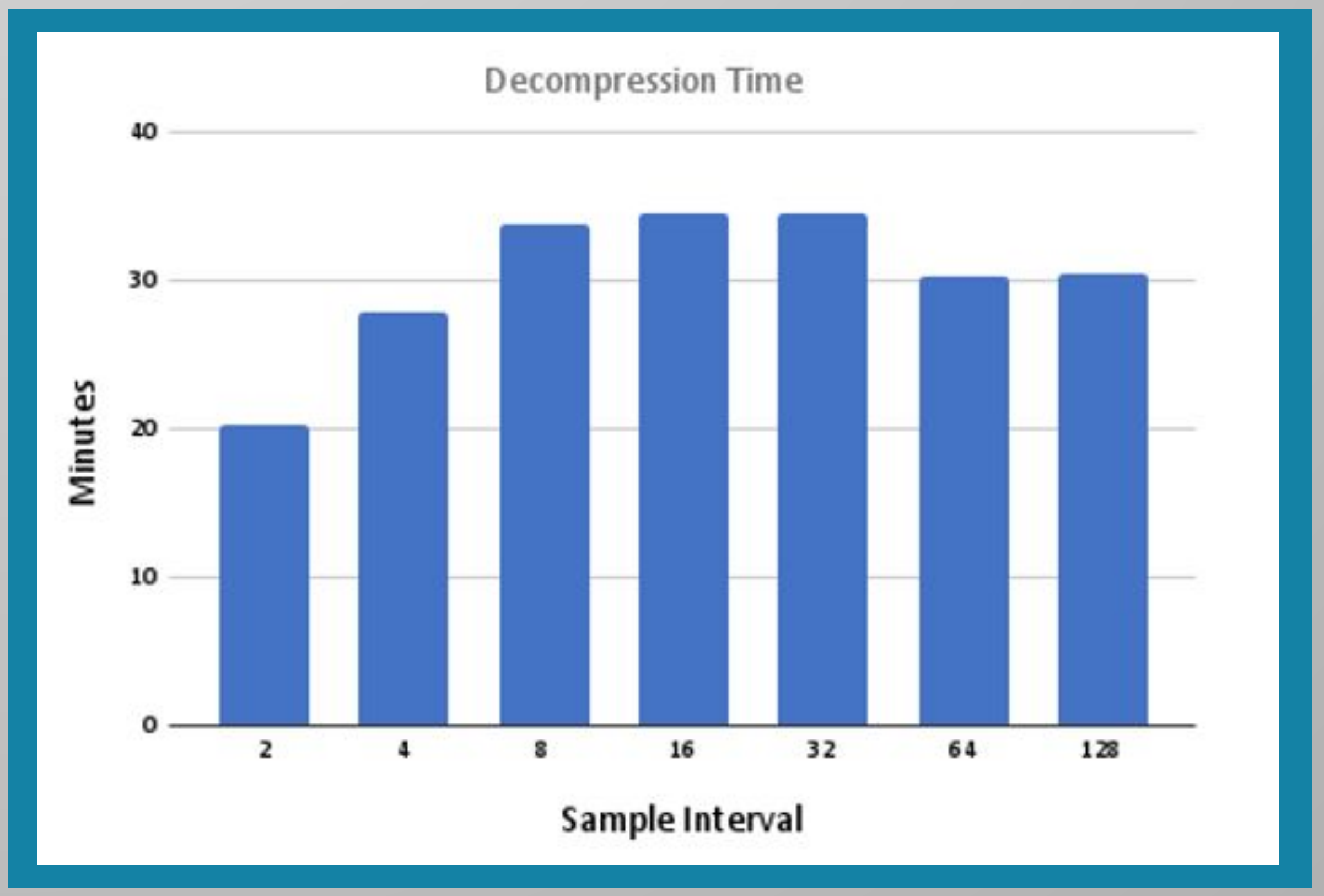


Figure 7: This graph shows the average decompression time for each sample interval. The stagnation and eventual dip in decompression time as sample interval increases suggests an optimal configuration at around the sample interval of 8 area.

Discussion

The primary quantitative metrics that determine the effectiveness of a compression algorithm are compression and decompression times, and compression ratio.

Beginning with timing, MLPEG is slower than H.264 at low values for the sample interval, but significantly faster at high values. Decompression times show a plateau and then, interestingly, a decrease. Though finding the exact cause will require further testing, a valid point of reasoning is that there could be some optimizations over time in doing many recursive interpolations with the FILM model.

MLPEG can achieve very small compression ratios when compared to H.264, with the compressed file taking up less that 5% of the size of an MP4 at a sample interval of 128.

Most notable is that the MLPEG algorithm achieved a smaller compression size than H.264 at the relatively low sample interval of 4. The qualitative metric that determines a compression algorithm's effectiveness is visual fidelity; at a sample interval of 4, the compressed videos visually appear very similar to their original counterparts. The H.264 algorithm had its compression ratio beaten by MLPEG with an average compression time of 41 seconds.

Future Implications

Future development of machine learning based compression algorithms of all types is highly encouraged. Development of low level but specialized machine learning models can lead to massive speed-ups in algorithms like MLPEG. As learned compression becomes more widespread, the dominance by video streaming on the world's internet bandwidth could be replaced with the transmittance of vital software and educational resources. At the same time, video streaming may be made available to historically excluded areas.