

AHG9: CNN-based driving of block partitioning for intra slices encoding

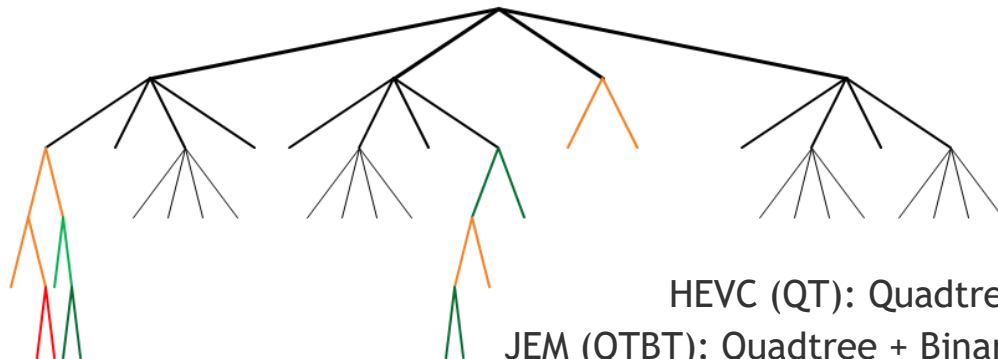
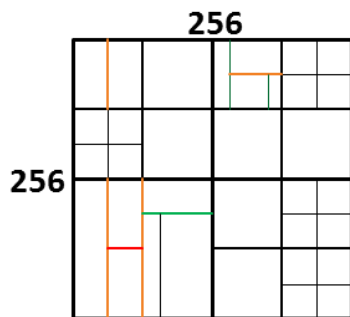
JVET-J0034



technicolor



Introduction



HEVC (QT): Quadtree Tree
JEM (QTBT): Quadtree + Binary Tree
Our (ABT): QTBT + Asymmetric Binary Tree

Tree structure	(almost) no heuristics
QT	8340
BT	186932 ($\times 22/\text{QT}$)
BT+ABT (J0022)	577689 ($\times 3/\text{QTBT}$)

Example of the combinatory variations when coding a single block of size 32×32

- Leverage the gains of QTBT and ABT, while keeping the combinatory reasonable
- Control RDO performance/complexity trade-off using a single parameter
- Perform CNN computation once, one pass encoding

Tree structure	(almost) no heuristics	With heuristics speed-ups	Using ou CNN-based speed-up
QT	8340	8181 (-2%)	n/a
BT	186932 ($\times 22/\text{QT}$)	58388 (-69%)	49769(-73%)
BT+ABT	577689 ($\times 3/\text{QTBT}$)	416281 (-28%)	182156(-68%)

Example of the combinatory variations when coding a single block of size 32×32

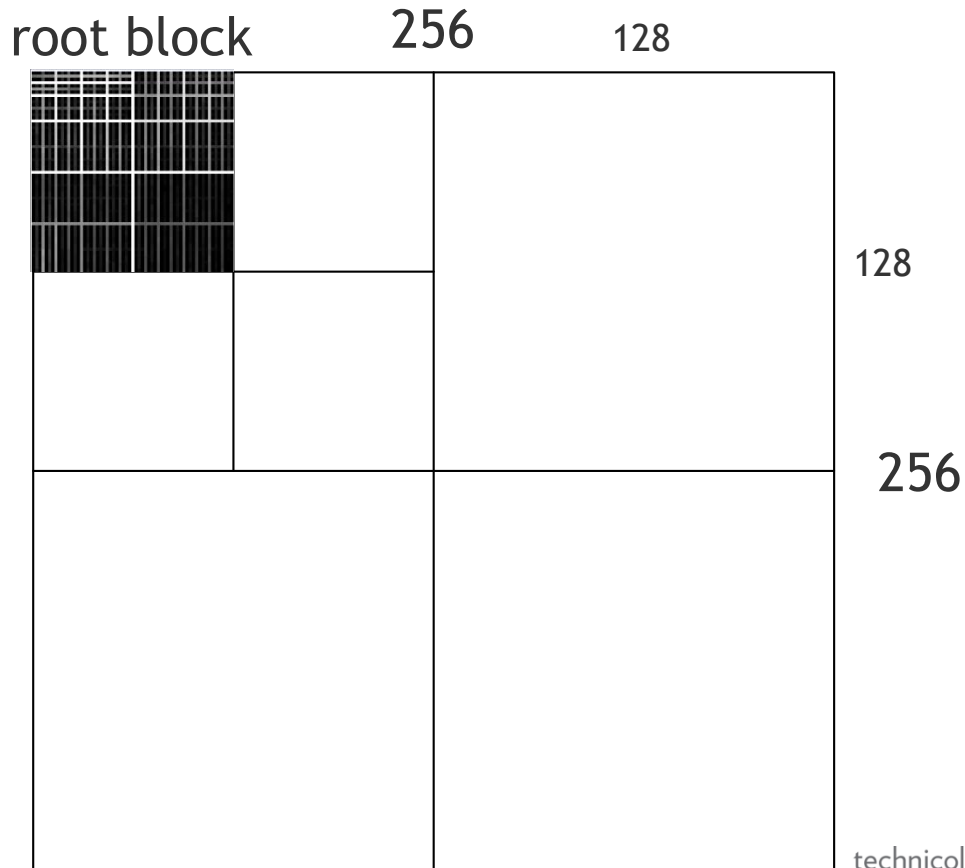
Proposed structure for intra slices

■ Luma:

