Challenges in incorporating ML in a mainstream nextgen video codec

Debargha Mukherjee
Google, LLC

[Ack: Open Codecs, Research, ML Hardware Teams in Google, Apple, Nvidia]

Outline

- Introduction
- Video Codec History
- Hardware constraints
- Current ML codecs
- Towards Practical ML
- Hybrid Codec Potential Research areas
- Make ML work Smarter
- Conclusion

Introduction

- Mainstream Video codec
  - Huge advancements in the last decade
  - HEVC, VP9, AV1, VVC, EVC
  - Getting harder to achieve gains [mostly with complexity constraints]
- ML based image/video codecs
  - Enormous advances in ML based image/video compression demonstrating their potential
  - Autoencoders, GANs, transformers
  - End-to-end trained frameworks
    - Constructive mechanism to come close to the Rate-Distortion function of a source
    - Transitioning from end-to-end image to end-to-end video compression
  - Hybrid frameworks
    - Enhance/replace certain parts of a conventional codec with ML based techniques
- How to incorporate the advances in ML compression in a mainstream codec

Video Codec History: Mainstream Video Codecs

- Historical evolution of mainstream video codecs:
  - 1991 - MPEG2 - DVD
  - 1998 - MPEG4 Part 2
  - 2003 - H.264/AVC - BlueRay, Streaming
    [End of DVD/Blue-Ray era; Enter streaming video era]
  - 2010 - VP8 [WebM]
  - 2013 - H.265/HEVC, VP9 [WebM]
  - 2018 - AV1 [AOM]
  - 2020 - H.266/VVC, EVC
  - Both AOM and MPEG are working on a nextgen codec - need gains at low enough complexity
- Typical Standardization Process
  - Cost-benefit analysis is integral part of standardization process:
    - Constant tug-of-war between hw / sw complexity and coding efficiency
    - Every tool is scrutinized meticulously and all unnecessary computations are eliminated.
  - Every new standard advances the state of hardware

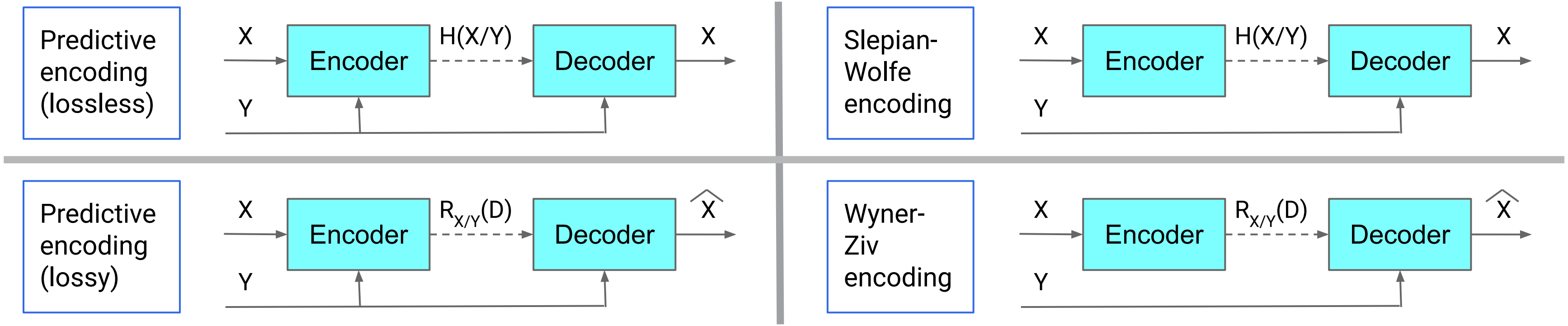
75% of all Internet bandwidth used by video today

Video Codec History: Mainstream Video Codecs Characteristics

- Characteristics of a mainstream video codec
  - Decoder:
    - Should be ubiquitous
    - Should be cheap or free
    - Should support decoding either in hw or sw at desired throughput of 4K60 or 8K30
    - Should have very small hardware footprint - silicon area very important f/ mobile chipsets
    - Should be low power
  - VOD:
    - Encoder can generally be much more complex
    - Software encoding better! but transition to HW encoders underway
  - RTC:
    - Stringent requirement for encoder + decoder together (sw only, hw only or sw/hw hybrid)
  - HW encoder has about 5x bigger budget than decoder for silicon area

Video Codec History: Distributed Video Coding

- For completeness of the discussion:
  - Reversed-complexity video coding - was very popular in the 2000-2010 decade
  - Encoder very simple - all Intra, little or no motion
  - Decoder complex - for offline decoding, or decoding with continuous feedback channel



- Practical SW and WZ coding: Source/Channel coding + Side-Info generation
- ML could be very useful in these scenarios, but not explored much

