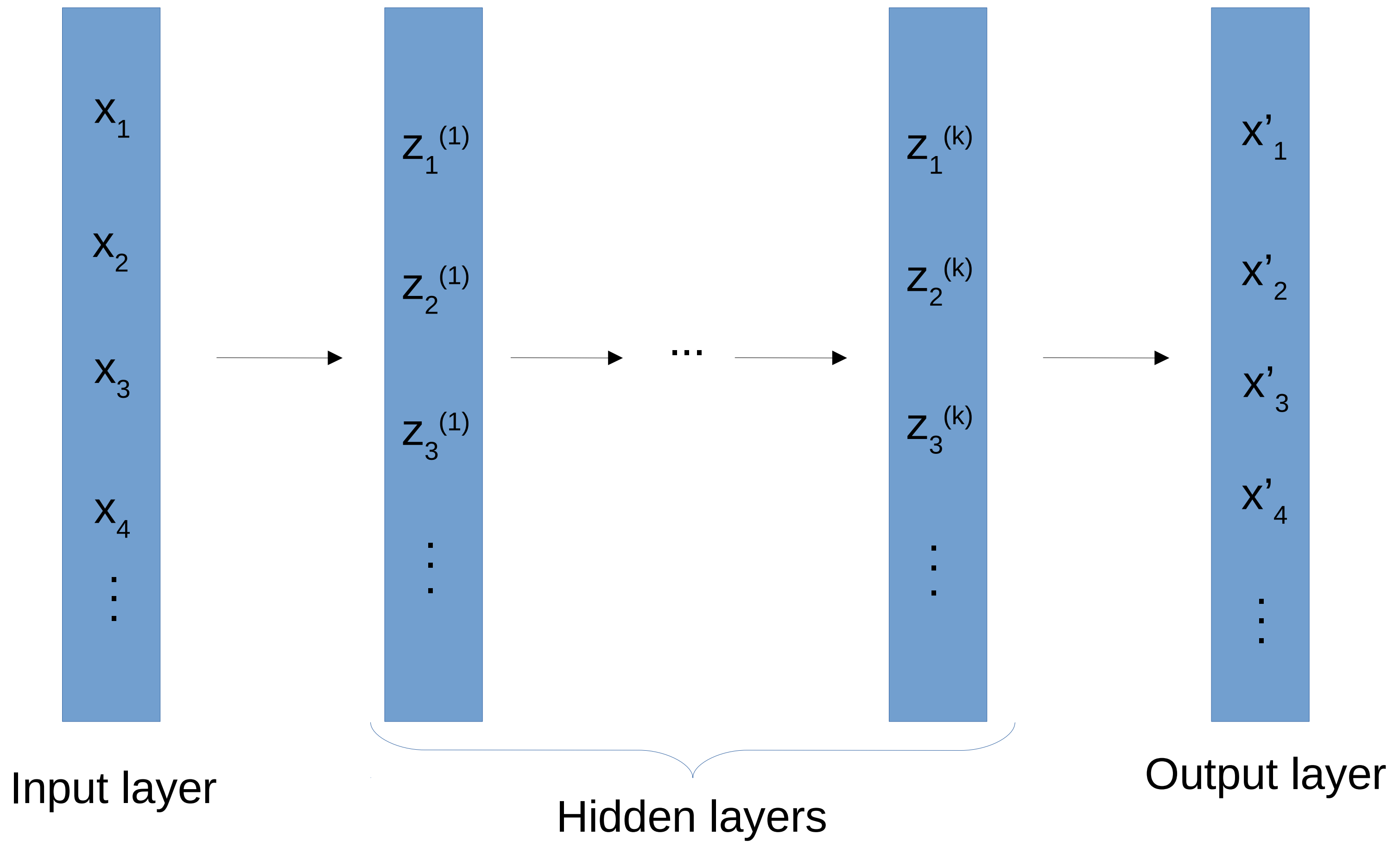
Deep learning based image compression

Li Wang, Giuseppe Valenzise

Laboratoire des Signaux & Systèmes

CentraleSupélec

What is Autoencoder ?



Autoencoder : $x_i = x_i'$

What is Autoencoder ?

- Encoder stage : map the input x to z

$$z=\delta(Wx+b)$$

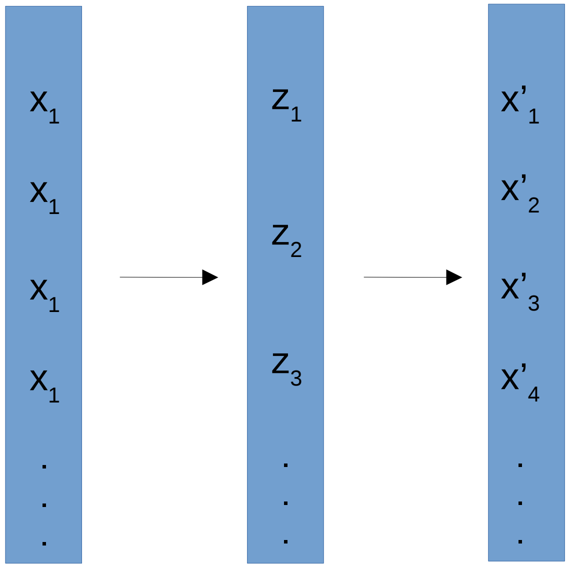
δ : element-wise activation function

- Decoder stage : map z to reconstruction x’

$$x'=\delta'(W'z+b')$$

- Autoencoders are trained to minimise :

$$L(x,x')=\|x-x'\|^2$$



What is Autoencoder ?

- Denoising autoencoder

$$x \xrightarrow{\text{Corrupt data}} \tilde{x} \rightarrow \boxed{\text{Autoencoder}} \rightarrow \tilde{x}'$$

$$min\, L(x, \tilde{x}')$$

- Sparse autoencoder

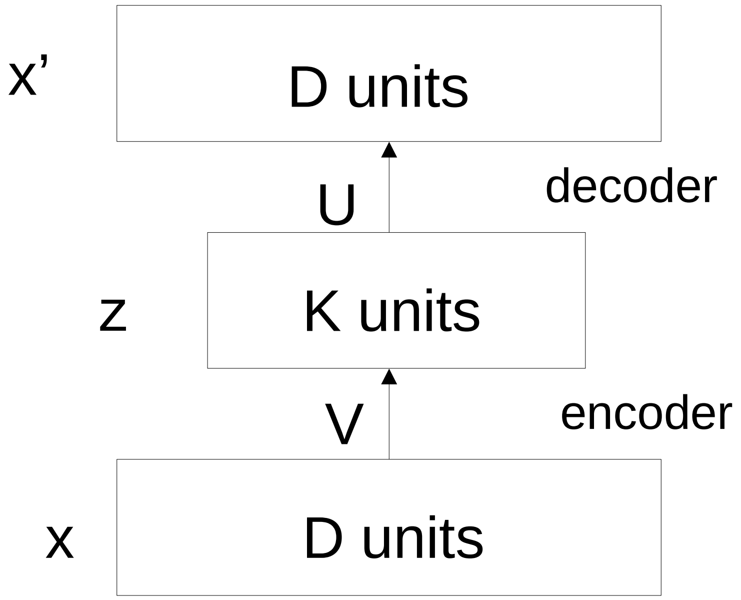
  Imposing sparse criterion on hidden units