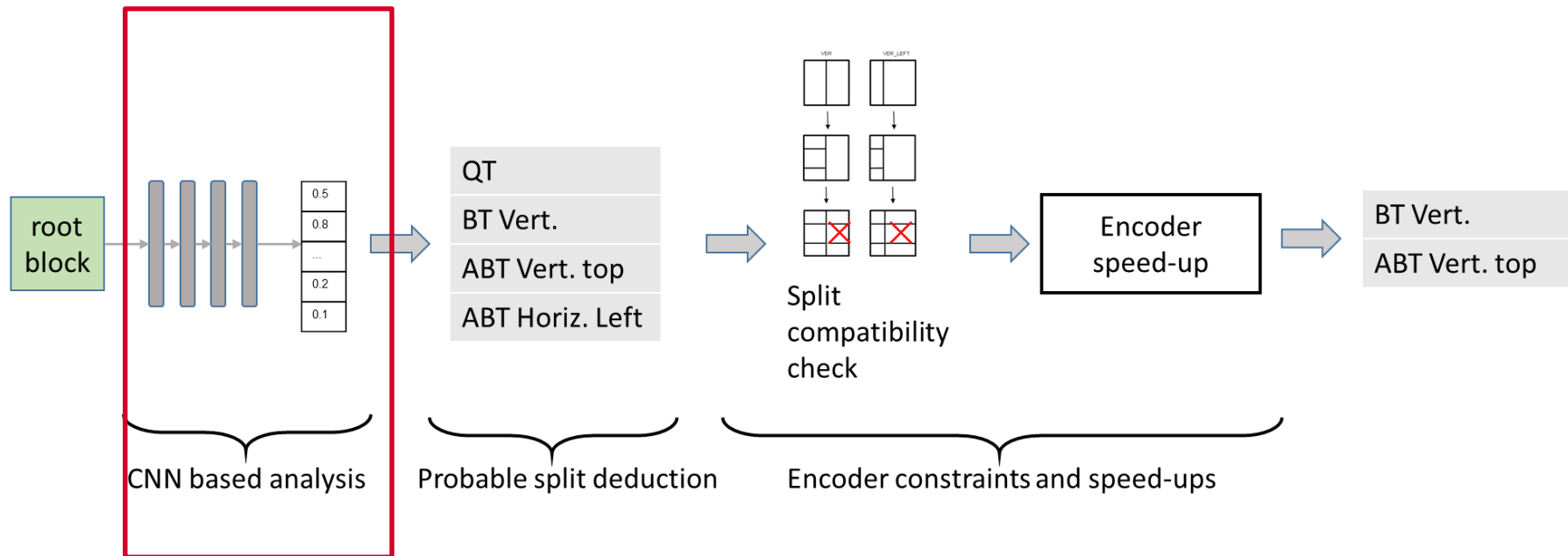
- 256×256 CTB
- 1st split inferred
- 2nd split: classical RDO
- 64×64: CNN-based decisions

