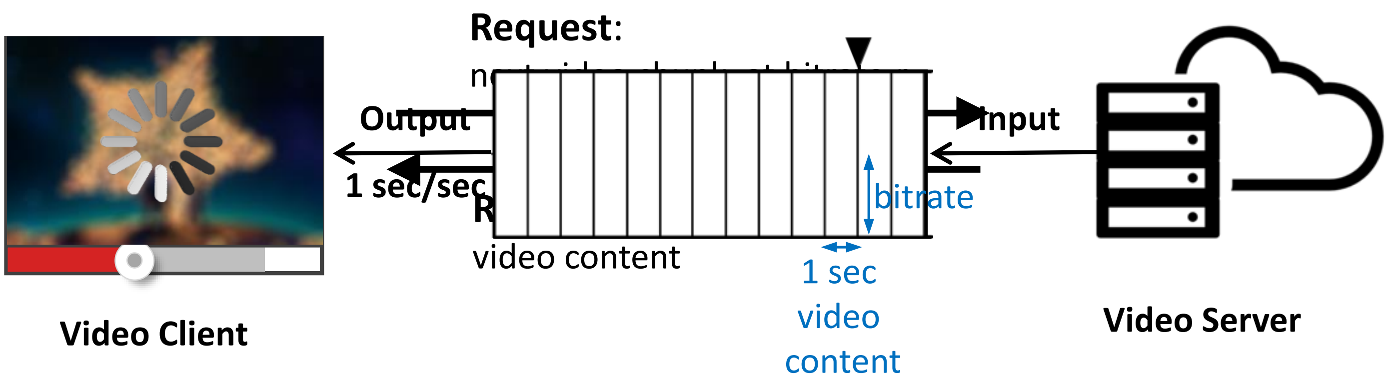
HTTP Streaming & Optimization Areas

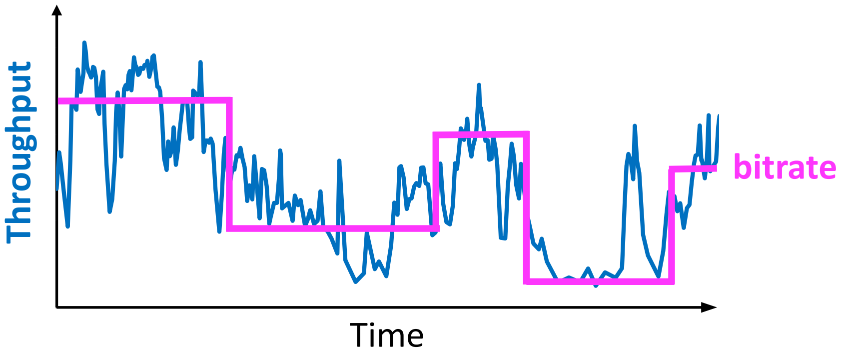
HAS Main Principles

- Store data in small chunks for different Representations (resolution, bitrate, frame rate, codec)
- Monitor network conditions
- Adapt the transmission data rate

Dynamic Streaming over HTTP (DASH)



Adaptive Bitrate (ABR) Algorithms



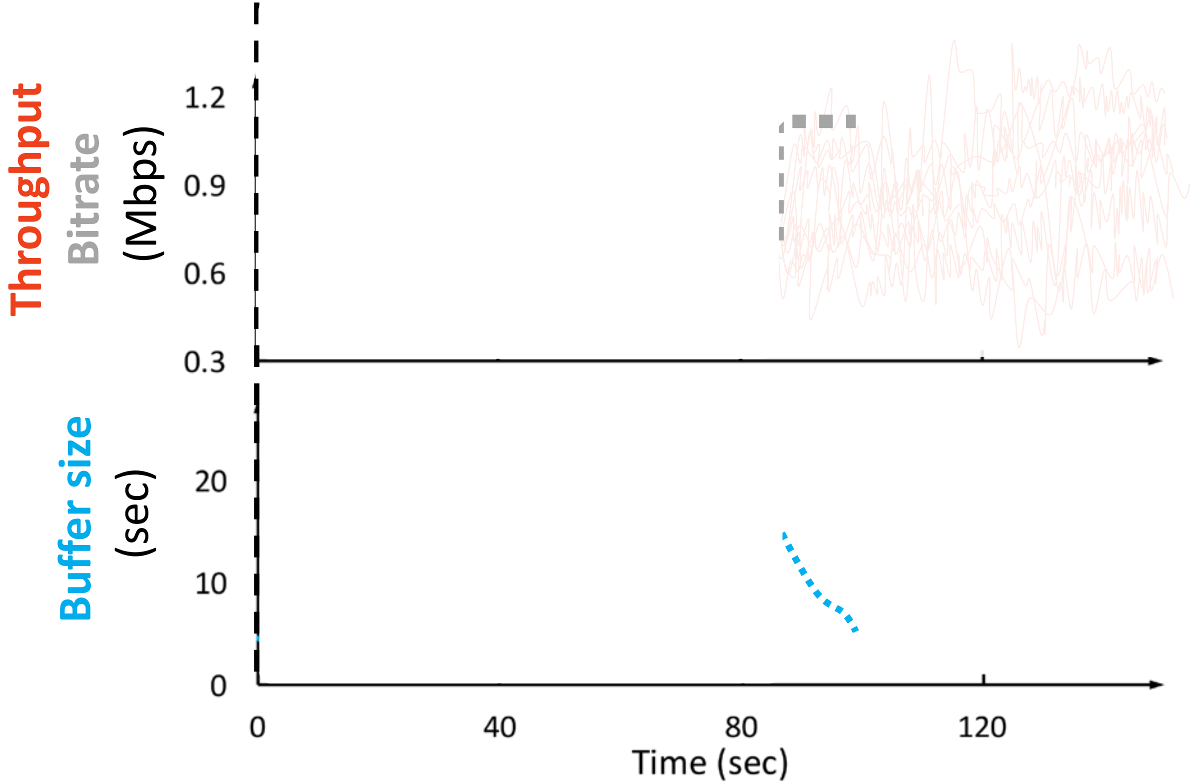
Animation borrowed from Te-Yuan Huang (SIGCOMM '14) http://conferences.sigcomm.org/sigcomm/2014/doc/slides/38.pdf

Regular ABR Algorithms

- Rate-based: pick bitrate based on predicted throughput
  - FESTIVE [CoNEXT’12], PANDA [JSAC’14], CS2P [SIGCOMM’16]
- Buffer-based: pick bitrate based on buffer occupancy
  - BBA [SIGCOMM’14], BOLA [INFOCOM’16]
- Hybrid: use both throughput prediction & buffer occupancy
  - PBA [HotMobile’15], MPC [SIGCOMM’15]

Simplified inaccurate model leads to suboptimal performance

Why is ABR Challenging?