Hardware Constraints

- Time horizon for these constraints: 5-6 years
  - AOM codec expected in 2 years
  - New MPEG codec expected in about 5 years - speculative
- Also, these constraints are only for mainstream level deployment
- Niche applications /standards can be more forgiving

Hardware Constraints

- On hardware for video decoder:
  - “Hardware implementations today cannot assume availability of external GPUs/NPUs/TPUs/DSPs that could perform outside of the coding loop ML operations while the rest of the processing is done with fixed function hardware.”
    - Source: **Google**, *“Recommendations for HW friendly ML-based tools in AV2,”* CWG-073, Alliance for Open Media, Codec Working Group
  - **Apple**, **Nvidia** - similar thinking on preferring dedicated HW IP for video decoding
  - Throughput: Today’s trend at least 4K60 or 8K30
    - Silicon area needed may be lower if throughput needed is lower (images) but not by a lot!
  - What matters most is the amount of raw computation (operations) needed in the decoder.
    - Memory Bandwidth, RAM size, etc. - are also important but architecture dependent
    - Secondary still to raw compute

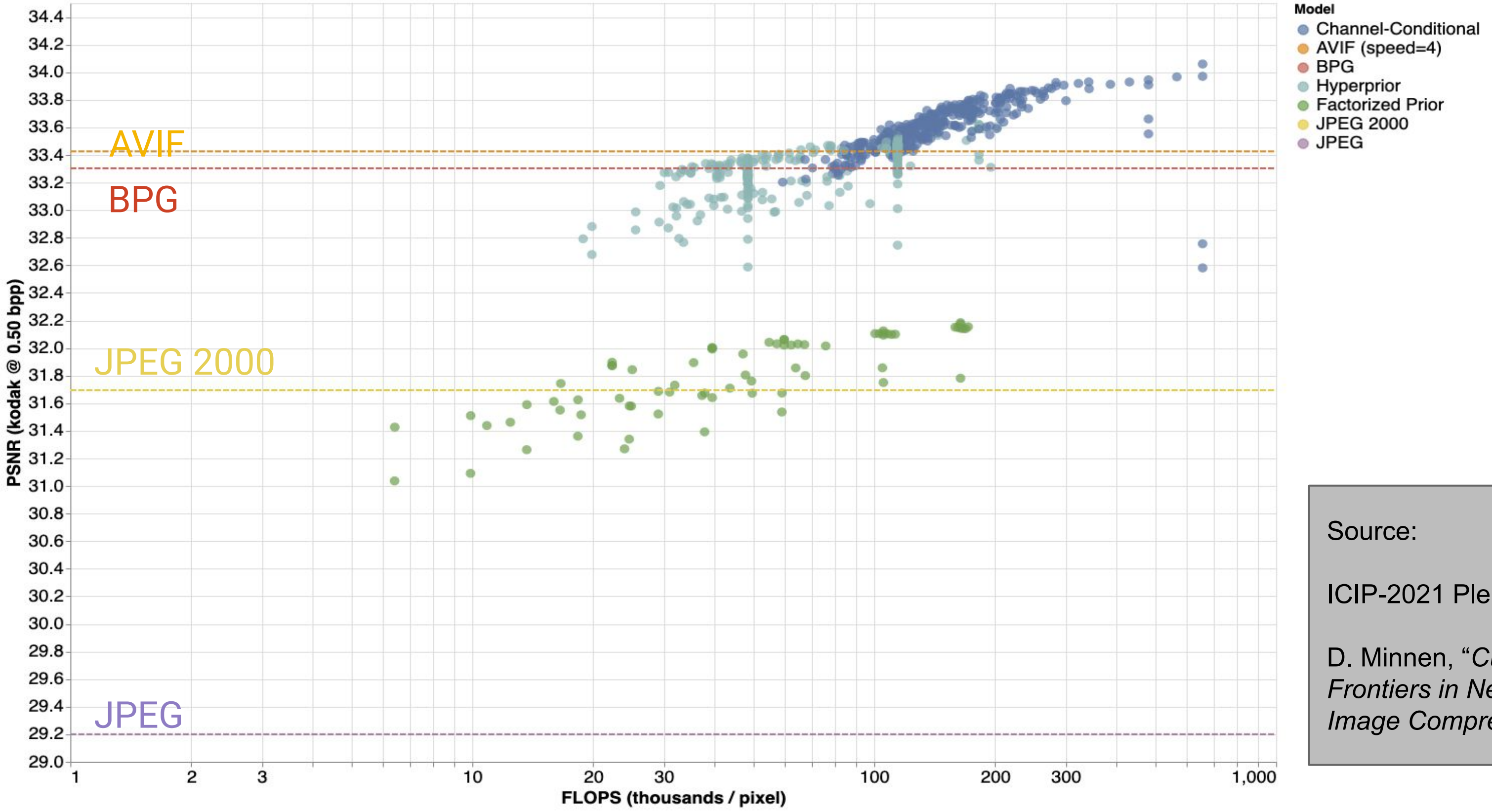
Hardware Constraints

- Disclaimer: Rough math, but okay for a general idea.
- Estimate of int8-MACs/pixel based on silicon area for an entire decoder
  - A full AV1 decoder needs lower ops/pixel than a modest MobilenetV1 network
  - Best equivalent estimate for a full AV1 decoder in ops/pixel is **~4K** int8-MACs/pixel
- Nextgen codec typical target:
  - Historically the nextgen increase in decoder silicon area is less than 200%
  - About 30-40% BDRATE reduction
- A per coding efficiency gain budget:
  - Starting from a state of the art video codec today, every 1% BDRATE gain can only have **~100-200** ops / output pixel.
  - Lenient estimate due to unavoidable common tasks. Real number is more like **<50**.
- HW Encoder can be 5-10 times more in silicon area than the decoder
  - Historically 10x increase from one generation to the next

Hardware Constraints

- Apple:
  - Source: **Apple**, *"Recommendations for HW friendly ML-based tools for AV2,"* CWG-082, Alliance for Open Media, Codec Working Group
  - Decoder side int8-MACs/pixel: D = 1000 * BDRATE / 30
    - Only 33 int8-MACs/pixel per 1% BDRATE gain is acceptable
  - Encoder side int8-MACs/pixel: E = 4 D
- Google/Nvidia:
  - Similar - may be a marginally higher

Current ML codecs



Source:

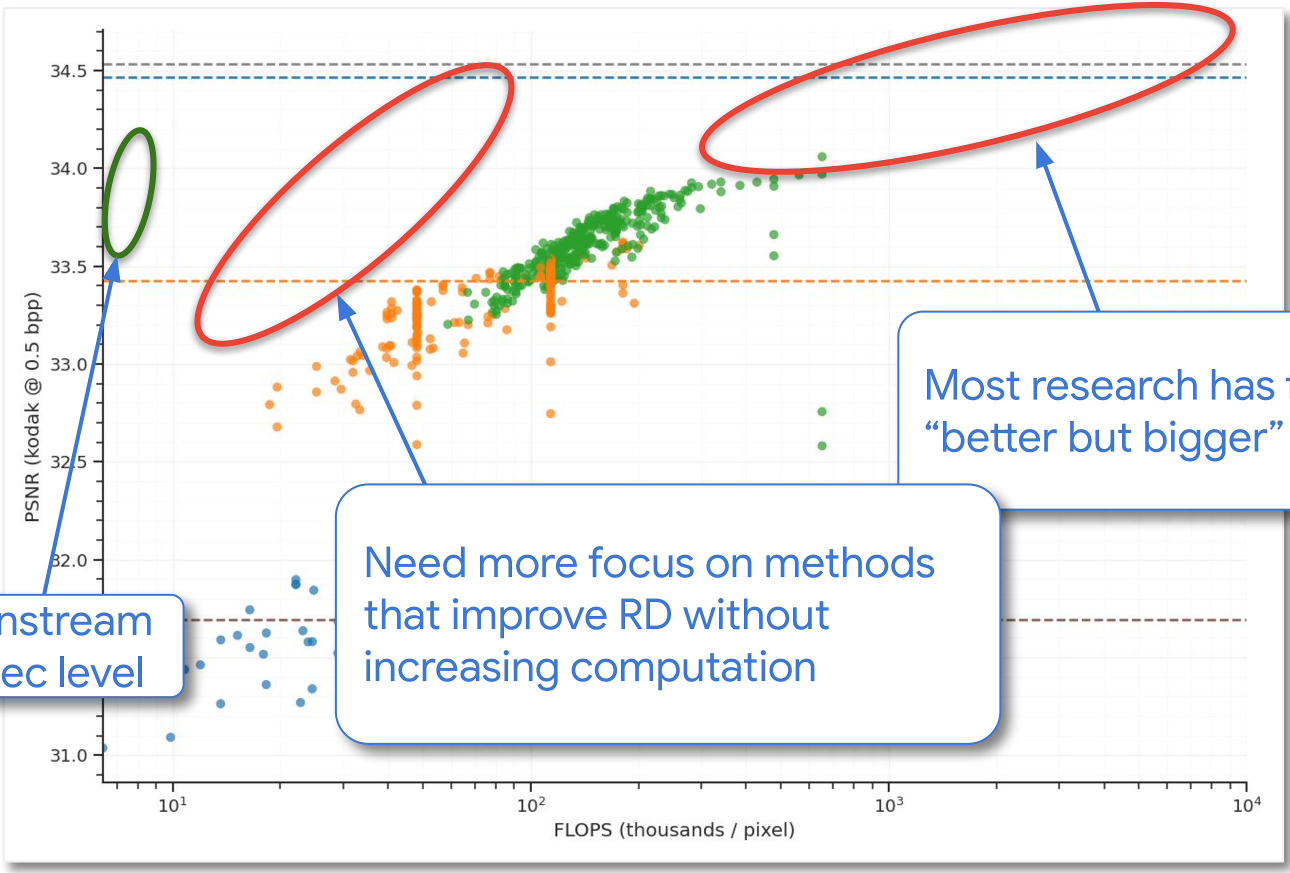
ICIP-2021 Plenary

D. Minnen, "*Current Frontiers in Neural Image Compression*"

Current ML codecs

- The best Learned codecs so far:
  - Learned Image codec: ~**50 - 500K** MACs/pixel
  - Learned Image codec with GAN loss: **~500K+** MACs/pixel
  - Learned Video codec: Possibly higher!
    - Models usually do the equivalent of motion on decoder side
- Huge discrepancy:
  - 500-1000x
  - ML tools need to be sub-1K MACs/pixel to be even within an order of magnitude of the target
- But mainstream codecs do need the gains from ML advances
  - Conventional codec development is getting to be tedious - like finding special-cases, and special-cases of special-cases.

Current ML Codecs



Mainstream codec level

Need more focus on methods that improve RD without increasing computation

Most research has focused on "better but bigger" models

Adapted from:

ICIP-2021 Plenary

D. Minnen, "*Current Frontiers in Neural Image Compression*"

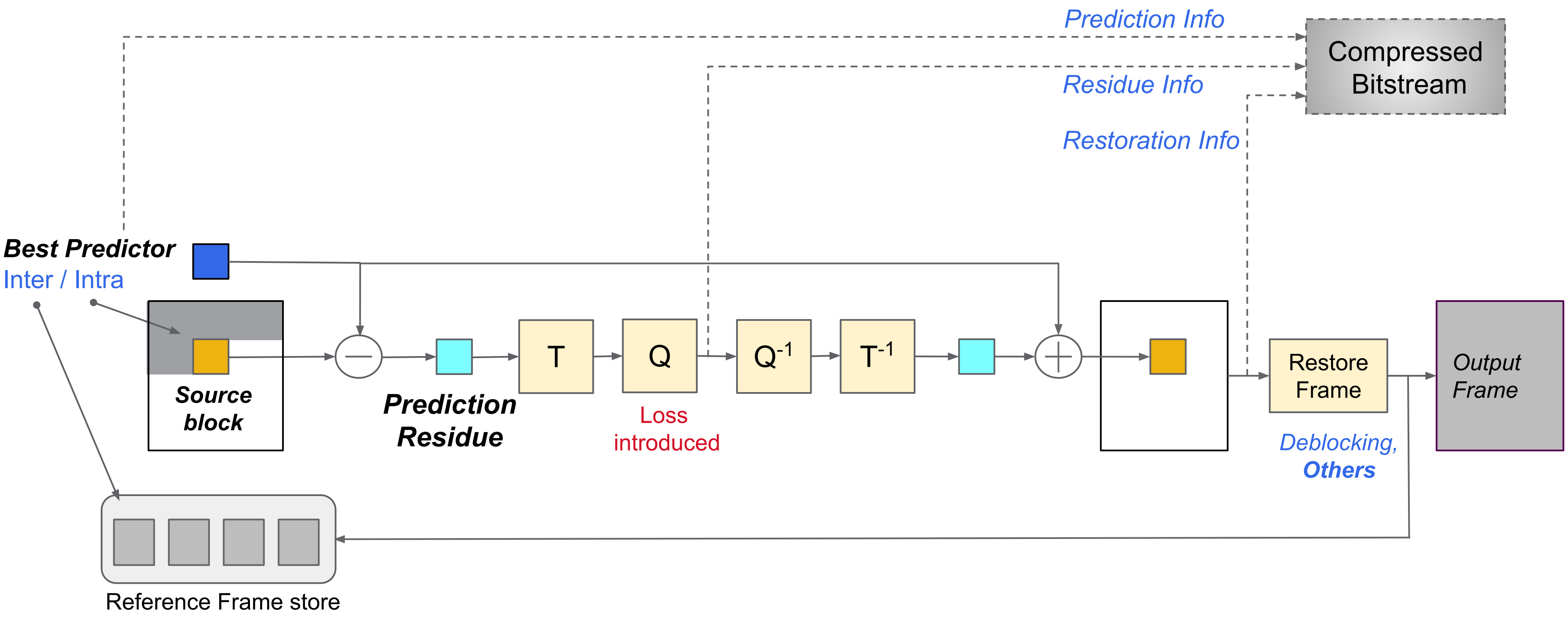
Towards practical ML

- Philosophy
  - Incremental gains in incorporating ML in a mainstream codec pipeline is okay!
  - Mainstream codecs are snapshots in time
    - State of evolution of hardware in video codecs is often incremental
    - Even if a simple but substantial ML tool gets into a mainstream nextgen codec, it makes it easier for a bigger tool in the next edition.
  - Need focus on hybrid techniques that add ML tools in a conventional video coding pipeline and satisfies the same cost-benefit trade-off as any other tool.
    - Cost-effective ML tools that give only 1-3% BDRATE gains are okay
  - A great deal of room for innovation in lightweight models
    - Not explored enough

Towards practical ML

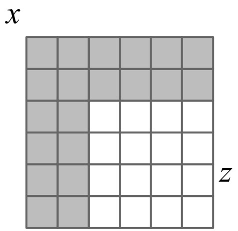
- Need RDC metrics that incorporate model complexity
  - Use a score that uses model complexity as a parameter in addition to BDRATE
  - Slowly move the community to look at lightweight models
  - A new track for CLIC 2023 ?
- Better understanding of what exactly a ML model is doing to reduce compute.
- Find places in a conventional codec pipeline where ML can be substantially beneficial over standard signal processing methods.
  - ML for finding the best prediction (inter, intra, inter/intra)
  - Nonlinear transforms for prediction residues
  - In-loop filtering likely has the best potential in the short term
    - Restore frames with ML after decoding through a conventional pipeline
  - Super-resolution: Both in-loop and out-of-loop
- Use ML to work on a smaller subspace (high-freq, low-res, mid-res)

Hybrid Codec Potential Research areas: Conventional Codec

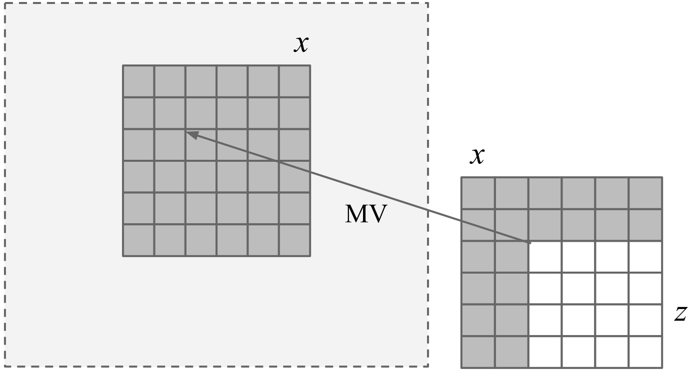


Hybrid Codec Potential Research areas: Prediction

- Improve prediction using ML
  - Intra prediction - In-painting
  - Inter-Intra prediction - a useful generalization
  - Multimode predictor design is critical
    - Modern codecs have multiple signaled predictors
    - Need unsupervised multimode design