■ Chroma in 4:2:0

- 128×128 CTB
- 1st split inferred
- 64×64: CNN-based decisions

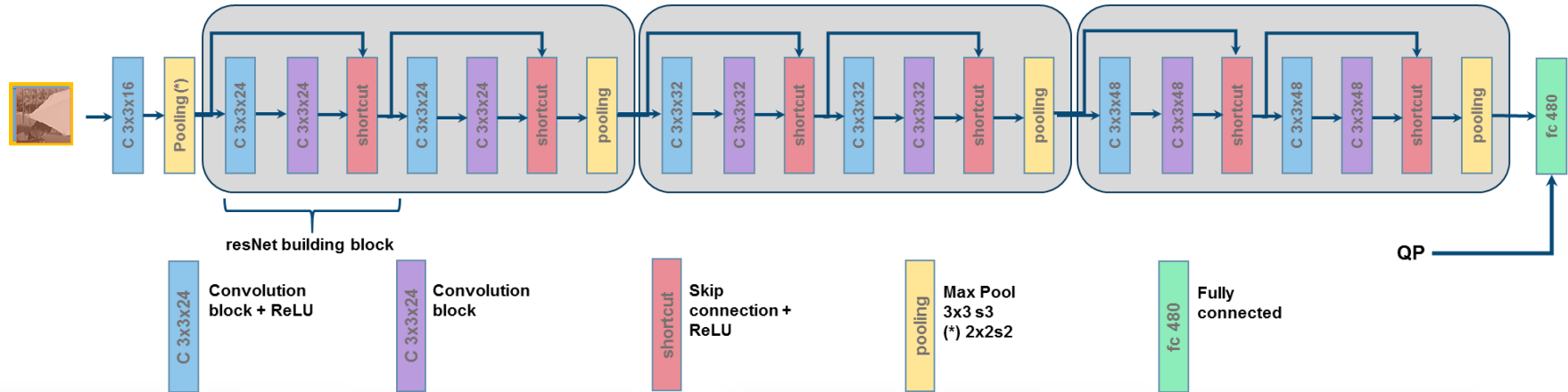


Overall framework



CNN-Model

- Structure derived from ResNet, ~250k parameters
- One CNN for Luma, one CNN shared for Cb and Cr, same structure
- Custom lightweight C++ implementation for Inference on single core CPU

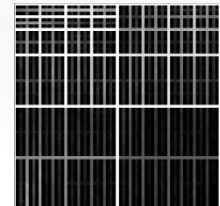


Inputs:

- 65×65 patches (64×64 block + LShape)
- QP

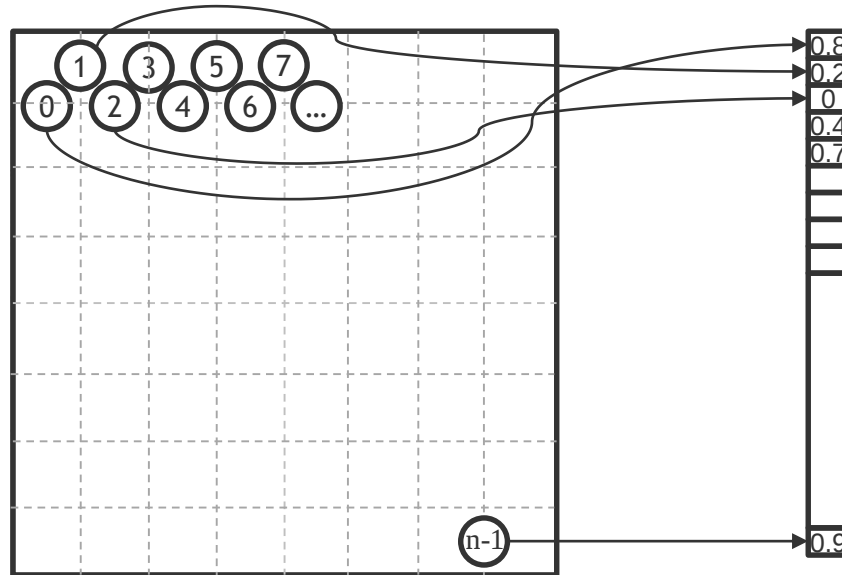
Output:

- boundary probabilities

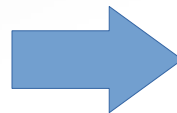
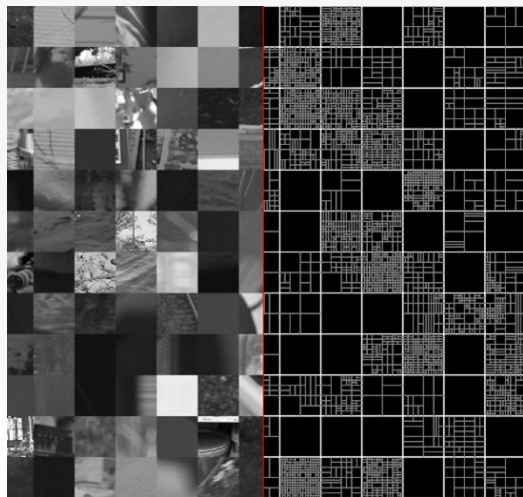


CNN output

- Output: 1 vector of probabilities for each elementary boundary
- Elementary block size: $4 \times 4 \rightarrow$ vector of size $n=480$



- Data set (training and validation)
 - encode a set of images in intra mode with the codec
 - large number of source images, excluding the JVET test sequences
 - 10M patches, encoded with QPs ranging from 19 to 41.
 - ground truth corresponds to boundary choices made by the RDO