Network throughput is variable & uncertain

Conflicting QoE goals

- Bitrate
- Rebuffering time
- Smoothness

Cascading effects of decisions

Neural Adaptive Video Streaming with Pensieve

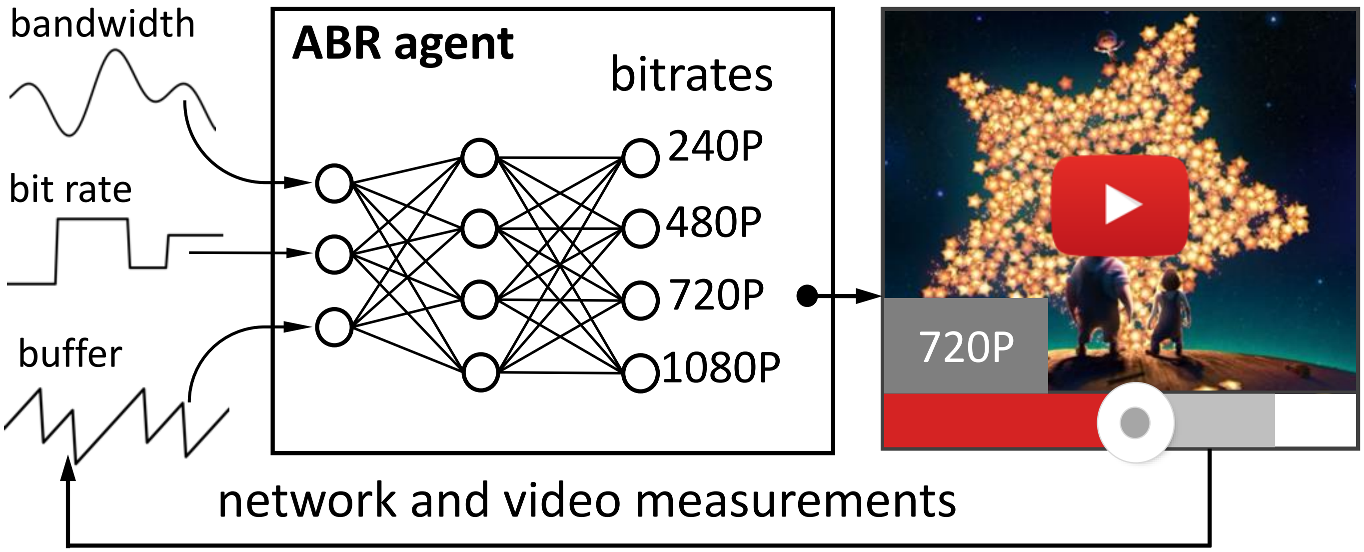
Hongzi Mao

Ravi Netravali   Mohammad Alizadeh

Massachusetts Institute of Technology

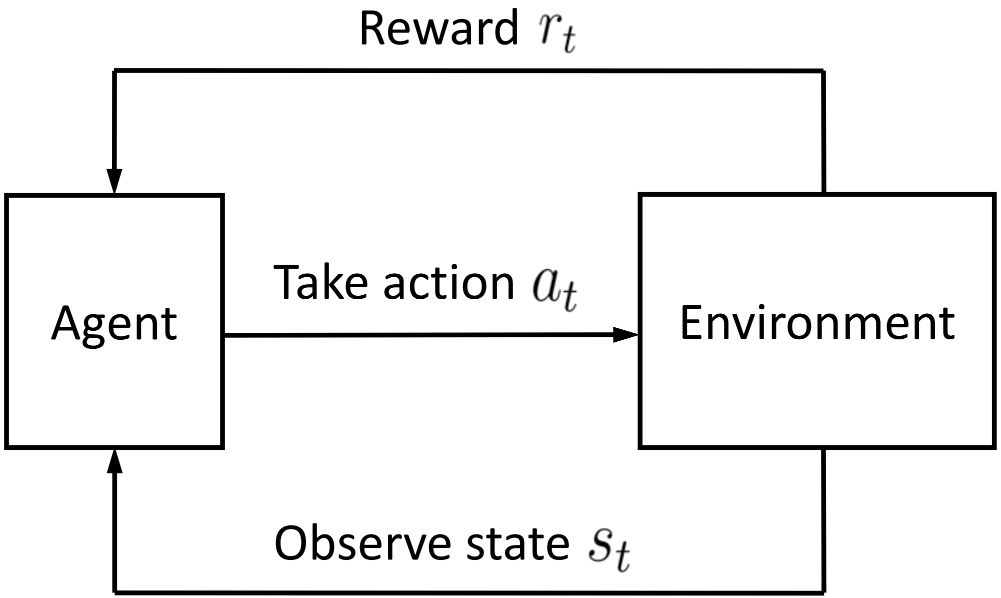
CSAIL

Pensieve



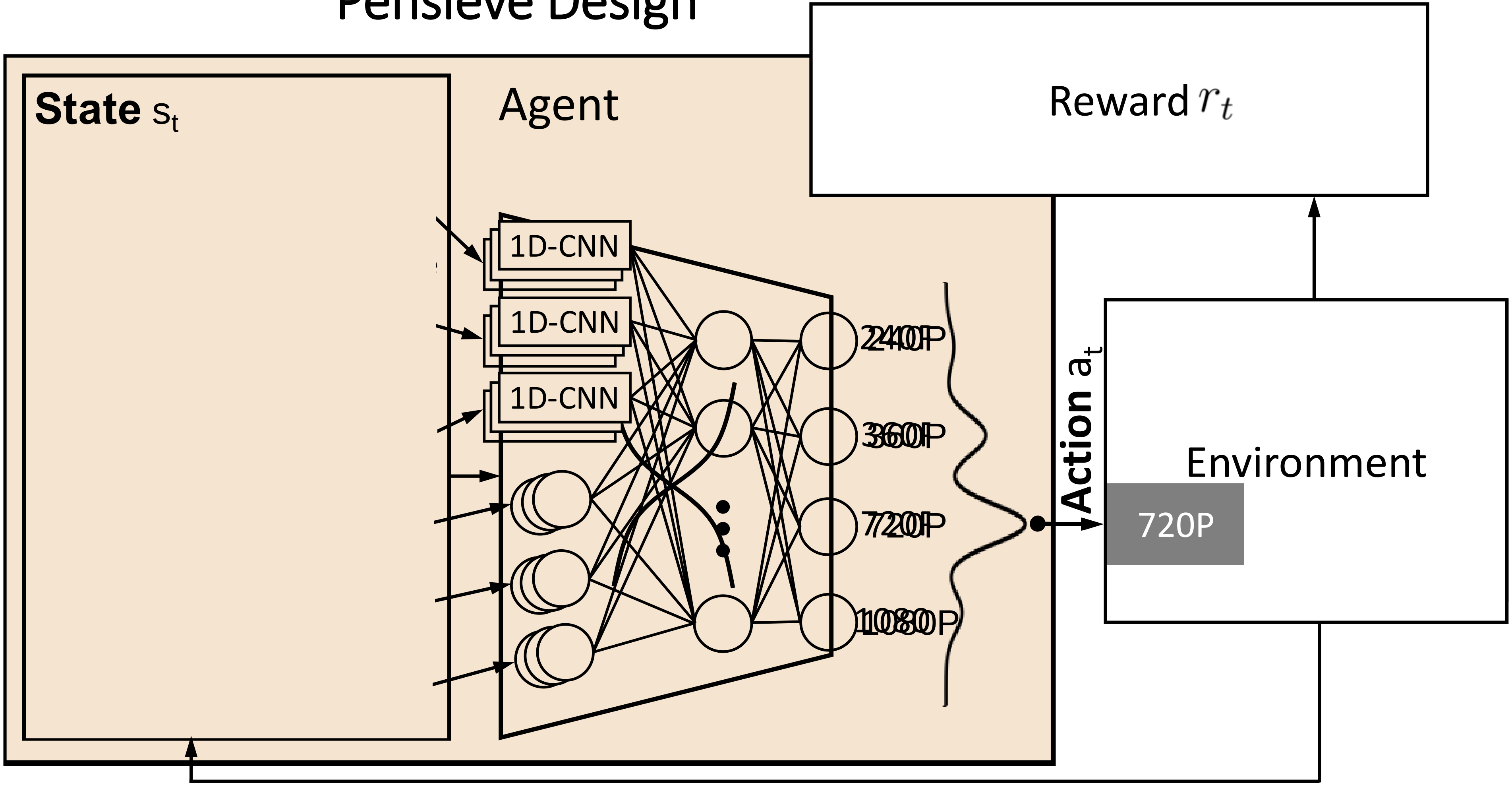
Pensieve learns ABR algorithm automatically through experience