- Variational autoencoder

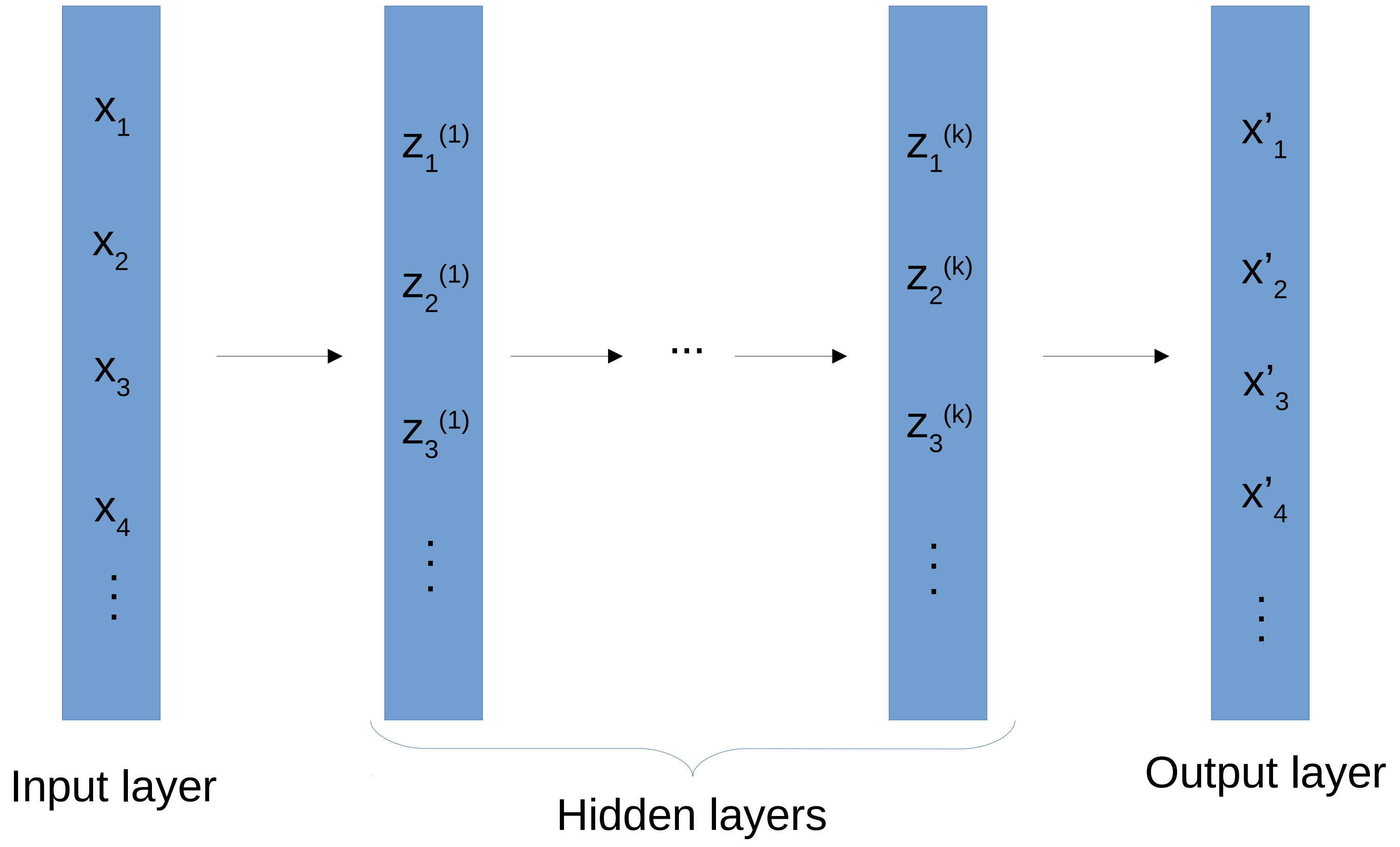
  Use a stochastic gradient variational Bayes algorithm for training

Principal Component Analysis (PCA)

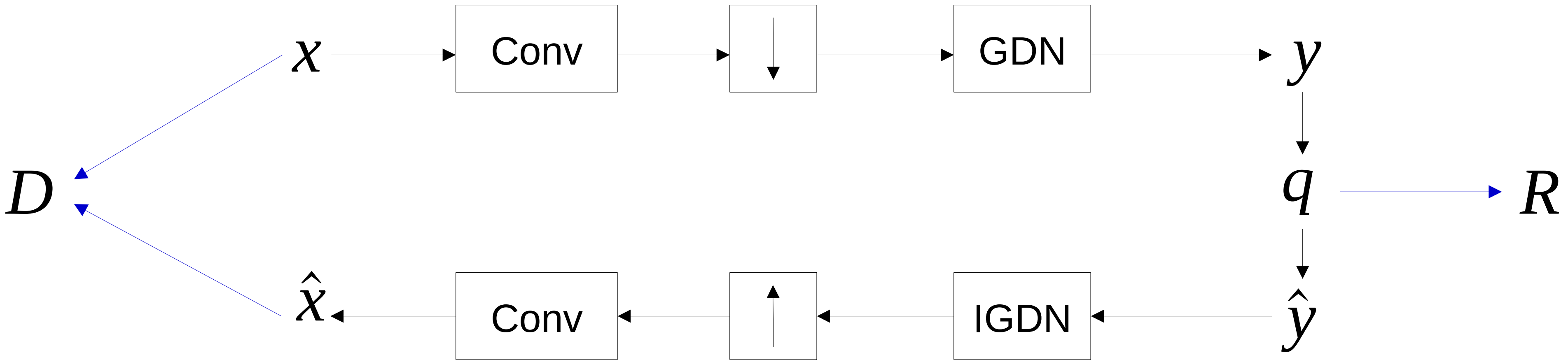
- Linear autoencoder
- One hidden layer
- Squared error loss



Deep Autoencoder



[Ballé et al. 2017]

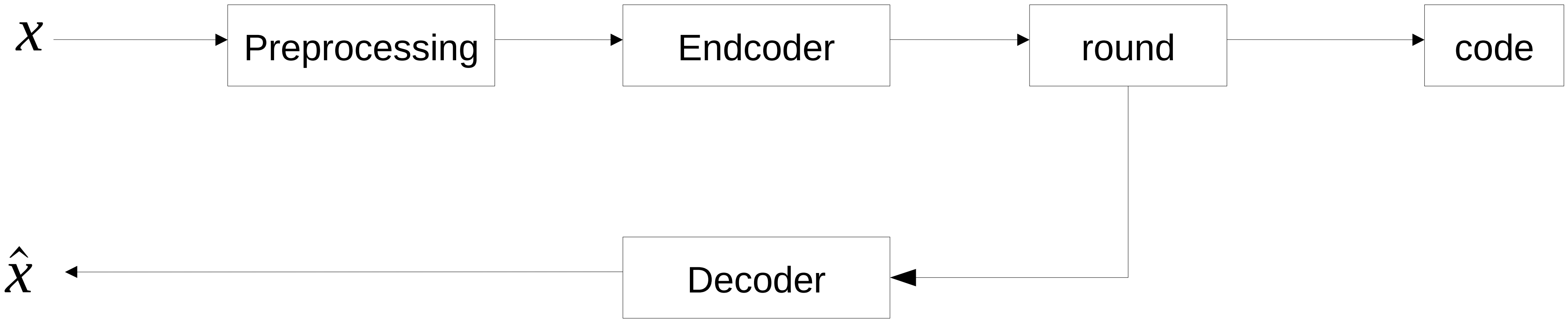


GND : generalized divisive normalization

Simulation results [Ballé et al. 2017]

JPEG	JPEG2000	Proposed
		
R:0.950 MS-SSIM: 0.9909	R:0.996 MS-SSIM: 0.9856	R:0.654 MS-SSIM: 0.9878
		
R:0.102 MS-SSIM: 0.8123	R:0.093 MS-SSIM: 0.8638	R:0.082 MS-SSIM: 0.9133

[Theis et al. 2017]