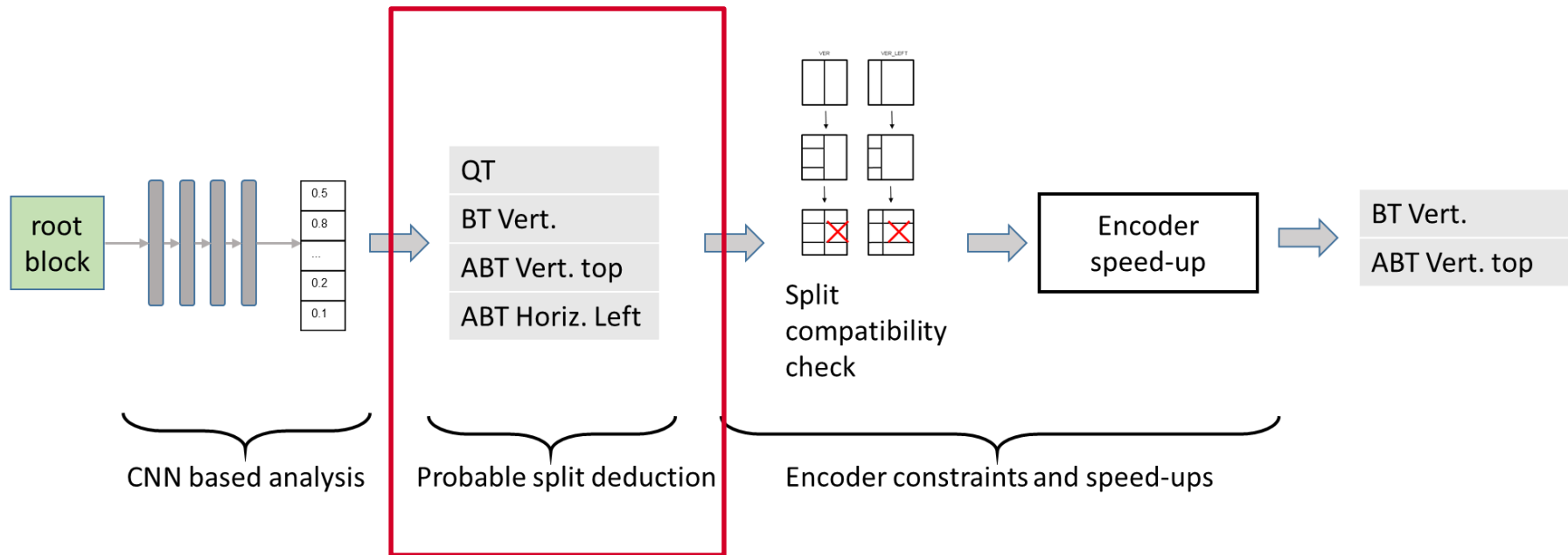


0
1
...
1
0

Ground truth V_{480}

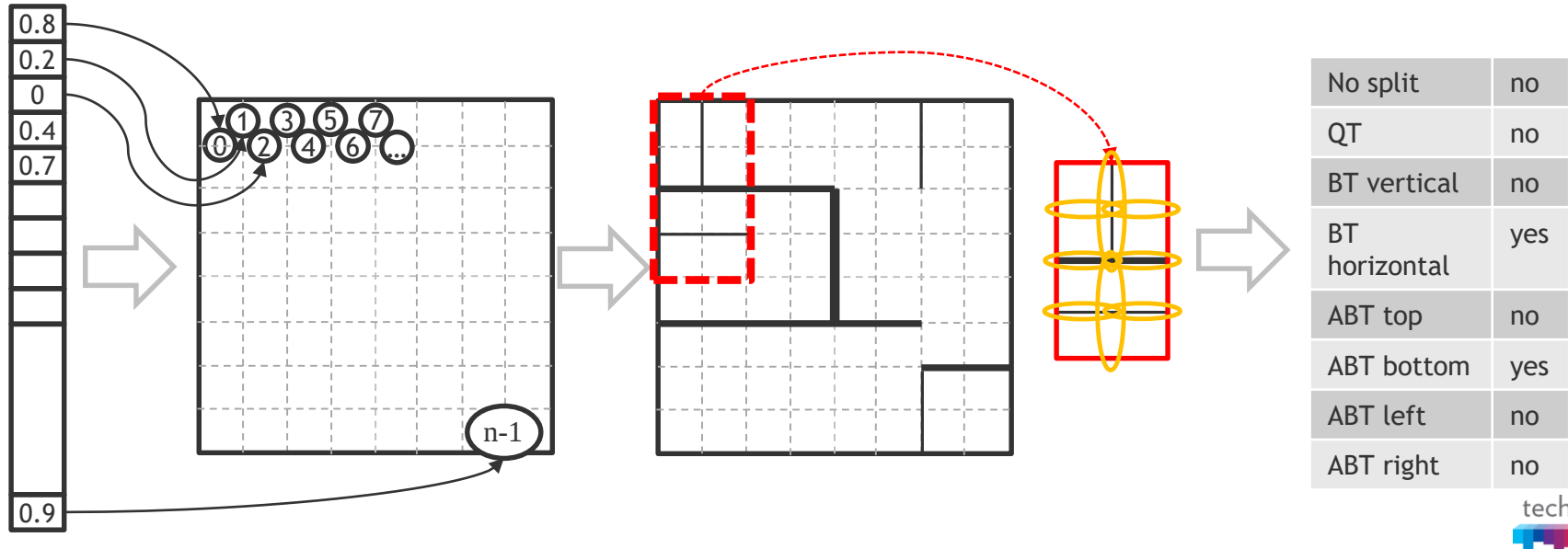
- Loss function: $\mathcal{L} = \sum_i \|\hat{V}_i - V_i\| + \lambda \sum_k \|c_k\|$
 - L_2 norm between the ground truth V and the output of the network $\hat{V}$
 - regularization term on the weights of the convolution
- The CNN is classically trained using the deep learning framework Tensorflow
 - Adam minimizer with
 - a learning rate of 10^{-3} ,
 - a regularization weight of 10^{-5}
 - a batch size of 256
- The model was trained on 4 epochs in a few hours on a single recent GPU.