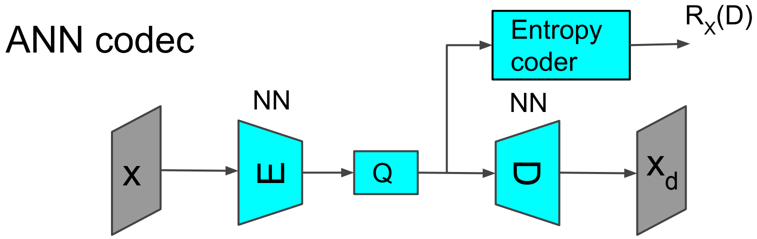
Intra Prediction

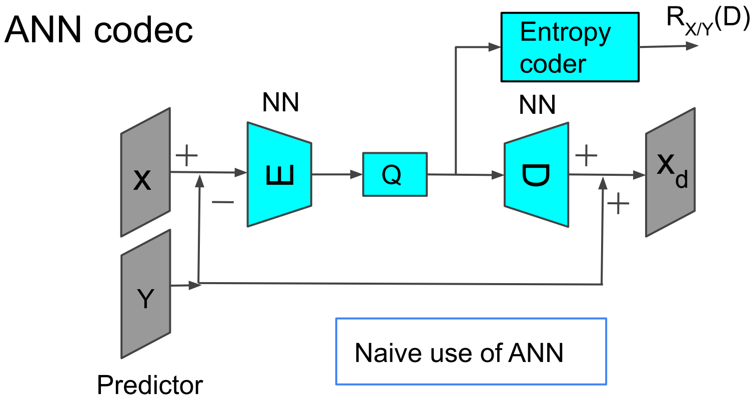


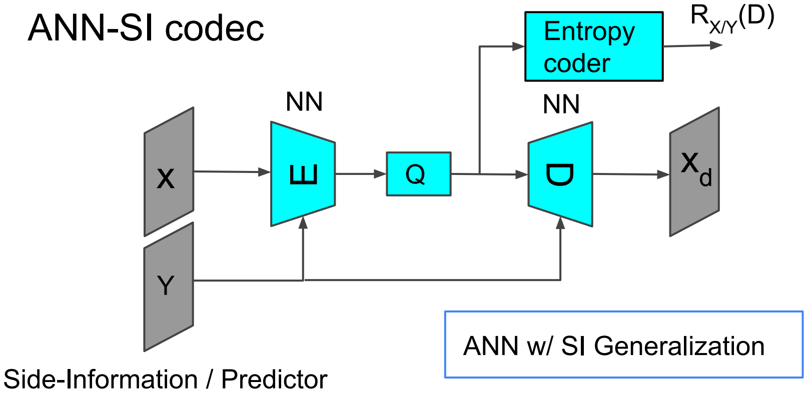
Inter-Intra Prediction

Hybrid Codec Potential Research areas: Residue Coding

- Use Learned Image Compression ideas for coding prediction residues
- Autoencoder with Side-Information (ANN-SI/Conditional Autoencoder)
  - Side-Information could be any predictor





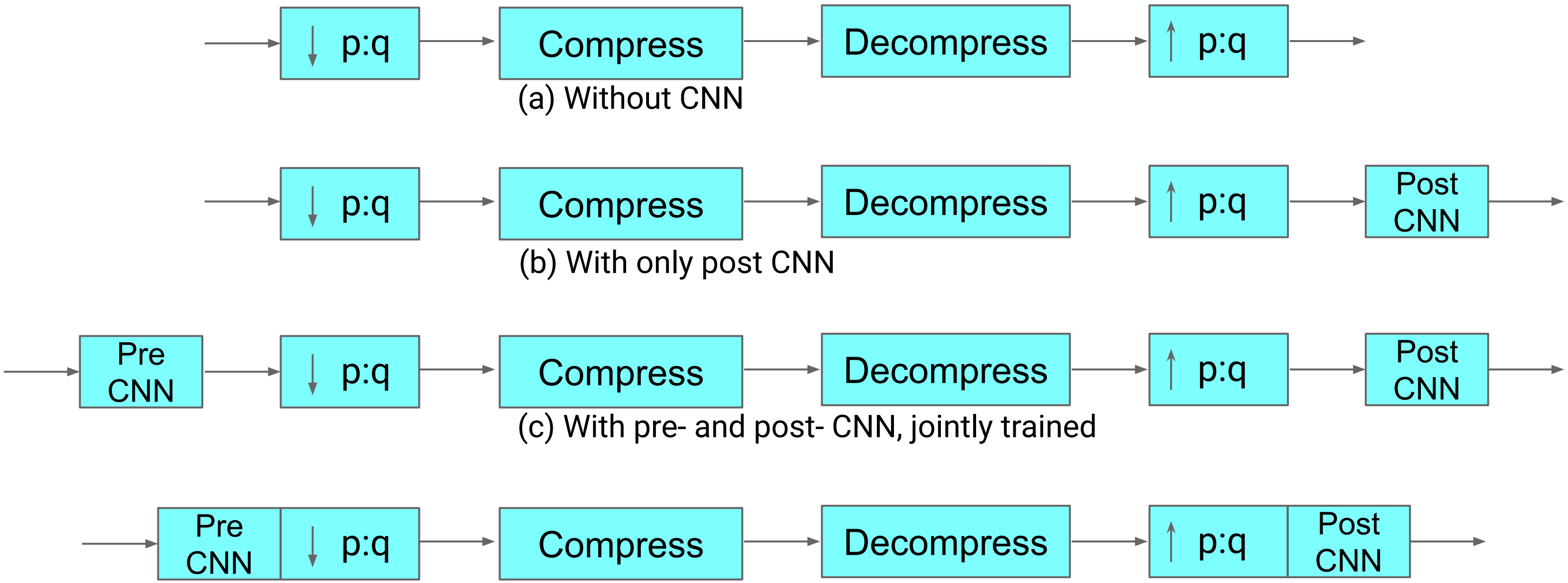


Hybrid Codec Potential Research areas: Loop filtering

- In-loop-filtering
  - Probably the biggest potential of all
  - Some recent work beyond AV1 and VVC
    - 3-5% BDRATE gain at about 15-40K FLOPS/pixel;
      6-8% BDRATE gain at 100k+ FLOPS/pixel
  - Need focused effort to reduce the ops down to sub-1000
  - Easier to integrate at the end of the pipeline
- Some experimental findings
  - With a model with 20K MACs/pixel: BDRATE ~3%
  - With a model with 600 MACs/pixel: BDRATE ~1%
- Need novel architectures with low ops and perhaps closer connection with standard signal/image processing methodologies