Quantization : rounding to the nearest integer

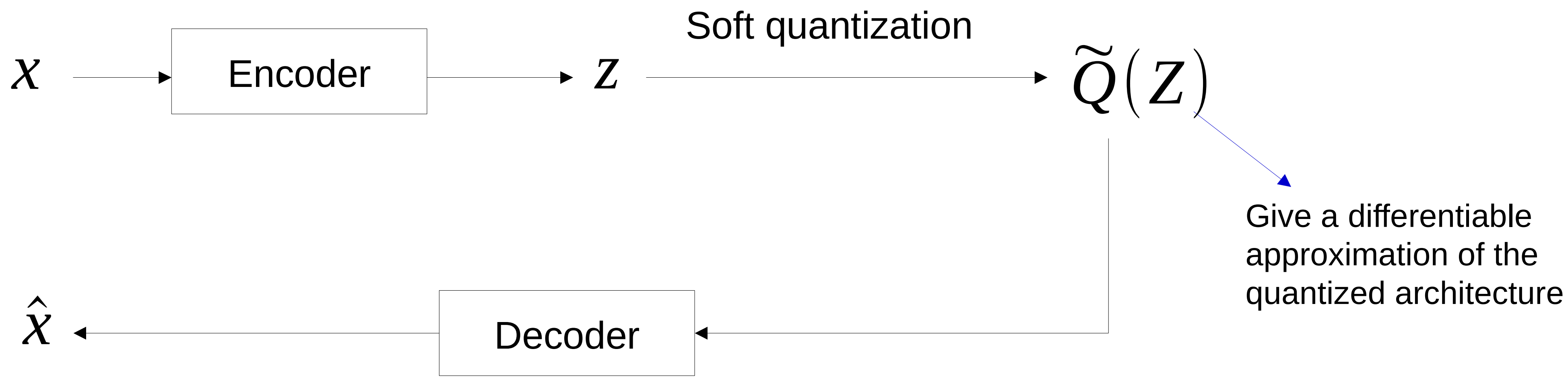
Simulation results [Theis et al. 2017]

Proposed	JPEG 2000	JPEG

0.245972 bpp	0.250468 bpp	0.248413 bpp

0.245626 bpp	0.249654 bpp	0.254415 bpp

[Agustsson et al. 2017]



Simulation results [Agustsson et al. 2017]

JPEG

0.22bpp
PSNR=19.77dB

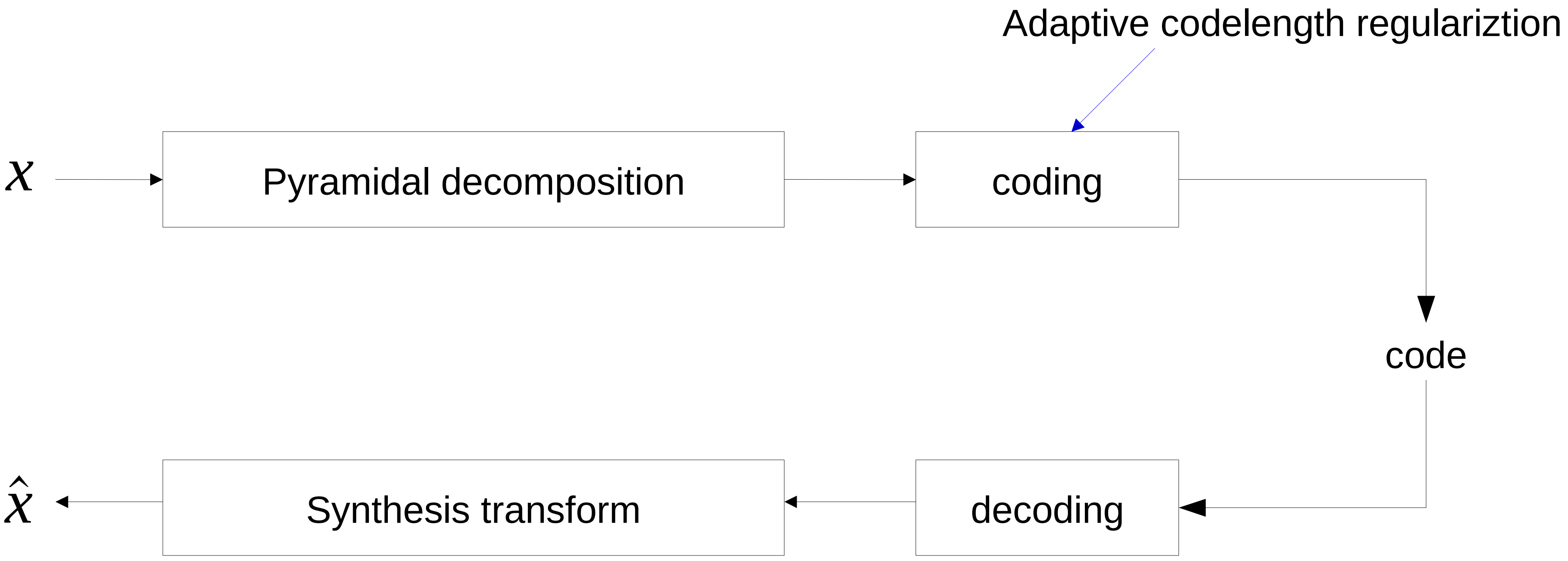
JPEG 2000

0.2bpp
PSNR=23,01dB

Proposed

0.2bpp
PSNR=23.88dB

[Rippel et al. 2017]



Simulation results [Rippel et al. 2017]