Reinforcement Learning

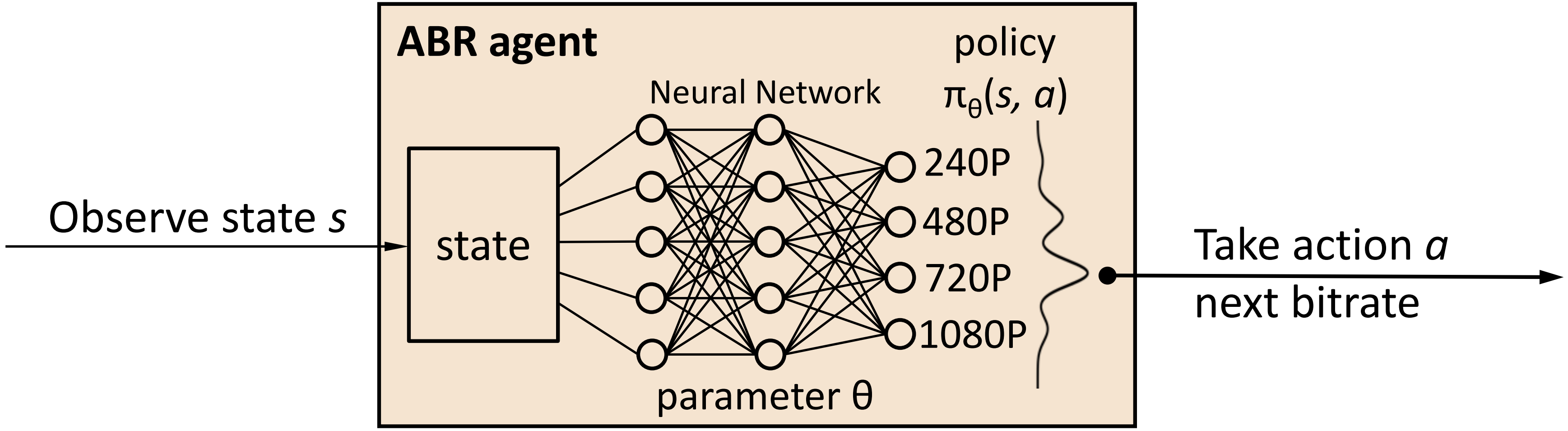


Goal: maximize the cumulative reward $\sum_t r_t$

Pensieve Design



How to Train the ABR Agent

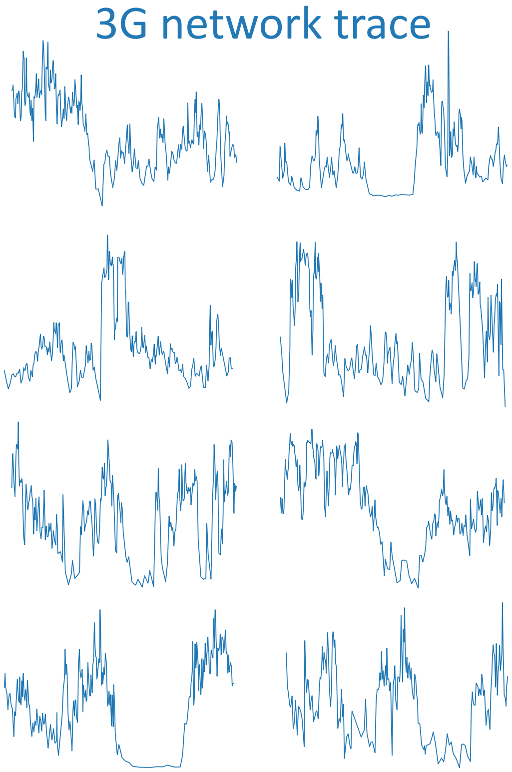


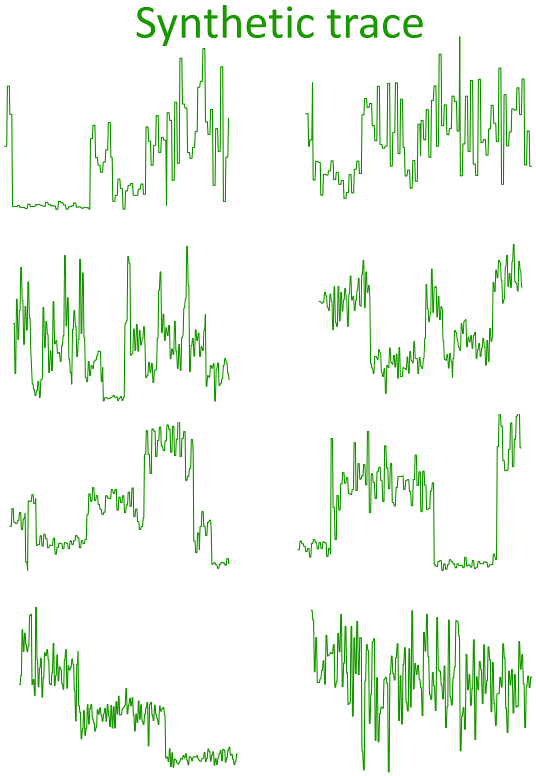
Collect experience data: trajectory of [state, action, reward]

