Optimal Hyper-parametrization for efficient Video Compression

Video Compression Hackathon - SBI

Powered by Microsoft Corporation Pvt. Ltd.

Problem

- With the advent of Video based Customer Identification, multiple use-cases have emerged for customer onboarding in a secure, paperless, cost-effective and friendly manner.
- Storage and retrieval of these video files is a challenge especially given the expected tsunami of video files that are expected to be generated on extending more use-cases to our client base of over 45 crore customers.

Approach

- Reinforcement Learning Models provide great methods for getting optimal hyperparameters for other methods/models.
- These methods have been used heavily in all domains.
- I made use of the same to learn optimal parameters for a vast array of these.

Approach

- These included (among others!) : Maximum Reference to L0, Early Skipping etc.
- These methods were tabulated into 4 optimal settings in `vals.txt`
- Then, these were made use of for encoding using `x265`.

Approach

- The RL part was done using Feedback Networks' Architecture in TensorFlow.
- The Encryption Algorithm used was AES.
- We also used `zlib` for further compression.

Video used

- `phase3.mp4`
- Profile : H.264
- Dimensions : 1920×1080
- FPS : 30.0
- Bit-Rate : 17036 kbps
- Size : 127.8 MB
- The file may be found here.

Decompressed Video

- `phase31.mp4`
- Profile (Main) : H.265
- Dimensions : 1920×1080
- FPS : 30.0 (as expected)
- Bit-Rate : 1381 kbps
- Size (compressed) : 9.5 MB

Hardware Details

- CPU : Intel(R) Core(TM) i5-1035G1 CPU 1.00GHz
- Memory : 8 GB
- Memory Clock : 3200 MHz
- L1 Cache : 128 kB
- L2 Cache : 2 MiB
- L3 Cache : 6 MiB

Why this model?

- Different Types of applications provide scope for different types of compression shorthands.
- Our approach enables us to learn these shorthands

Functional Requirements

- `x265`
- `opencv4`
- Python's `cryptography`
- Python's `zlib`
- `ffmpeg`

Corresponding Microsoft Tools

- `x265` $\equiv$ `HEVC Video Extensions`
- `opencv4` $\equiv$ Microsoft Media Foundation `MFIDL`
- `ffmpeg` $\equiv$ `FFmpegInterop`

All these tools are easily available in Azure Cloud Services and thus, can be used to leverage the application into production.

Plan of Deployment

- **Scope :**The Concept of using RL Methods to optimally hyper-parameterize HEVC can be leveraged according to specific applications.
- **Pre-requisite :**If the Bank could provide us with a training set of videos, it wants to compress/encrypt, we would get vastly better results!
- **Infrastructure :** Azure Cloud Deployment
- **Time Required to setup :** 2-3 Weeks (Debugging included!)

Non-functional Requirements

- The Video isn't recorded in camcorders which make use of Compressed Sensing.
- The Noise is minimal and patterns in the images remain consistent.
- The Machine on which the Application is used has $\geq$ 8GB RAM.
- This would enable greater compression due to easier Identification of Generic features.

Our Edge

- `HEVC` is known to better than state-of-the-art Deep Learning Methods (*DeepCoder*) by an additional 50 %.
- Use-case specific optimization in *HEVC* is generically obtained by making use of hyperparameter optimization.
- Feedback functions could also be tinkered to better reflect our Requirements (should they ever change!)

Github Repository

My Actual Github ID is thevaliantthird

I've kept the Video Hackathon submission Repository via another profile, SBI-Video-Hackathon

Video Demonstration

I have demonstrated usage of my application here.

Thank You!

Name : Shubh Kumar
Email : shubh5796@gmail.com
Phone Number : 9470434205
Github ID (original) : thevalianttthird Website : thevalianttthird