JPEG | JPEG 2000 | Proposed

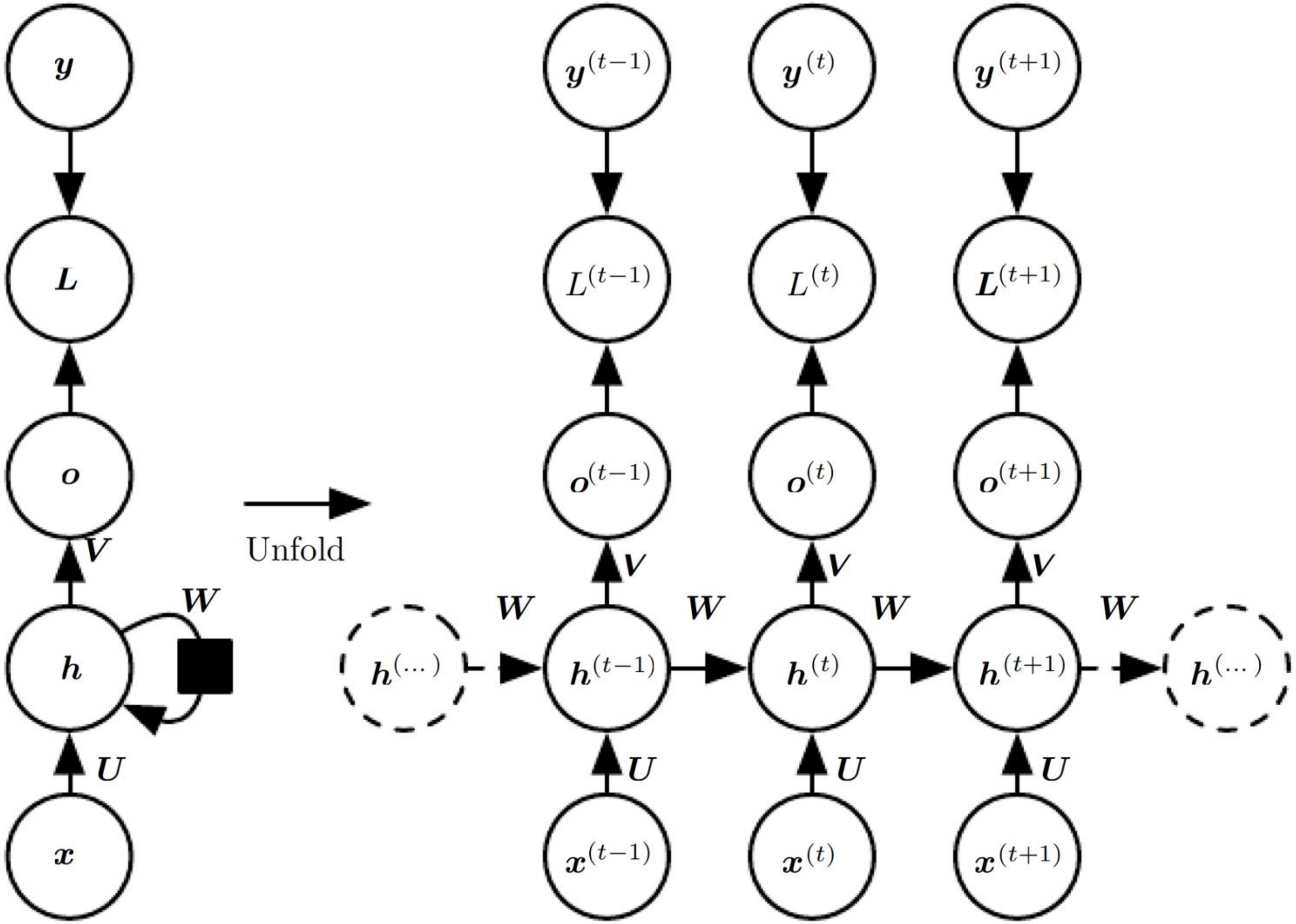


0.0909bpp | 0.0847bpp | 0.0840bpp



0.0949bpp | 0.0941bpp | 0.0928bpp

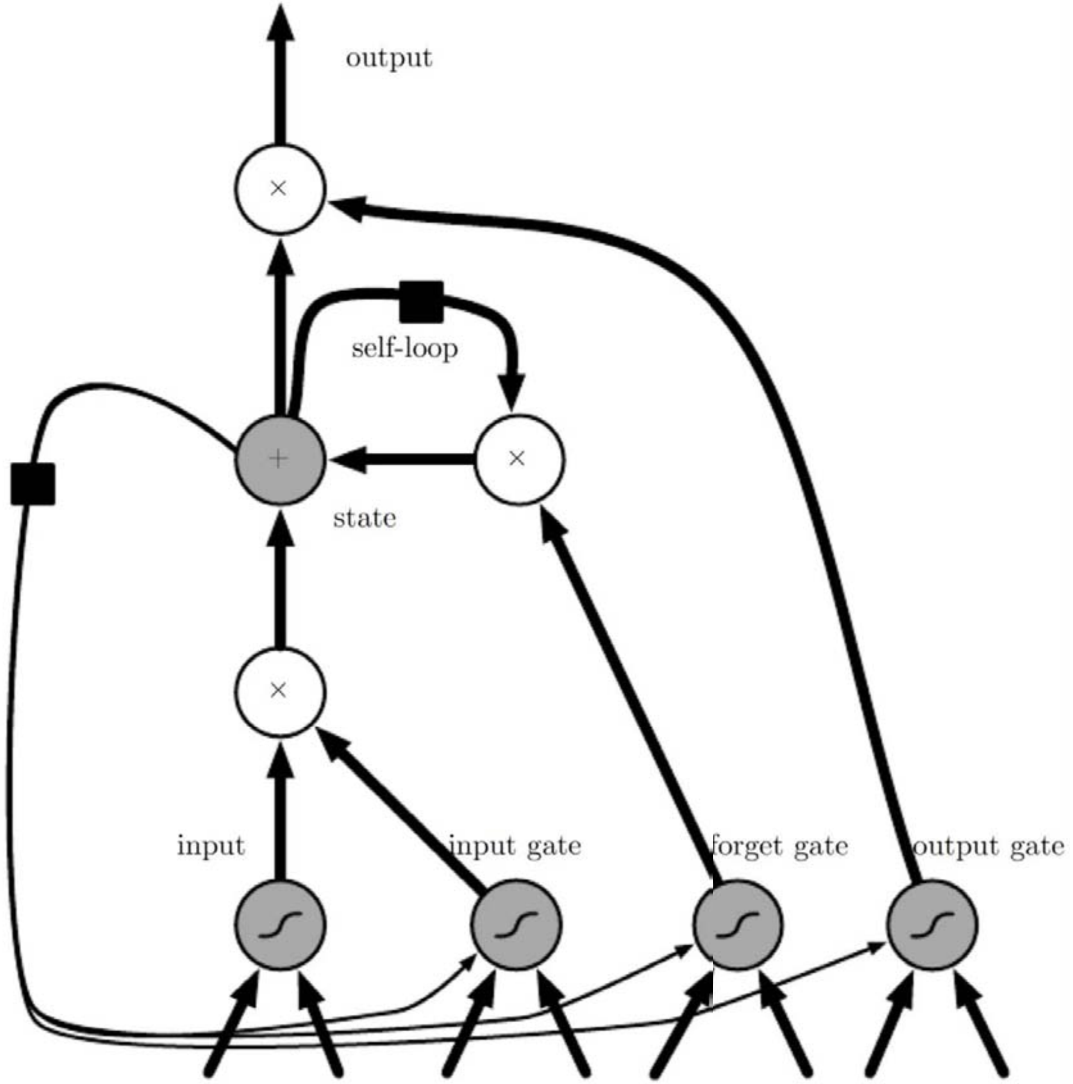
Basic tools: Recurrent Neural Networks (RNN)



[Goodfellow et al., Deep Learning, 2016]

RNN: the challenge of long-term dependencies

- RNNs are similar to dynamical systems
- Problem: propagation of gradients during learning
- Common solutions:
  - Resnet units (for vanishing gradients)
  - Clipping (for exploding gradients)
  - **Long-short term memory units (LSTM)**: enable to store and forget the current state



[Goodfellow et al., Deep Learning, 2016]

Variable rate image compression with recurrent neural networks (Todorici et al. 2016)

- Variable rate is achieved using progressive encoding
  - Residuals are progressively encoded on top of previous residuals

- Chain of multiple copies of a residual auto-encoder F_t:

$$F_t(r_{t-1}) = D_t\left(B\big(E_t(r_{t-1})\big)\right)$$

  Where r_0 is the original patch

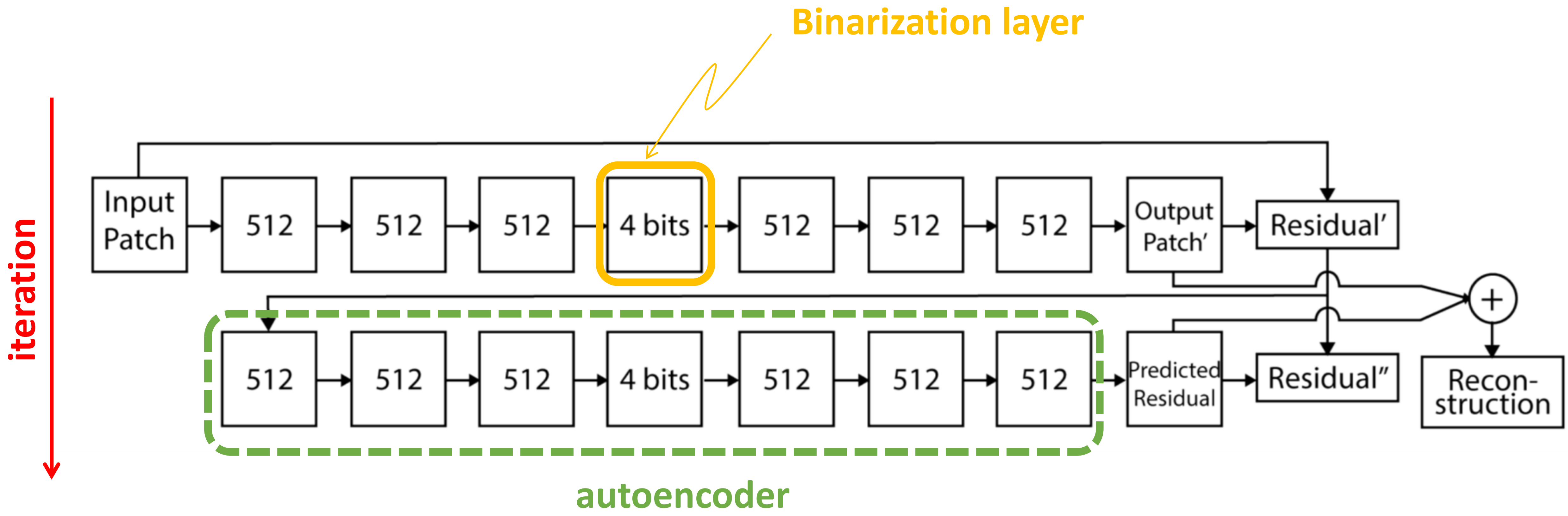
- Loss function:
  - L2 norm
  - W.r.t residual, for non-LSTM architectures
  - W.r.t. original patch, for LSTM architectures