Training: $\theta \leftarrow \theta + \alpha \nabla_\theta \mathbb{E}_{\pi_\theta}\left[\sum_t r_t\right]$

estimate from empirical data

Traces for Generalization





- Trace generated from a Hidden Markov model
- Covers a wide range of average throughput and network variation

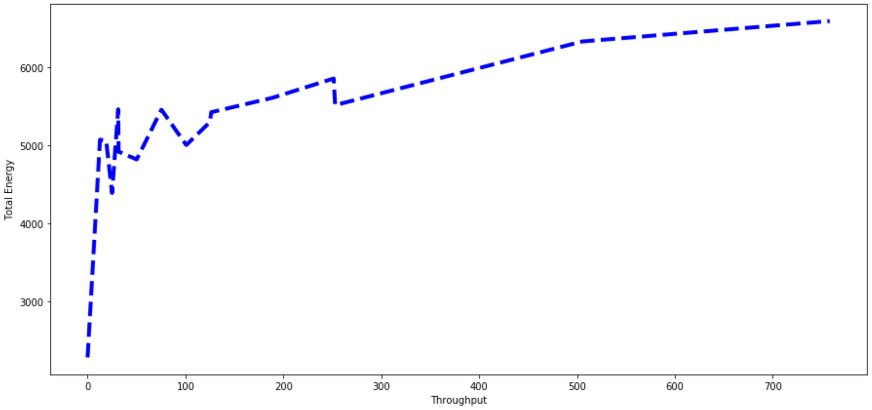
Shortcomings of Existing ABRs

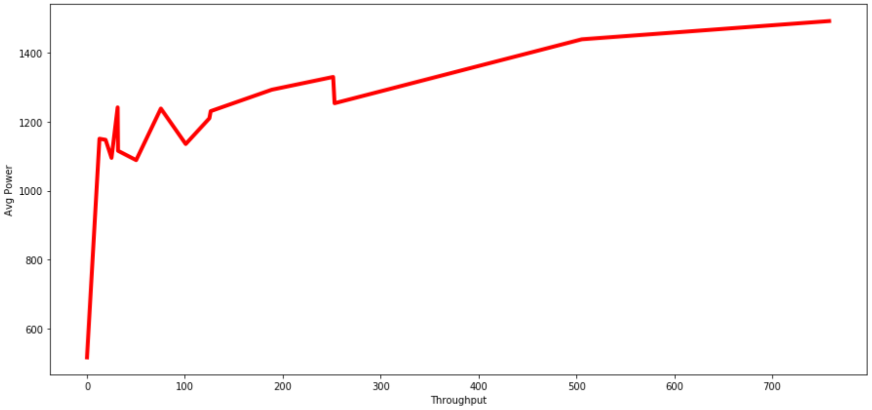
- Greedy to bitrate (Do not consider perceptual quality)
- Energy consumption generally not included
- Does not reflect real world implementations
- Quality models are linear and have some wrong assumptions
  - Penalize all rebuffering events same (in the beginning or in the middle)
  - Penalize all oscillations same (highest to lowest or highest to 2nd highest)
  - Higher bitrate is always necessary for better experience

Optimization Problem

How to optimize energy consumption without sacrificing quality of experience?

Preliminary Results





What we have?

- Information about representations
  - File size
  - Quality metrics
  - Power model
- Real world network traces
- Simulator Environment

Using Reinforcement Learning

- Environment provides information for
  - Energy consumption of each available option
  - Quality metrics of each available option
  - Past network conditions
  - Current buffer conditions

Using Reinforcement Learning

- State Space
  - Buffer size
  - Current chunk size/number
  - Throughput
  - Download time
  - Rebuffer time
  - Remaining chunks
- Action Space
  - Different versions of the video for each chunk

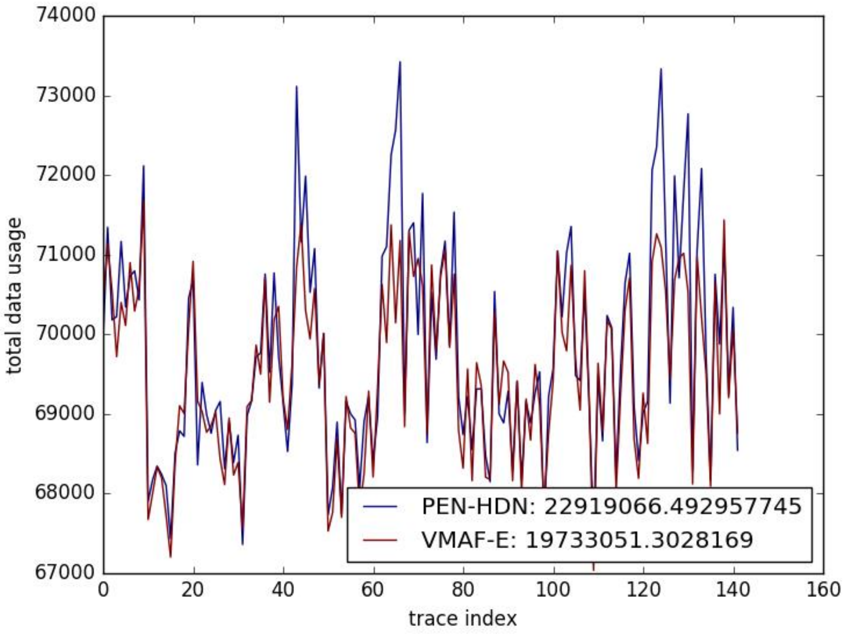
Using Reinforcement Learning

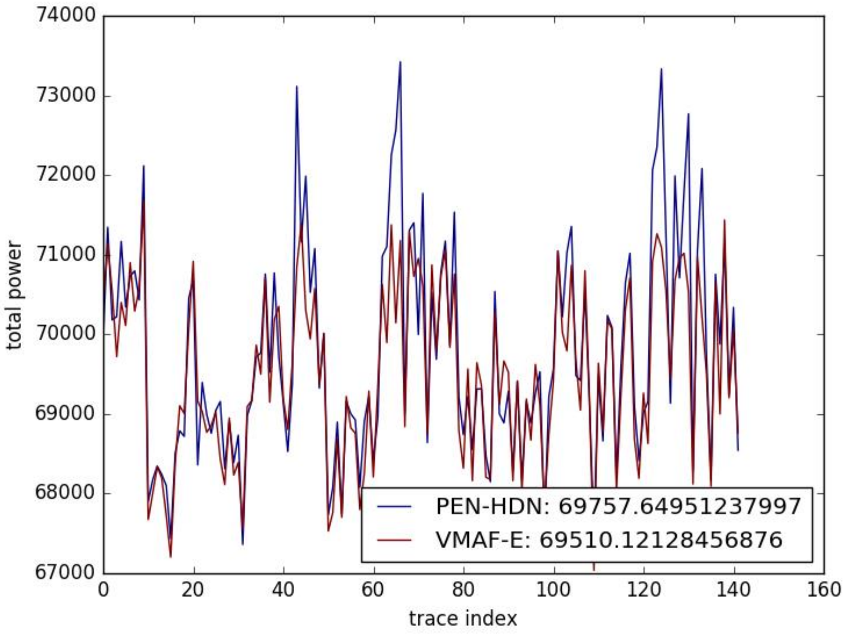
- Reward model should contain
  - Energy consumption
  - Quality metrics
  - Oscillations
  - Rebuffering