Overall framework

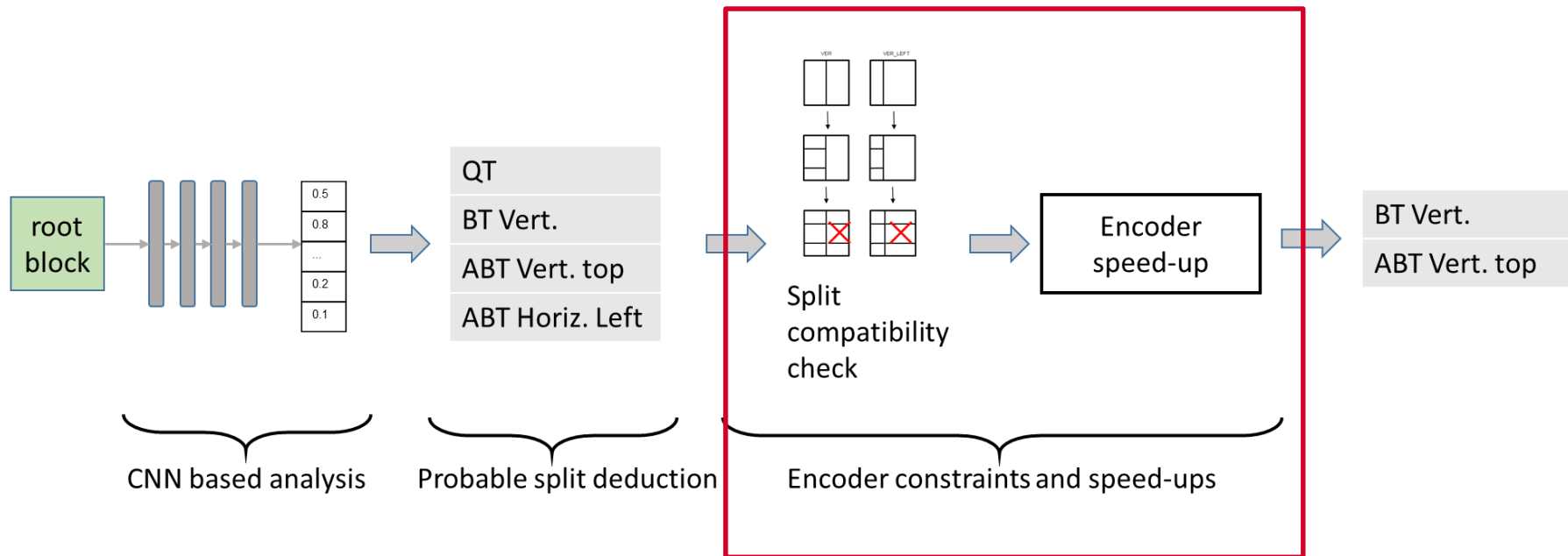


From boundaries to split selection

- Relevant boundaries are extracted
- Average value of elementary boundaries corresponding to each half-split is computed
- Comparison with pre-determined thresholds for each split



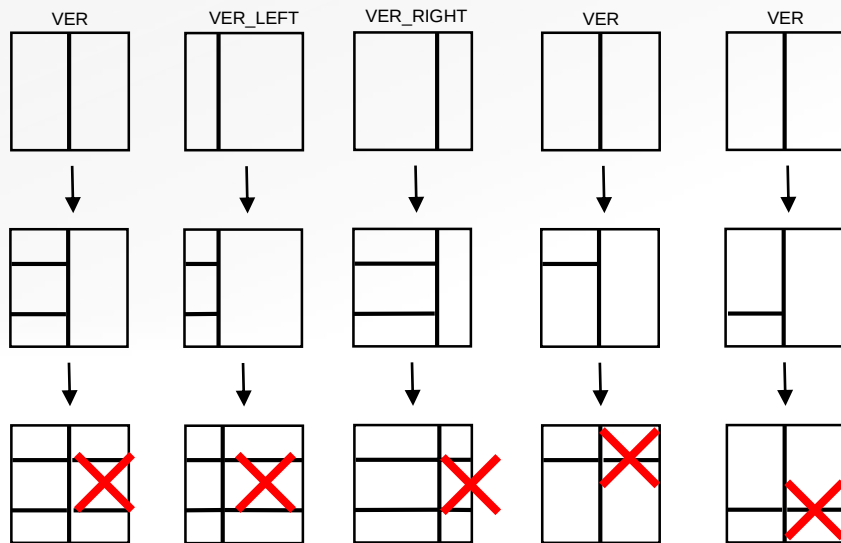
Overall framework



Encoder constrains and speed-ups

Some splitting modes may be removed to be consistent with encoder heuristics, when:

- not compatible with the current tree structure,
e.g. quadtree emulation with an horizontal and a vertical split,
- already explored in a previous configuration.



