Dejavu: Enhancing Videoconferencing with Prior Knowledge

Pan Hu, Rakesh Misra, Sachin Katti
Stanford University and Uhana Inc.
Poor Quality of Videoconferencing over Wireless Links

- Low uplink bandwidth over LTE
- Conservative bandwidth utilization to maintain interactivity

![Speed Results](image)

- Available bandwidth
- Actual sending rate
Poor Quality of Videoconferencing over Wireless Links
Prior Solutions for better videoconferencing

- Application
  - Dejavu
- Presentation
  - Video codec
- Session
  - Congestion control
- Transport
- Network
- Data Link
  - MIMO, mmWave
- Physical
**Key Insight: visual similarities**

Arbitrary videos from Youtube: few visual similarities

Videoconferencing in the same room: abundant visual similarities

Dejavu: is a system that optimizes videoconferencing performance by leveraging similarities across past and current sessions.
How to Leverage the Similarities?

- **Dejavu**: let neural network learn the similarities

  ![Diagram](image)

  - Low quality training input
  - **NN with Knowledge**
  - High quality training output
How to Leverage the Similarities?

- Dejavu: let neural network learn the similarities

- Low quality test input

- NN with Knowledge

- High quality test output
Current System

Sender

Live video-conferencing sender

Low quality video over network

Live video-conferencing receiver

Receiver
Dejavu Offline Stage

Server

- Historical sessions of the same scenario
- Generate training data
- Low quality

Dejavu Learning Engine

- High quality
- Learned neural network model
- Share model with receiver

Sender

- Live video-conferencing sender
- Low quality video over network

Receiver

- Live video-conferencing receiver
- Historical sessions of the same scenario
- Generate training data

Low quality

Historical sessions of the same scenario

Generate training data

Low quality

Dejavu Learning Engine

Share model with receiver

Learned neural network model

Low quality

Historical sessions of the same scenario

Generate training data

Low quality

Dejavu Learning Engine

Share model with receiver

Learned neural network model
Dejavu Online Stage

**Sender**
- Live video-conferencing sender
- Low quality video over network

**Server**
- Generate training data
- Low quality
- Share model with receiver
- Dejavu Learning Engine
- High quality
- Learned neural network model

**Receiver**
- Live video-conferencing receiver
- Dejavu quality enhancing neural network
- High quality video

**Historical sessions of the same scenario**

**Dejavu Learning Engine**
- High quality

**Generate training data**
- Low quality

**Share model with receiver**
- Learned neural network model

**Low quality**
- Historical sessions of the same scenario

**Dejavu quality enhancing neural network**
- Historical sessions of the same scenario

**Historical sessions of the same scenario**
Design of Dejavu Learning Engine

- **NN Architecture**
  - Convolution layers + Residual blocks

- **Preprocessing**
  - Compress videos into different quality levels with FFmpeg
  - Train on shuffled small patches rather than full frame
  - Convert RGB into YUV, processing on Y only to speed up inference
Trained/validate on the first four videos and test on the last one.

Video format:
- 270p (480*270) and 540p (960*540)
- [100, 200, 300, 500, 800, 1000, 2000] kbps

Measure Peak Signal to Noise Ratio

540p: up to 1.3dB PSNR gain

270p: up to 1.1dB PSNR gain
Performance Benchmark: Bandwidth Saving

- **Up to 30%** bandwidth saving for the same PSNR

- Similar performance gain as developing a new generation of video codec
Performance Benchmark: Visual difference
Future Work

- **Evaluate real-world performance**
  - Collect large-scale, real-user dataset.
  - Evaluate user experience that includes processing delay in real system.

- **More efficient inference**
  - Exploit inter-frame similarity based on motion estimation (from codec) or reuse part of NN.
  - Knowledge distillation or model compression / quantization to speed up / fit in small RAM.
Conclusion

- **Summary**
  - Dejavu leverage similarities across videoconferencing sessions to improve future video performance – *caching live content!*
  - Similar mechanism could be applied to a wide range of video streaming apps (like Twitch / Youtube) to improve quality or reduce CDN cost!

- **Future work**
  - Evaluate real-world performance
  - More efficient inference