Experiments

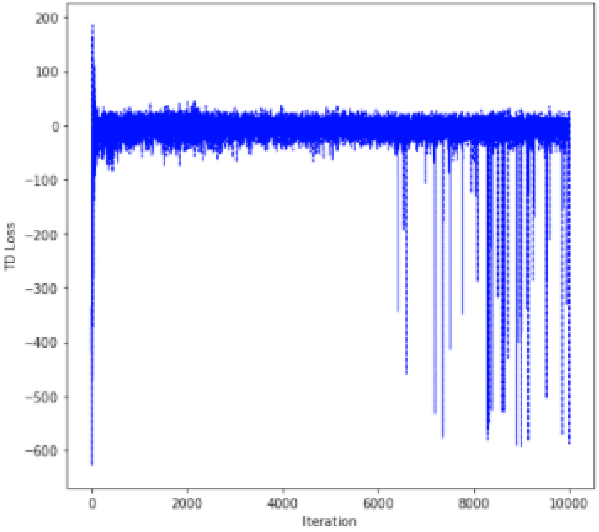
- On Pensieve
  - Pensieve is trained with updated reward models
  - VMAF-E, VMAF-Q, VMAF-EQ, VMAF-LN
- On DQN based model
  - Trainings with updated state space
  - Trainings with different network architectures
    - MLP, 1Conv1D+MLP, 2Conv1D+MLP
  - Trainings with different reward models
    - VMAF-LN, VMAF-E, VMAF-ES

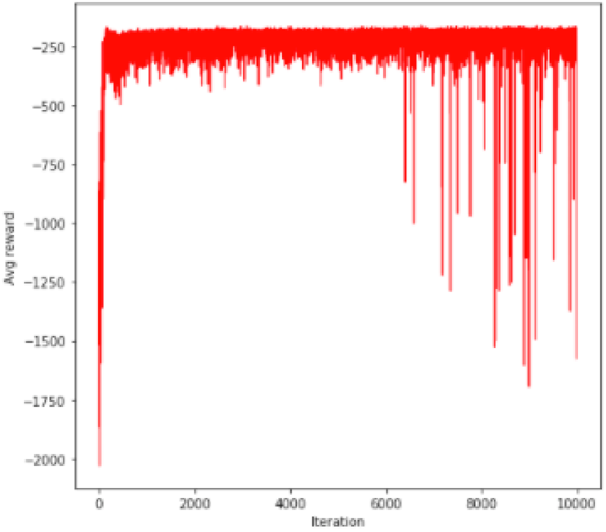
Pensieve Experiments

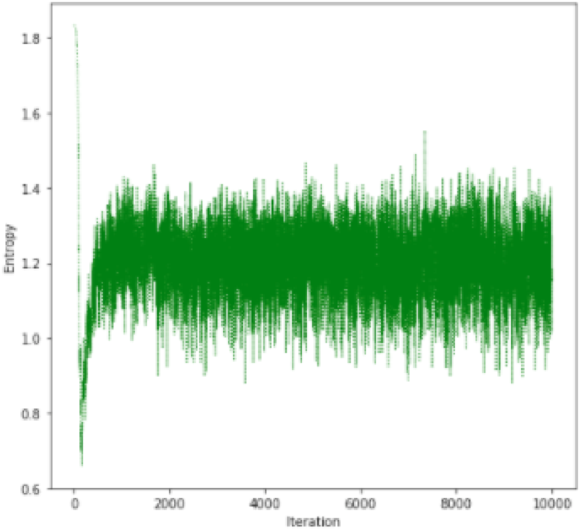




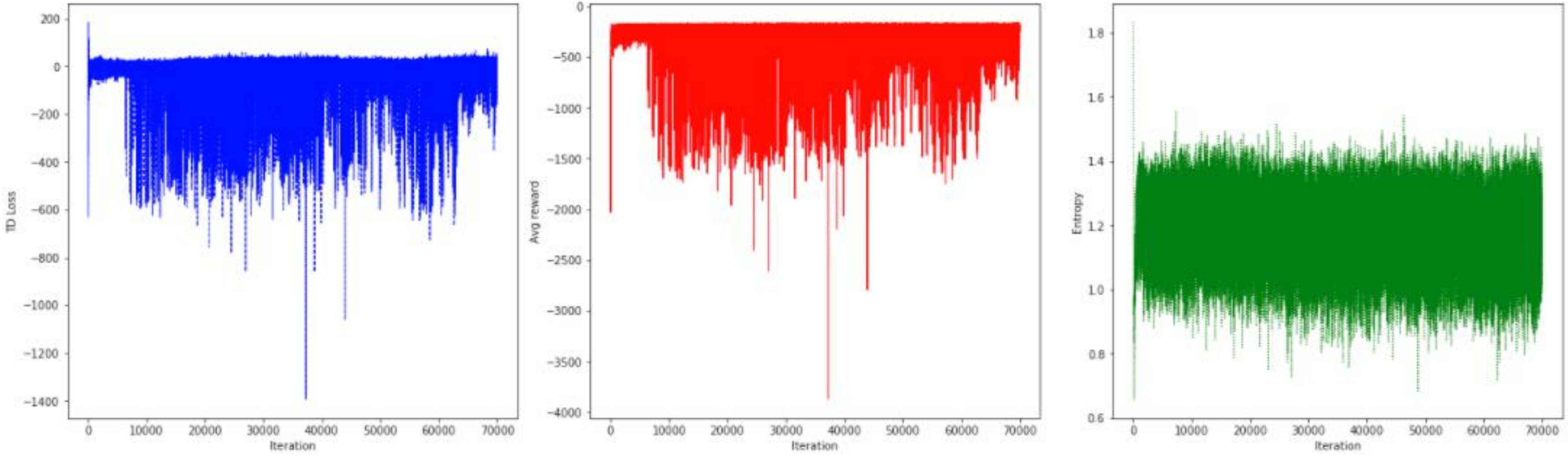
Pensieve Training Results Avg Reward 10K ~ -232