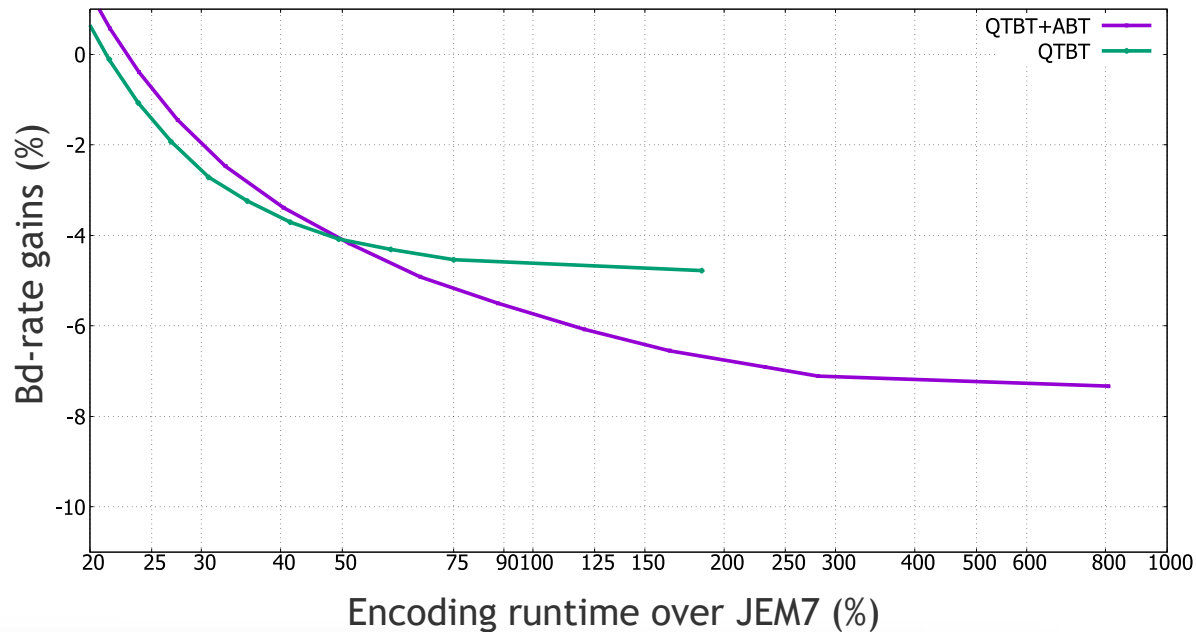
Structure modifications

Result using J0022

AI configuration

Structure of tree kept

Threshold changes only



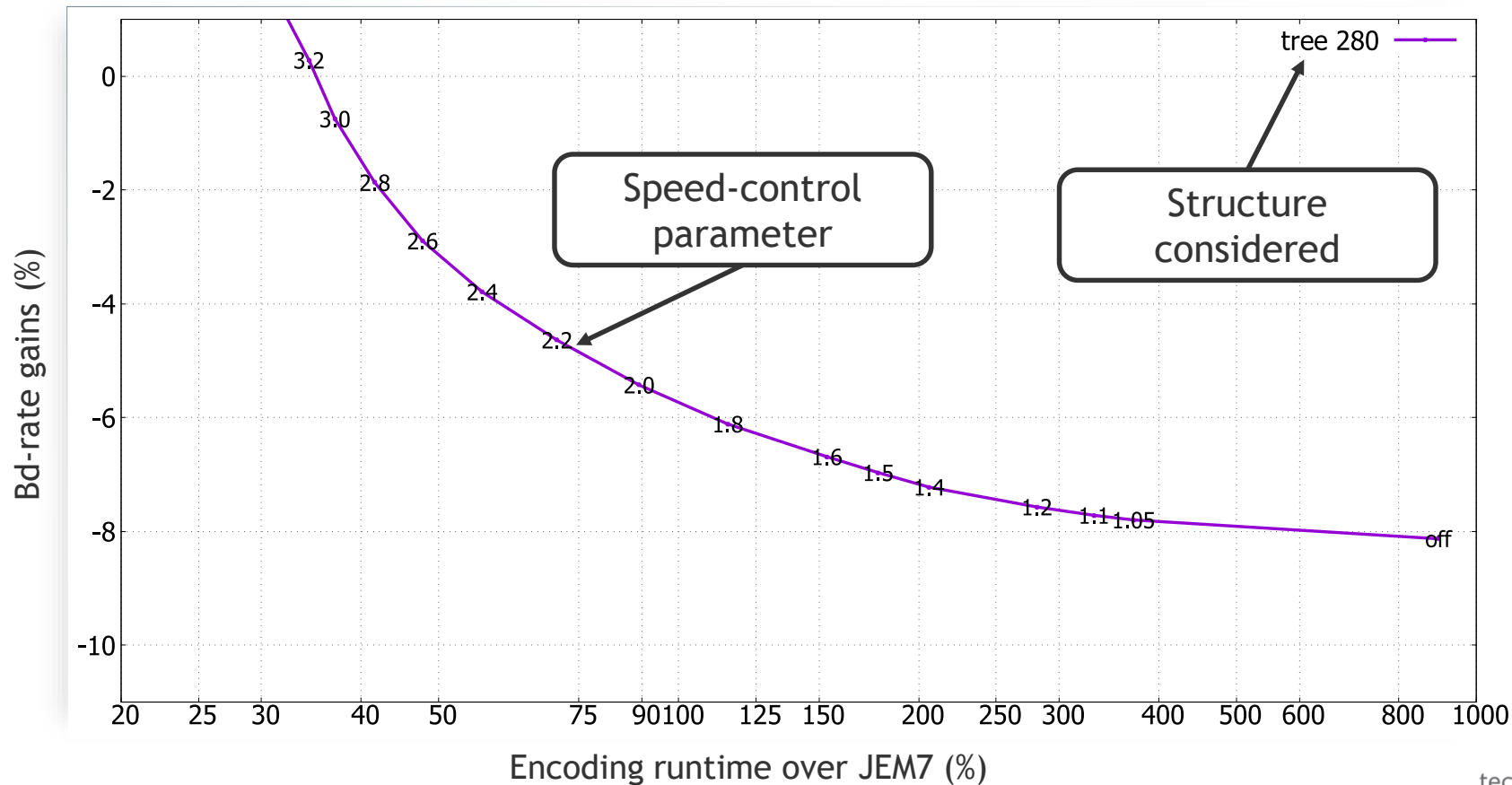
- ABT almost as effective at low complexity
- ABT outperforms QTBT from 50% of JEM7 encoding runtime
- ABT+CNN enable to get higher maximum gains