Binarization

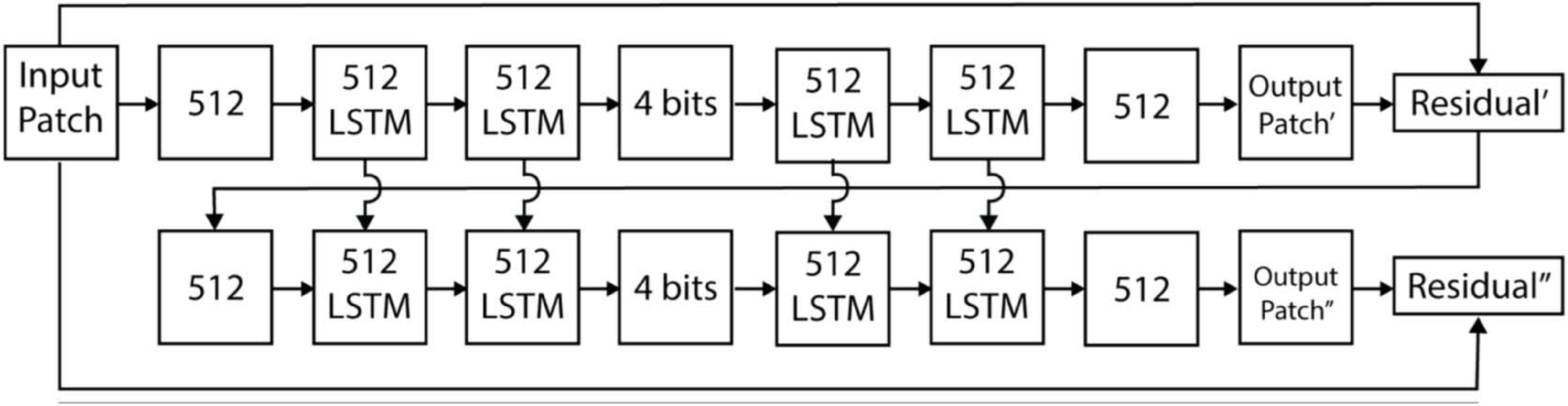
- Fixed number of output bits n
- Two steps:
  - Generate n outputs in [-1, 1] using a fully connected layer with tanh activation
  - Thresholds the outputs

- Total number of bits:
  - Number of outputs n times the number of repetitions of the residual autoencoder structure

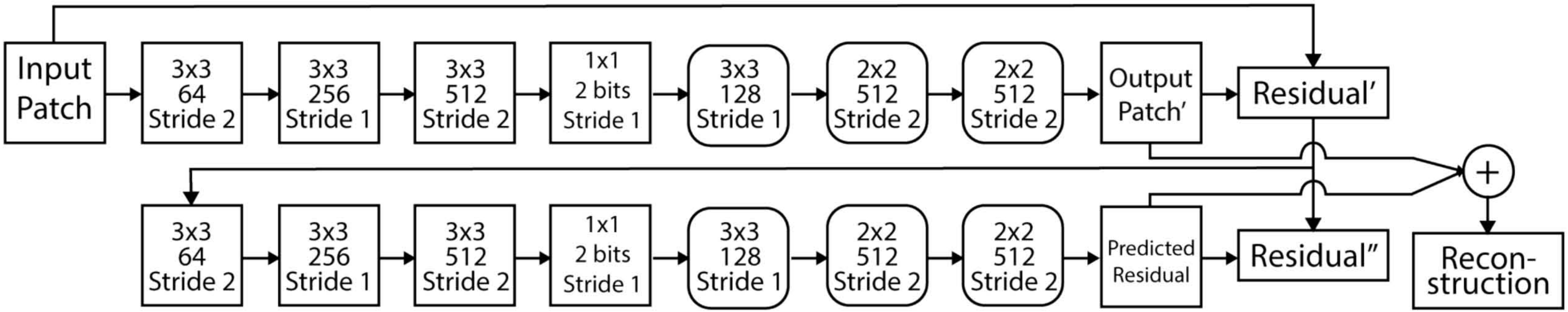
Feed-forward fully connected residual encoder



Fully connected LSTM residual encoder



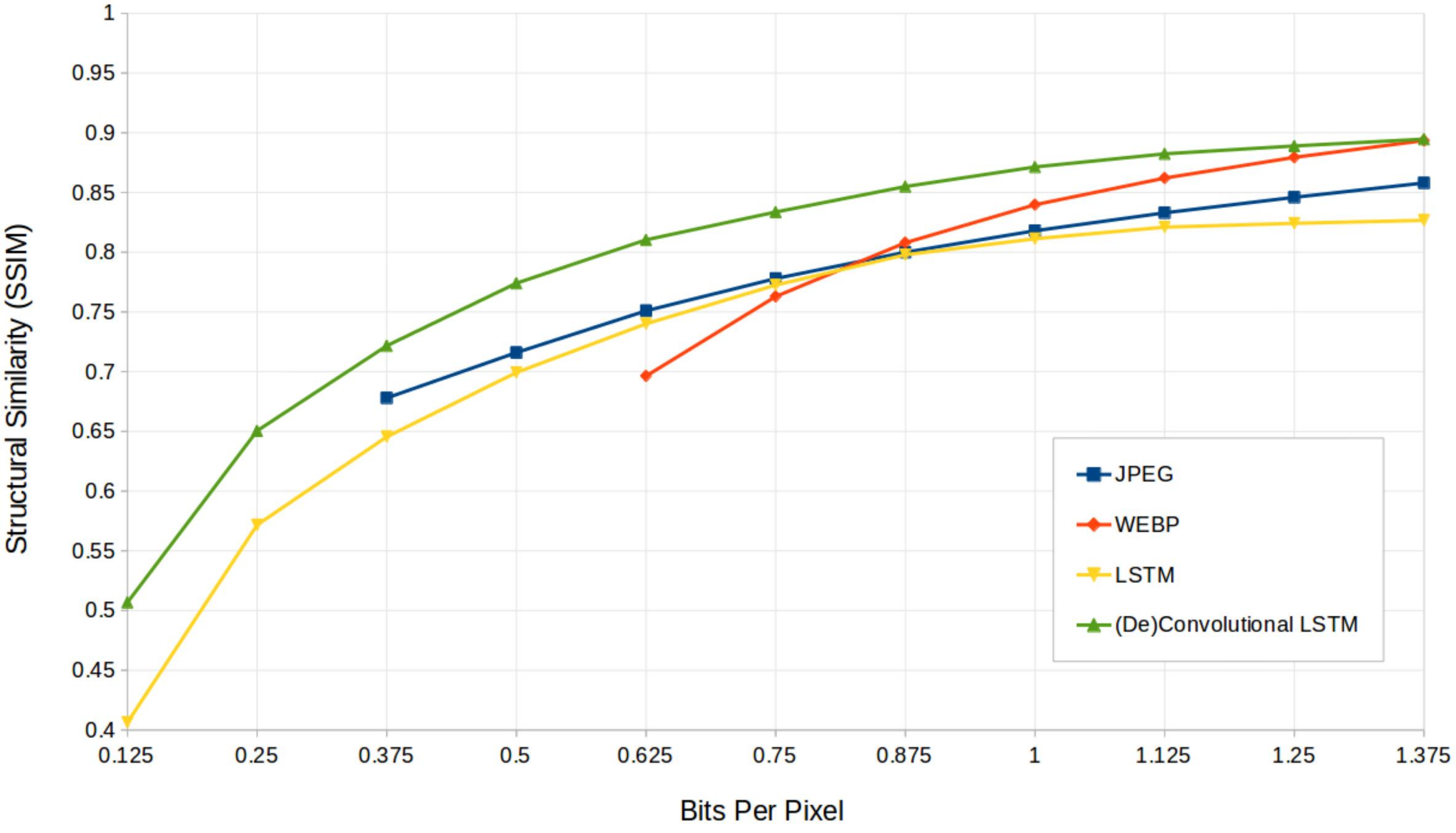
Feed-forward convolutional/deconvolutional residual encoder



Training & results

- 32x32 patches (216 millions)
- No perceptual loss function in the training

- Evaluation using SSIM
- Best architectures: LSTM and convolutional LSTM


1
0.95
0.9
0.85
0.8
0.75
0.7
0.65
0.6
0.55
0.5
0.45
0.4
Structural Similarity (SSIM)
0.125
0.25
0.375
0.5
0.625
0.75
0.875
1
1.125
1.25
1.375
Bits Per Pixel
JPEG
WEBP
LSTM
(De)Convolutional LSTM

Original (32×32)