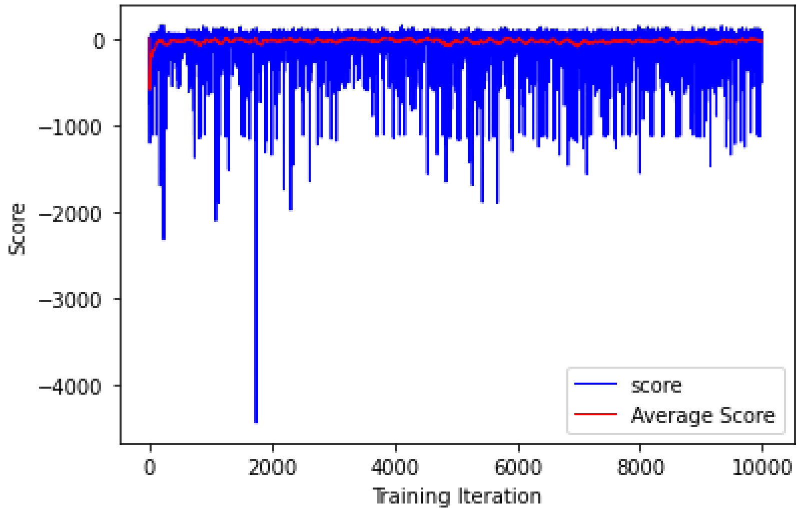


Pensieve Training Results Avg Reward 70K ~ -232

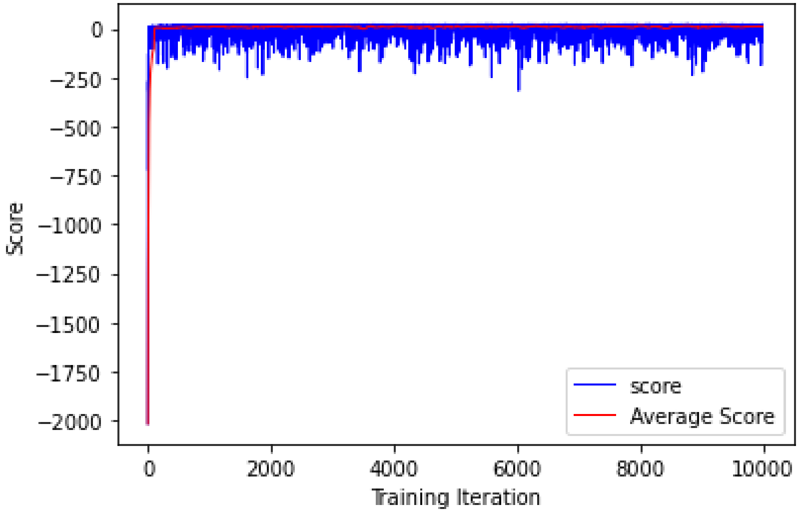


DQN Experiments Different Network Traces

Multiple Traces – Avg reward ~ -20

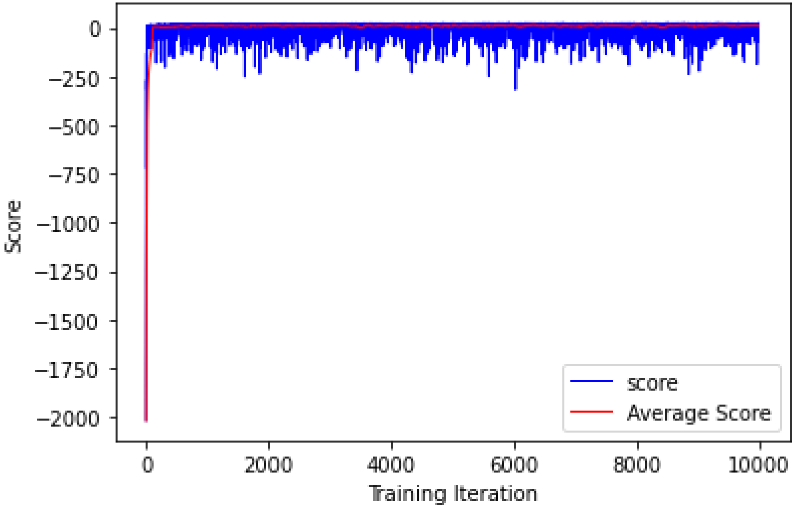


Single Trace – Avg reward ~ 8

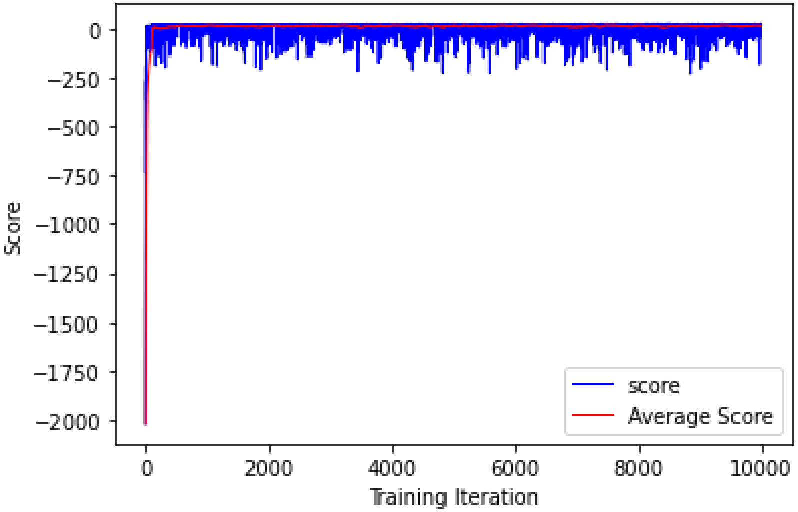


DQN Experiments Different States

St. Space with 5 components (avg 8)

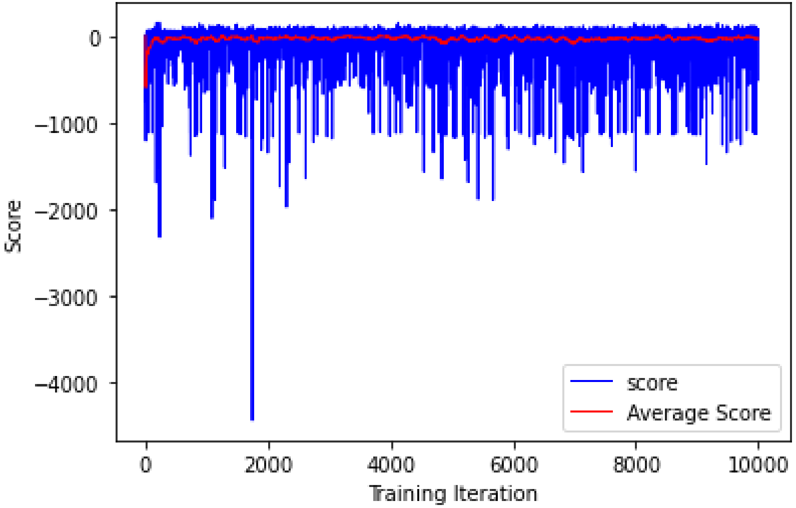


St. Space with 6 components (Avg 11)