A coder control parameter “speed-control” jointly selects:

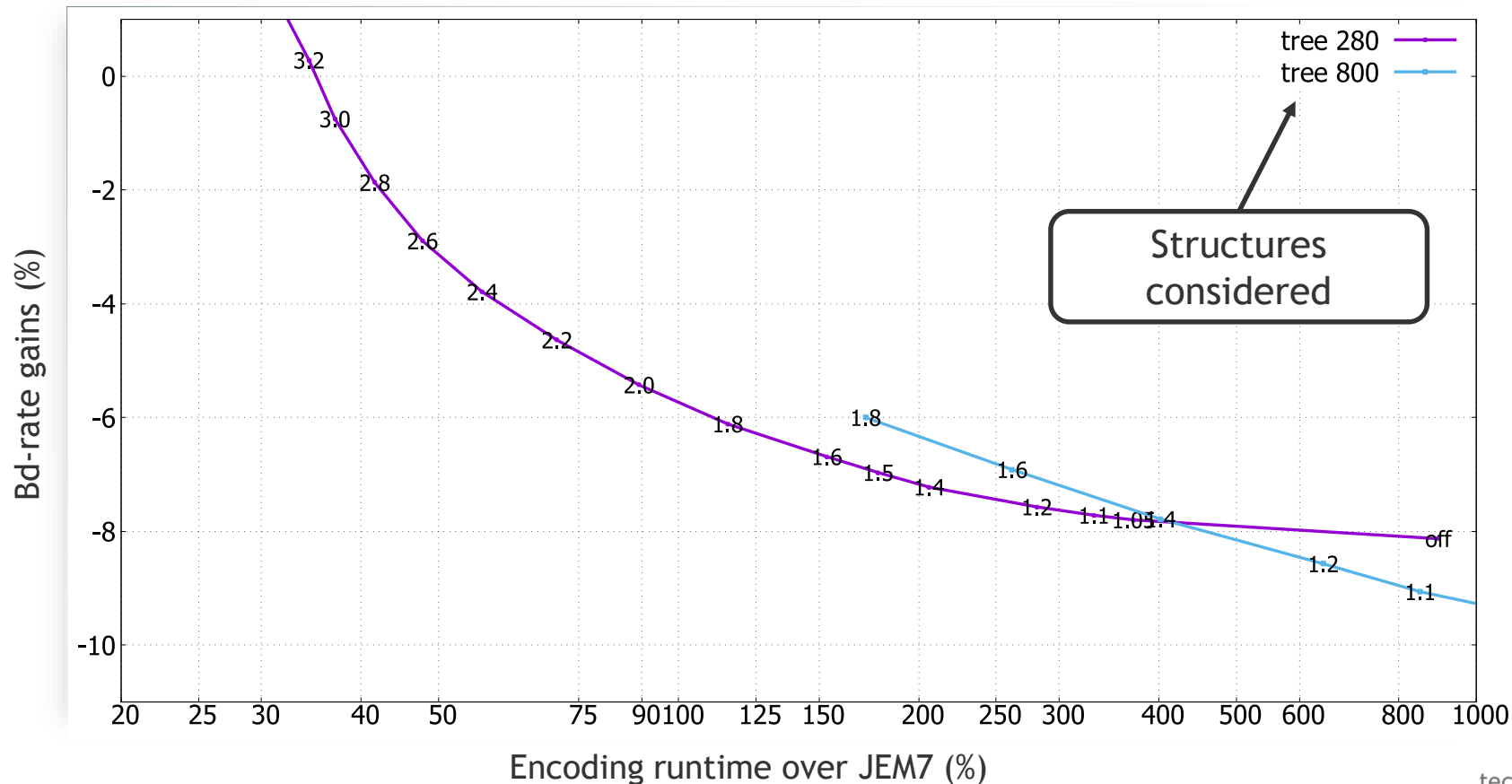
- the structure of the tree used:
 - maximum and minimum QT depths
 - maximum BT/ABT depths
- the probability thresholds, used to select split mode candidates.