JPEG compressed images

WebP compressed images

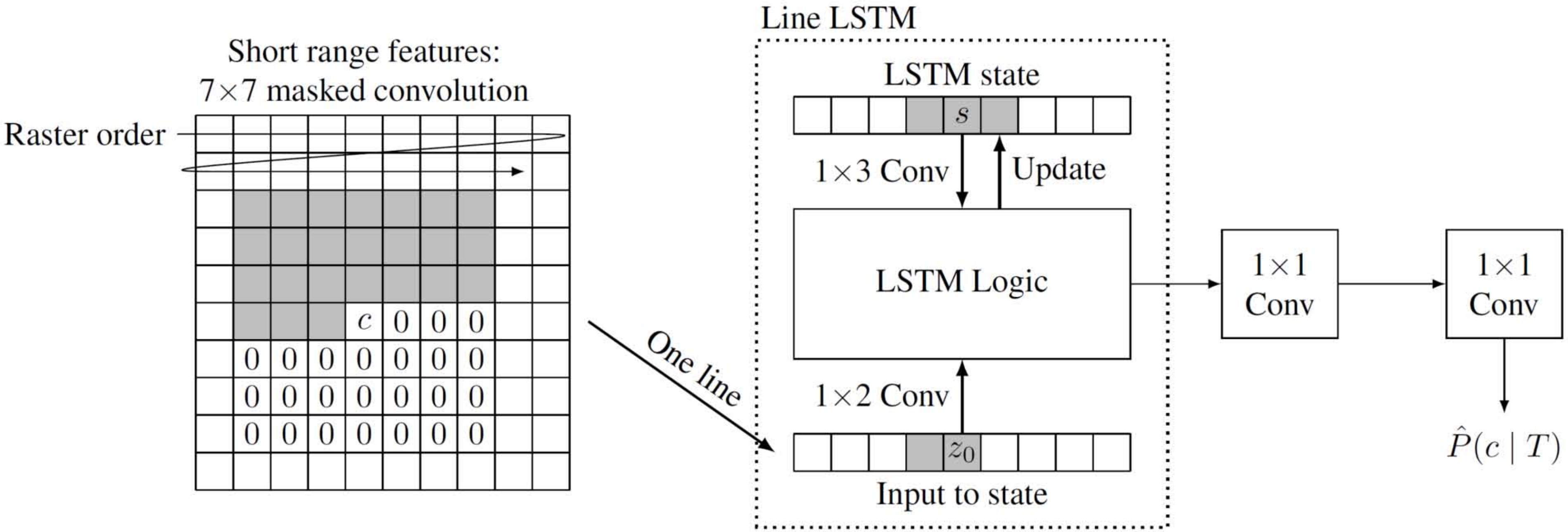
	From left to right [bpp]			
JPEG	0.641	0.875	1.117	1.375
WebP	0.789	0.914	1.148	1.398
LSTM	0.625	0.875	1.125	1.375
(De)Convolutional LSTM	0.625	0.875	1.125	1.375

Compressed images with LSTM architecture

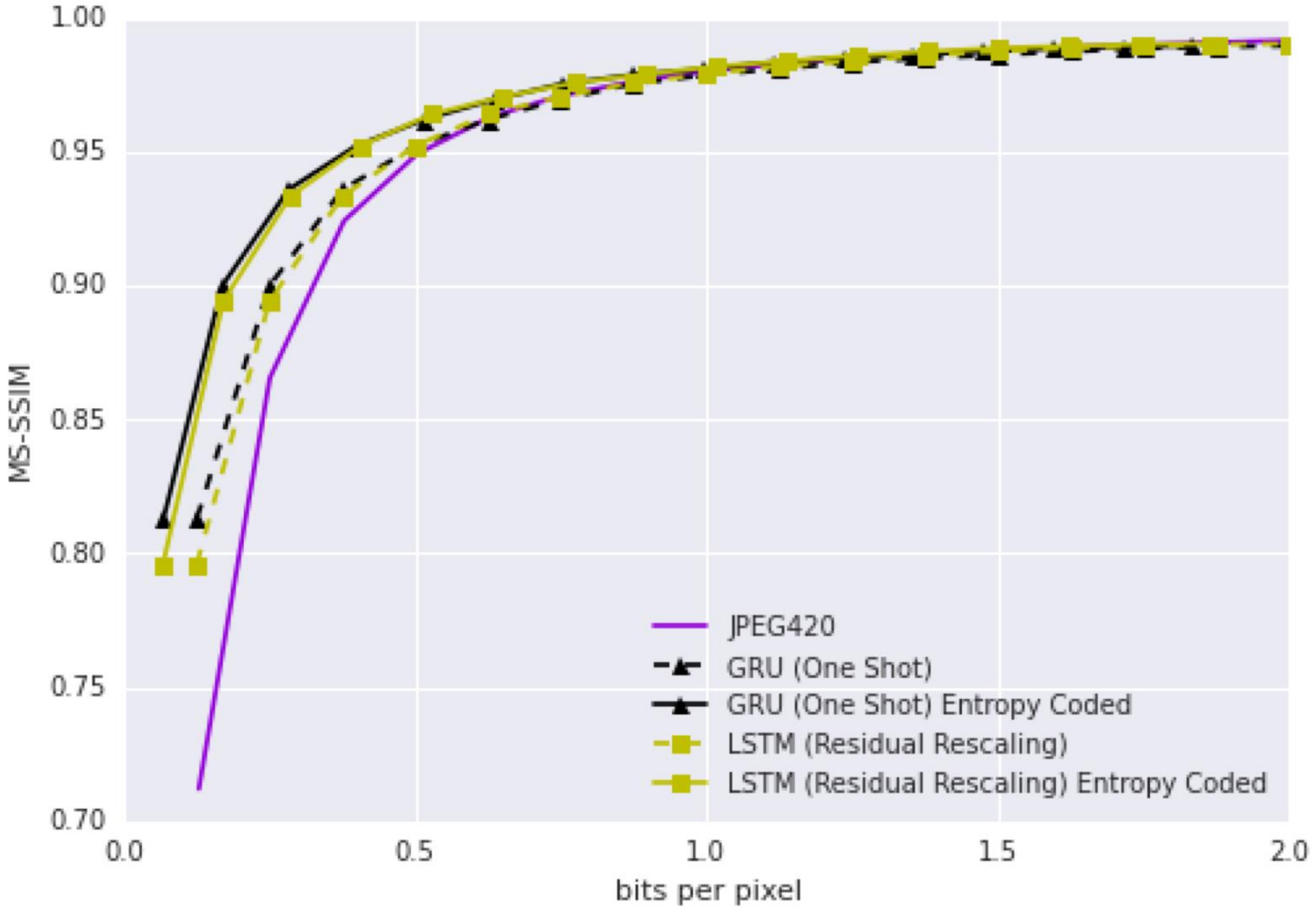
Compressed images with conv/deconv LSTM architecture

Extensions

- Application to full-resolution images (Todorici et al. (2017), Full-resolution image compression with RNN)
  - Long-term dependencies between patches
  - Binary recurrent network to predict symbol probabilities for an arithmetic encoder