Hybrid Codec Potential Research areas: Super-resolution

- A great deal of interest in <downsample - compress - upsample> pipeline



(a) Without CNN

(b) With only post CNN

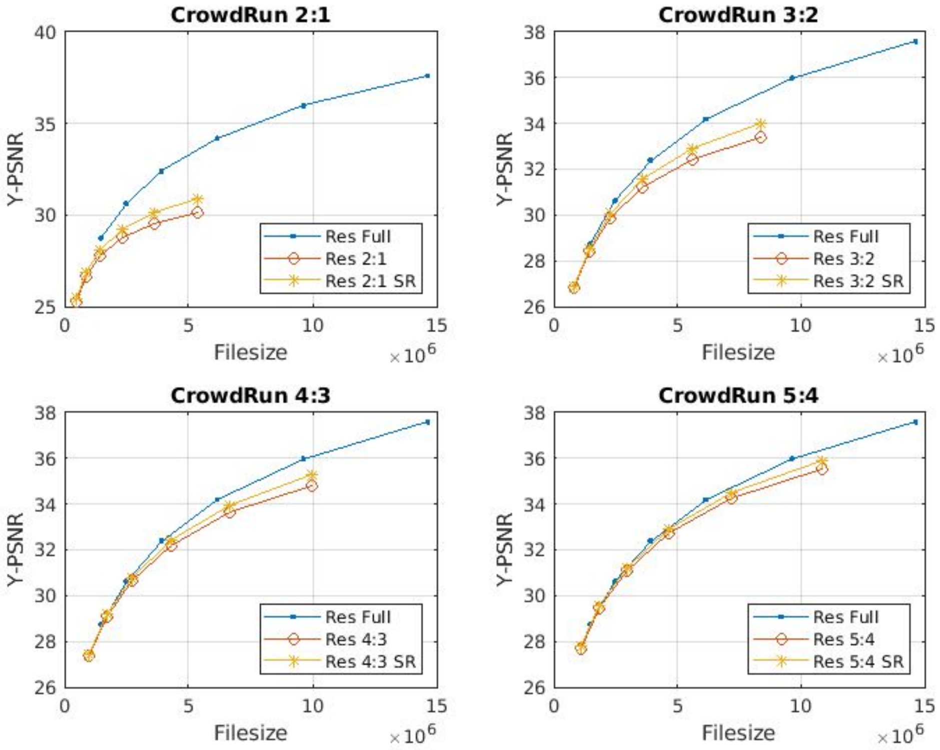
(c) With pre- and post- CNN, jointly trained

(d) For q=1, pre and post CNN can include down and upsampling

Hybrid Codec Potential Research areas: Super-resolution

- CrowdRun
  - BDRATE [2:1] = -8.84%
  - BDRATE [3:2] = -6.23%
  - BDRATE [4:3] = -4.76%
  - BDRATE [5:4] = -4.39%
- What matters is convex hull
- Smaller models seem to be sufficient for super-resolution

2K #params; 2K MACs/pixel

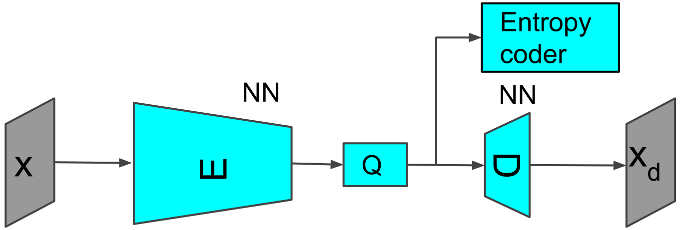


Make ML work Smarter: Some General strategies

- Multiplier depth and quantization optimization
  - QKeras: https://github.com/google/qkeras
    - Many layers in a deep neural network can do just as well with low-bit-depth multipliers
  - Vizier: https://cloud.google.com/ai-platform/optimizer/docs/overview
    - Black-box parameter search
- Use more processing at lower resolution
  - U-Net type architecture preferred
- Use origin symmetric kernels
  - Reduces multiplies by ~ a factor of 2
- Use networks where the initial layers are integer analysis filterbanks
  - Haar, Integer wavelets, etc.