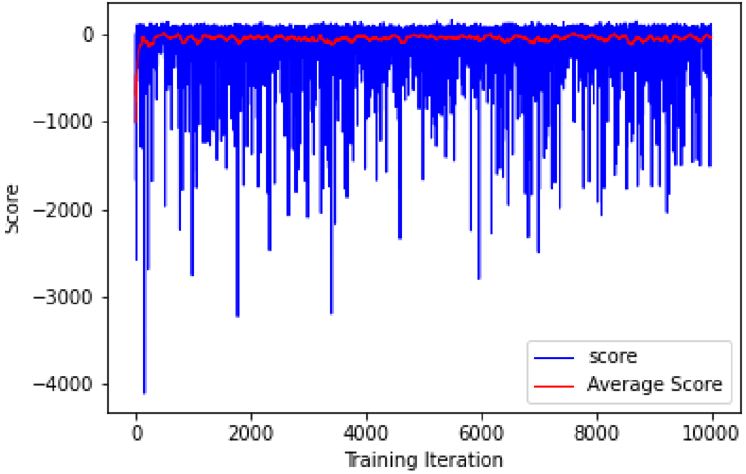


DQN with Different Neural Networks

MLP Avg ~ -20

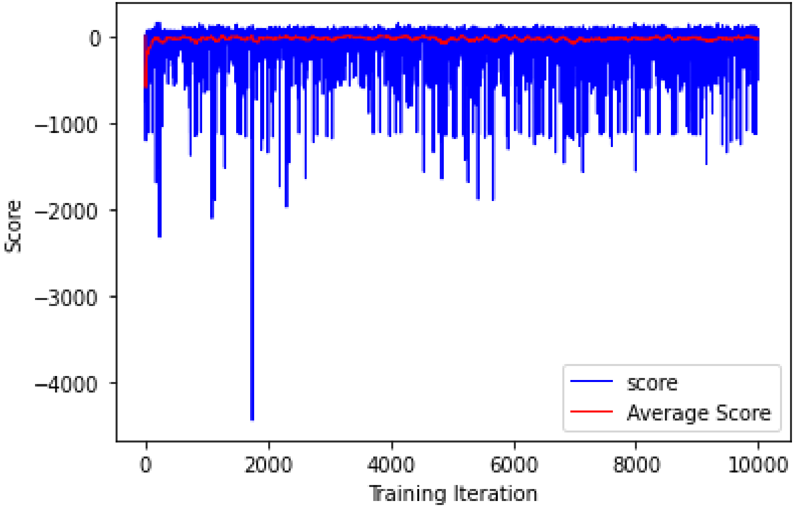


1 layer Conv1D + MLP Avg ~ -44

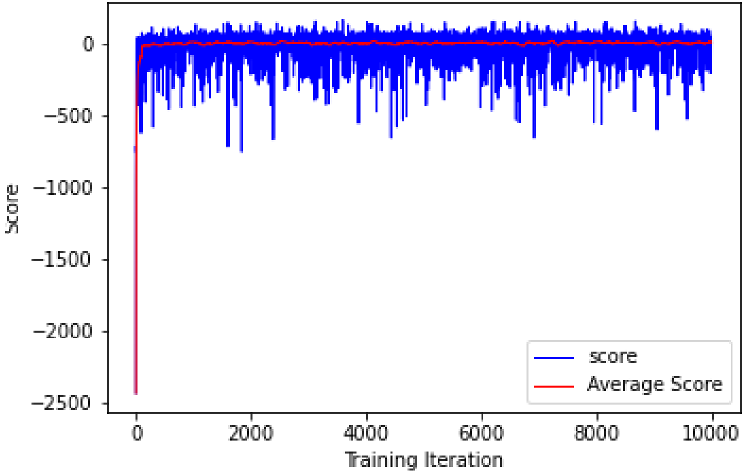


DQN with Different Neural Networks

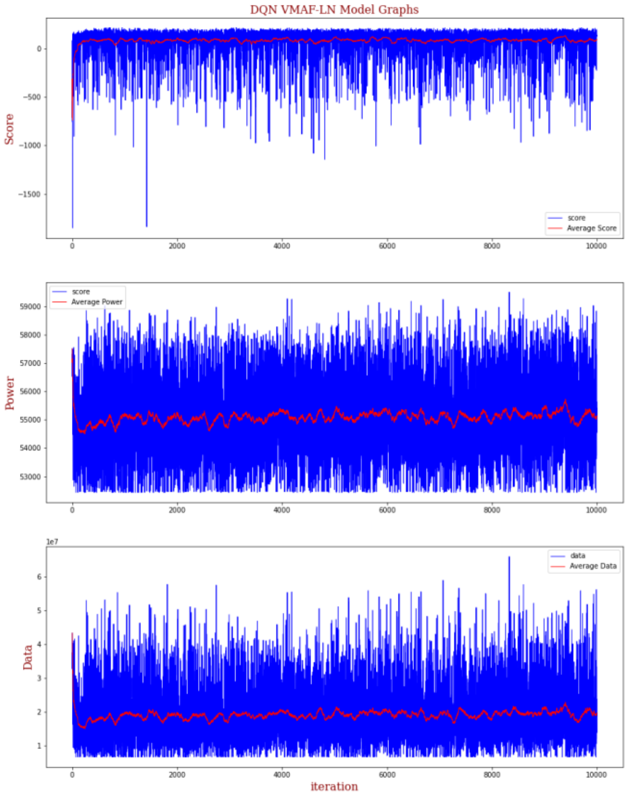
MLP Avg ~ -20

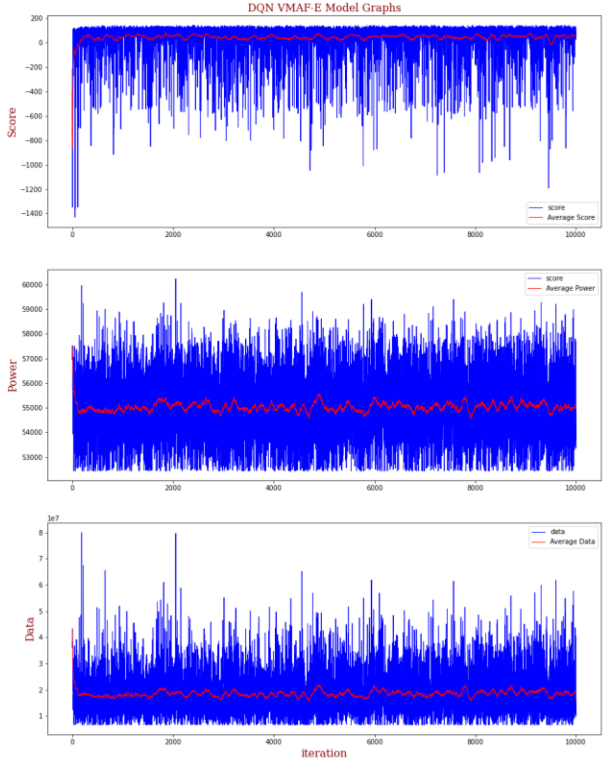


2 layer Conv1D + MLP Avg ~ - 24

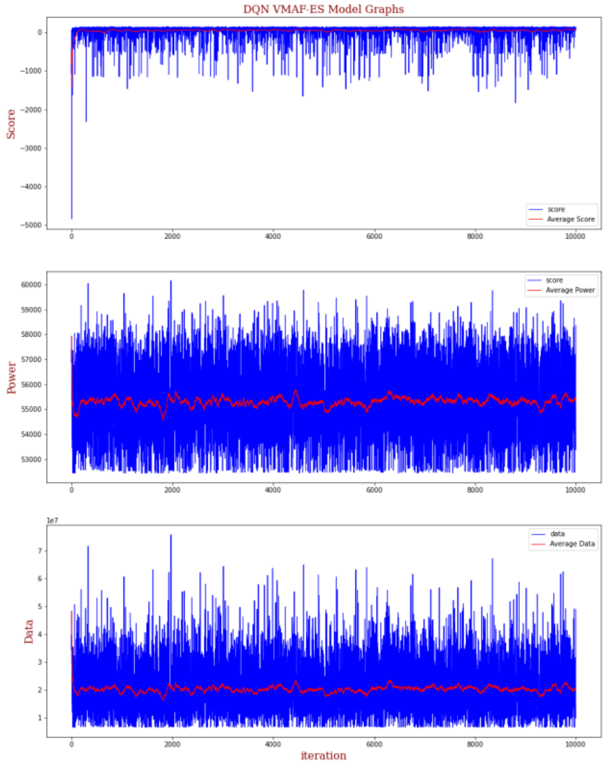


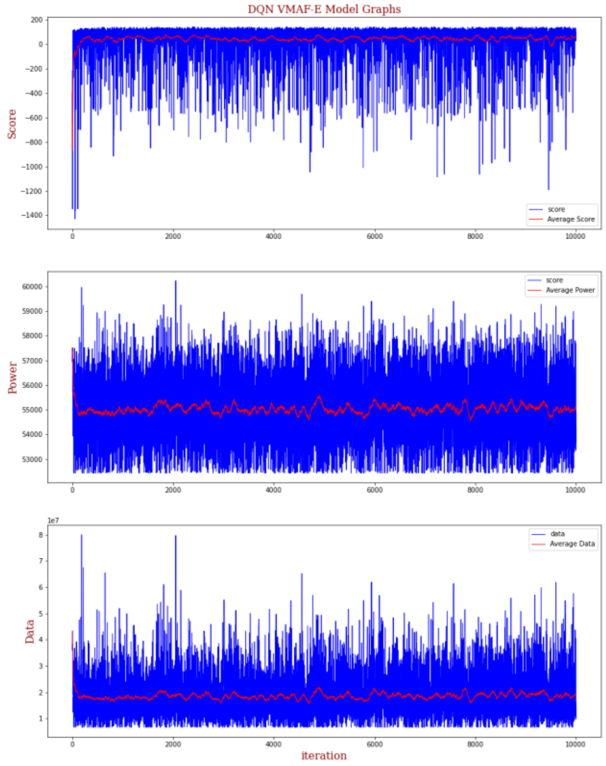
DQN Different Reward Models





DQN Different Reward Models





```
[20] avg_all= np.mean(data['score'])
     print('Average score of simple dqn(MLP) for VMAF-LN with multiple traces and s_dim=5 all = ', avg_all)
```

```
Average score of simple dqn(MLP) for VMAF-LN with multiple traces and s_dim=5 all =  82.05340882832236
```

```
[21] avg_power_all= np.mean(data['power'])
     print('Average power of VMAF-LN with multiple traces and s_dim=5 all = ', avg_power_all)
```

```
Average power of VMAF-LN with multiple traces and s_dim=5 all =  55083.385851427374
```

```
avg_data_all= np.mean(data['data'])
print('Average data of VMAF-LN with multiple traces and s_dim=5 all = ', avg_data_all)
```

```
Average data of VMAF-LN with multiple traces and s_dim=5 all =  19050886.5075
```

```
[22] avg_all= np.mean(data['score'])
     print('Average score of VMAF-E model with multiple traces and s_dim=5 all = ', avg_all)
```

```
Average score of VMAF-E model with multiple traces and s_dim=5 all =  45.427869038274004
```

```
[23] avg_power_all=np.mean(data['power'])
     print('Average power consumption of VMAF-E model with multiple traces and s_dim=5 all = ', avg_power_all)
```

```
Average power consumption of VMAF-E model with multiple traces and s_dim=5 all =  55031.87571176224
```

```
[24] avg_data_all=np.mean(data['data'])
     print('Average data consumption of VMAF-E model with multiple traces and s_dim=5 all = ', avg_data_all)