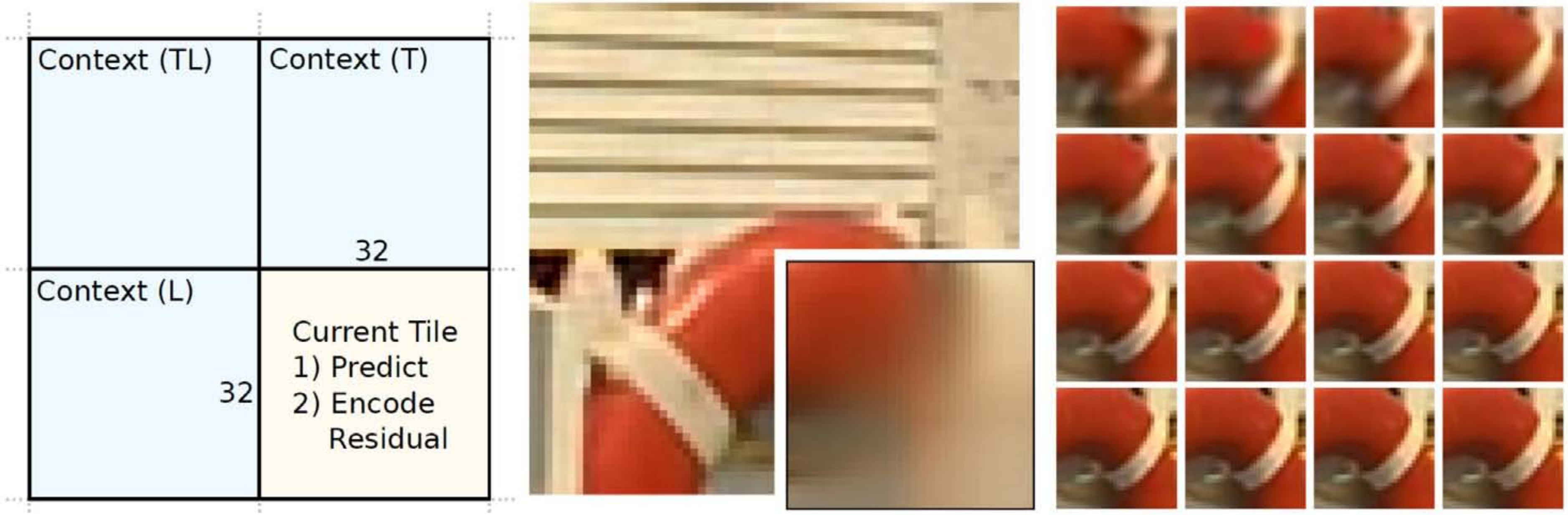


Extensions


1.00
0.95
0.90
0.85
0.80
0.75
0.70
MS-SSIM
0.0
0.5
1.0
1.5
2.0
bits per pixel
JPEG420
GRU (One Shot)
GRU (One Shot) Entropy Coded
LSTM (Residual Rescaling)
LSTM (Residual Rescaling) Entropy Coded

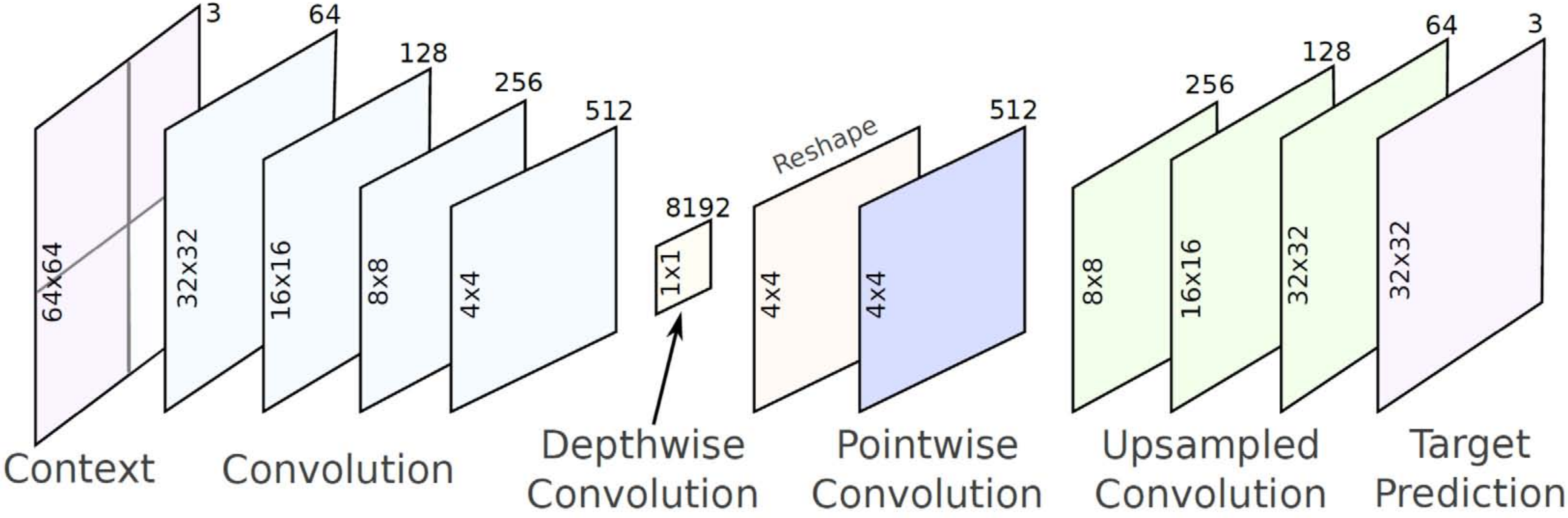
Extensions

- Spatially adaptive image compression using a tiled deep network (Minnen et al., ICIP (2017))
  - Spatial prediction (similar to inpainting of Pathak et al., CVPR 2016)
  - Residual coding



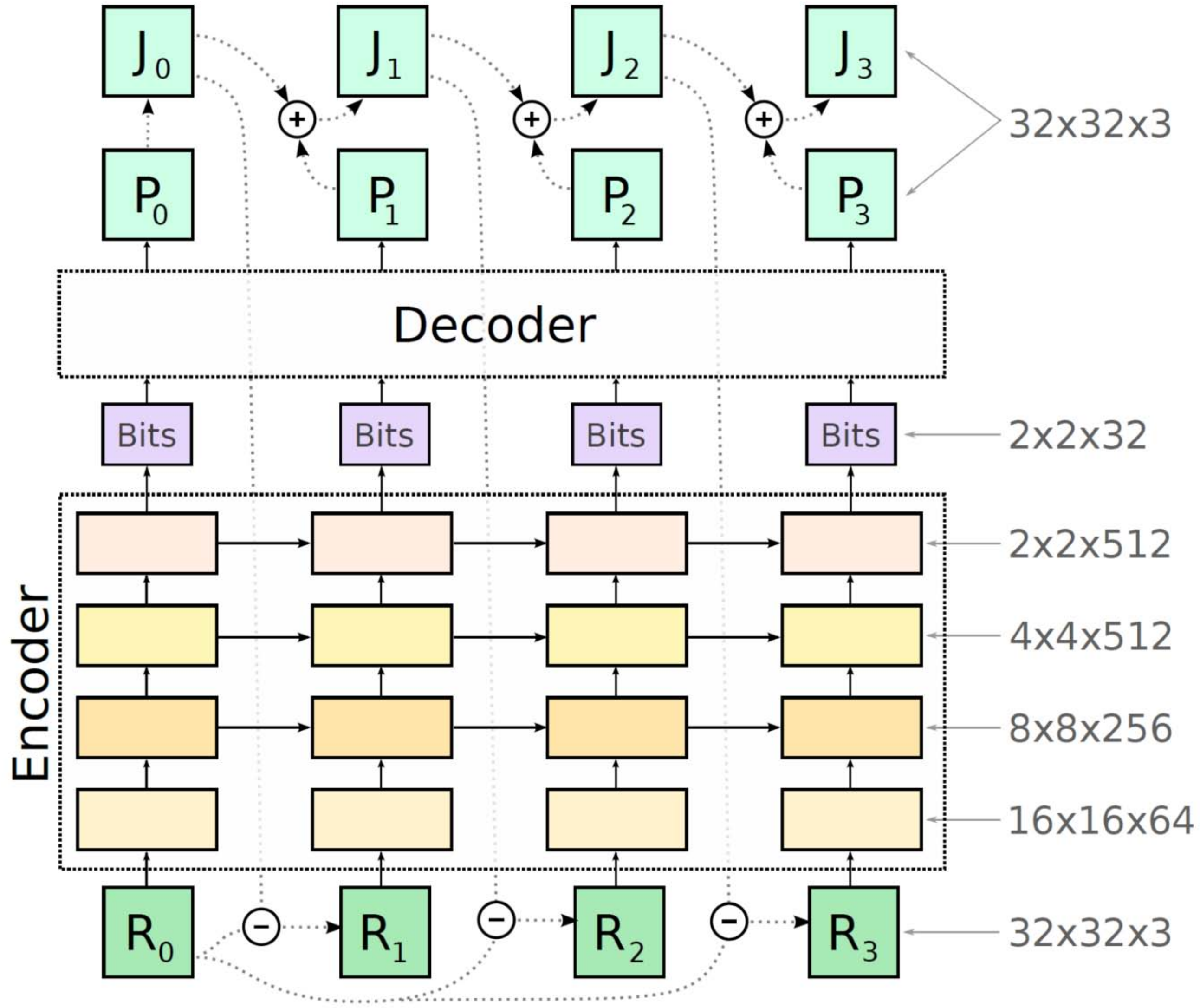
Spatial prediction

- Strided (downsampled) convolution/deconvolution architecture
- Similar to a denoising auto-encoder (used here for inpanting)