Make ML work Smarter: Change balance

- Change balance of computation between Encoder and Decoder networks
  - HW constraints are more stringent on decoder side
    - So try to reduce decoder side network size
  - Becomes similar to Vector Quantization

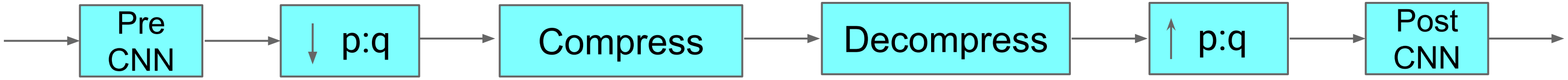


Can be bigger

Make smaller

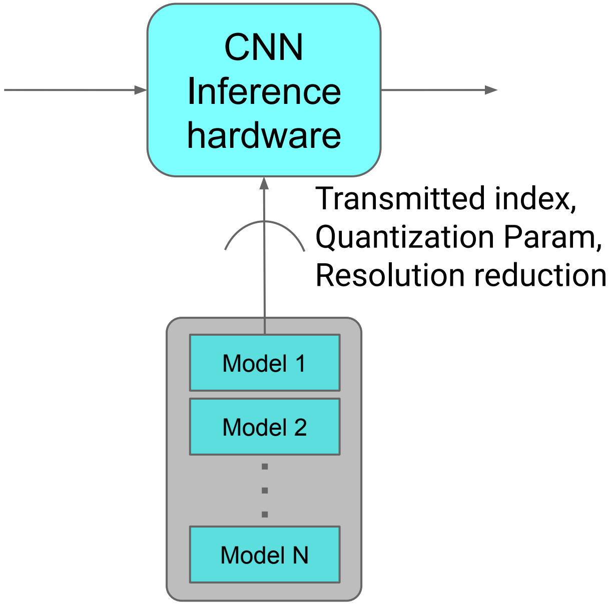
Can be bigger

Make smaller



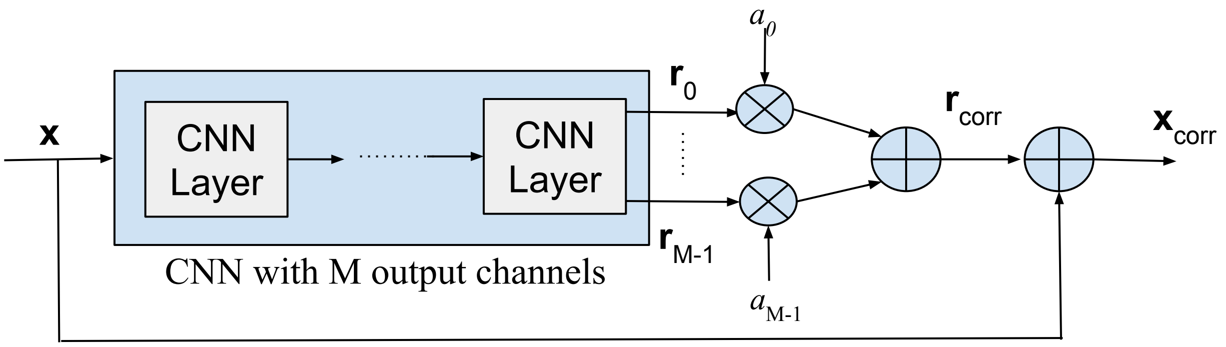
Make ML work Smarter: Switchable Networks

- Switchable models on common hardware
  - Use one of N different models based on information already available, classification based on image characteristics and/or encoder-side search, and signaling of the model index in the bit-stream
  - The specific model is downloaded to decoder hardware at decoding time at a suitable interval
    - Frame/tile level switching is okay
  - Video chipsets already designed to store large frame buffers; storing/retrieving multiple models not hard.
- Each instance of inference is low complexity
- Multimode design needed
  - Unsupervised clustering-cum-training can be useful



Make ML work Smarter: Guided Networks

- Guided CNN
  - Certain weights and layers of a pre-trained CNN are signaled by encoder to decoder
    - Benefit: Each inference instance can be lower complexity
  - Particularly convenient if the last layer weights are signaled to decoder
  - CNN learns to produce M-channel output in a suitable subspace (w/ modified loss function)
    - Weights for combination are signaled to decoder at a suitable interval
- CNN followed by Wiener filter is a special case of this formulation



Explicitly signal linear combination weights (a_0, a_1, …) of output channels

Conclusion

- Mainstream video codecs
  - Further advancement becoming tedious (still possible without complexity constraints)
  - Very stringent requirements on decoder side complexity for cost-effectiveness
- ML codecs
  - Potential has been adequately demonstrated in recent years
  - May be okay for niche applications from a complexity perspective
  - But… big gap in complexity/cost requirements for mainstream deployment in the next 5-10 years
- ML Research focus:
  - Too much focus on demonstrating potential with bigger and more complex models
  - Too little focus on bringing ML advances into the mainstream domain by optimizing compute
    - Need more focus on: what can we do with smaller/lighter models ?
    - New research opportunities in hybrid approaches with traditional signal processing