```

```
Average data consumption of VMAF-E model with multiple traces and s_dim=5 all =  18505274.2634
```

```
[25] avg_all= np.mean(data['score'])
     print('Average score of VMAF-ES model with energy in state with multiple traces and s_dim=5 all = ', avg_all)
```

```
Average score of VMAF-ES model with energy in state with multiple traces and s_dim=5 all =  45.31072084486101
```

```
[28] avg_power_all= np.mean(data['power'])
     print('Average power of VMAF-ES model with energy in state with multiple traces and s_dim=5 all = ', avg_power_all)
```

```
Average power of VMAF-ES model with energy in state with multiple traces and s_dim=5 all =  55302.963429216434
```

```
avg_data_all= np.mean(data['data'])
print('Average data of VMAF-ES model with energy in state with multiple traces and s_dim=5 all = ', avg_data_all)
```

```
Average data of VMAF-ES model with energy in state with multiple traces and s_dim=5 all =  20268641.2828
```

```
[22] avg_all= np.mean(data['score'])
     print('Average score of VMAF-E model with multiple traces and s_dim=5 all = ', avg_all)
```

```
Average score of VMAF-E model with multiple traces and s_dim=5 all =  45.427869038274004
```

```
[23] avg_power_all=np.mean(data['power'])
     print('Average power consumption of VMAF-E model with multiple traces and s_dim=5 all = ', avg_power_all)
```

```
Average power consumption of VMAF-E model with multiple traces and s_dim=5 all =  55031.87571176224
```

```
[24] avg_data_all=np.mean(data['data'])
     print('Average data consumption of VMAF-E model with multiple traces and s_dim=5 all = ', avg_data_all)
```

```
Average data consumption of VMAF-E model with multiple traces and s_dim=5 all =  18505274.2634
```

Q&A

References

- **Neural Adaptive Video Streaming with Pensieve**
  Hongzi Mao, Ravi Netravali, Mohammad Alizadeh
  Proceedings of the 2017 ACM SIGCOMM Conference