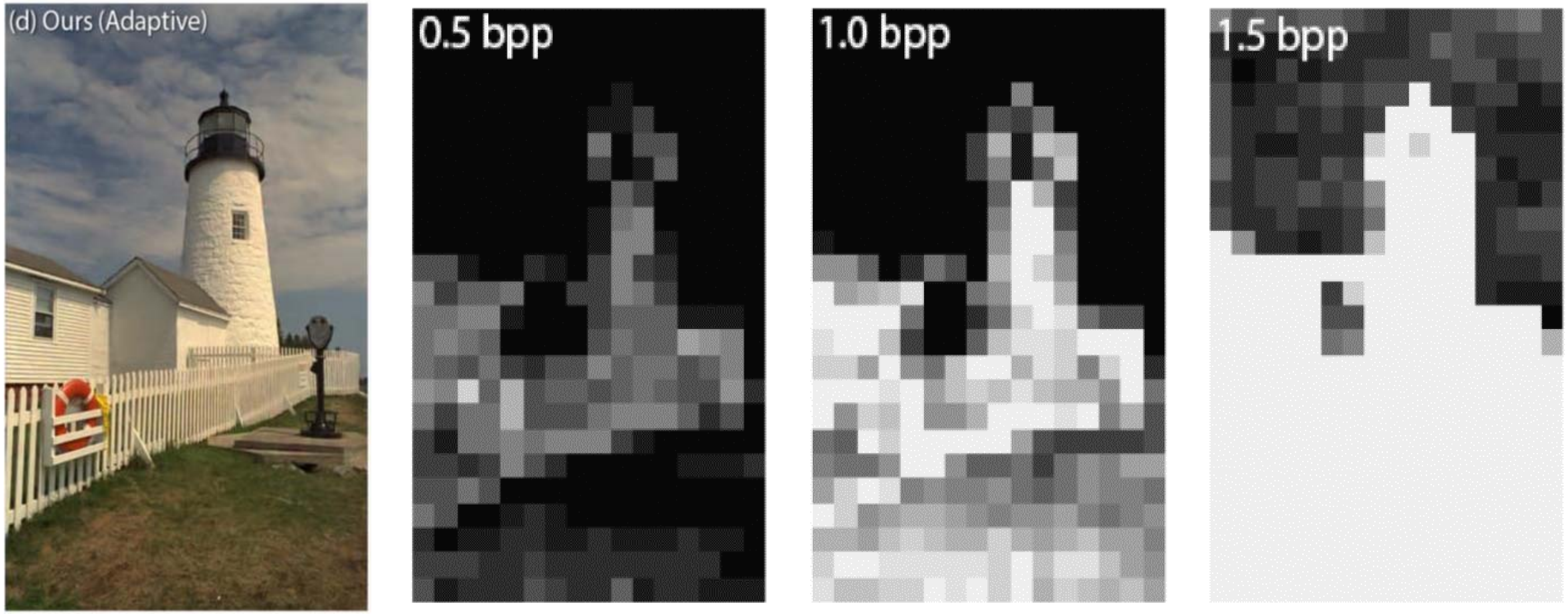
Coding of prediction residual

- Based on a recurrent auto-encoder as in Todorici et al. 2016
- Spatially adaptive bit allocation by stopping the iterations when a PSNR criterion for the tile is met

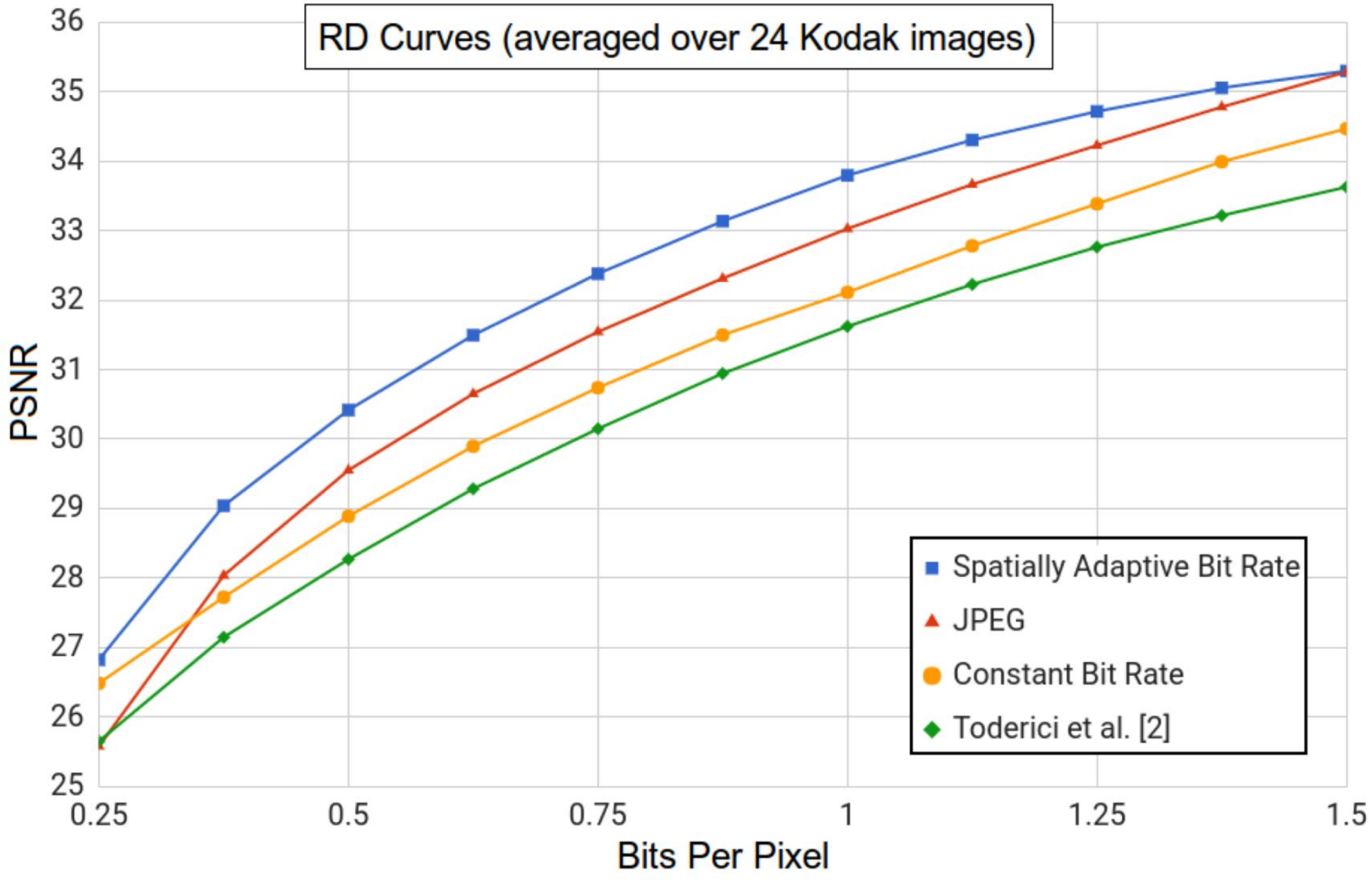


Spatial bitrate adaptation

- Allocation maps:



RD results



Results of subjective experiments are also reported:

Visual differences w.r.t. JPEG significant only for low bitrates

A revolution in image/video compression?

- CVPR 2018 challenge on learned image compression
  http://www.compression.cc/
- Results sometimes impressive
- But *very* dependent on the training/test conditions

- Conventional coding:
  - Based on simple signal models (generally used as an ensemble)
  - Innovation has been mainly incremental for the past 20+ years

- *Can DL revolutionize image/video compression?*