Thresholds and structure can be different for the luma and chroma trees, enabling the encoder to make different trade-offs for each.

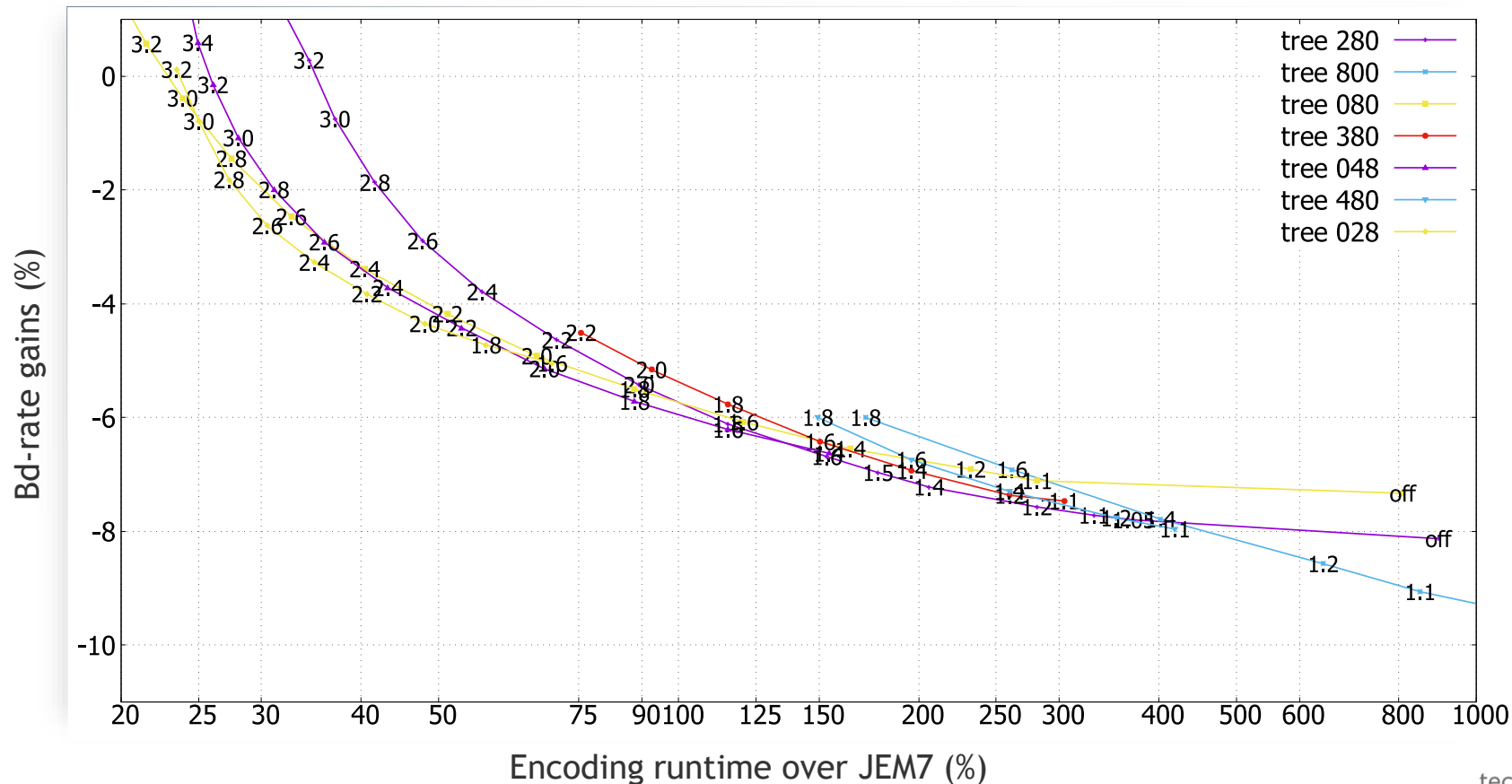
J0022 vs. JEM7, All Intra configuration



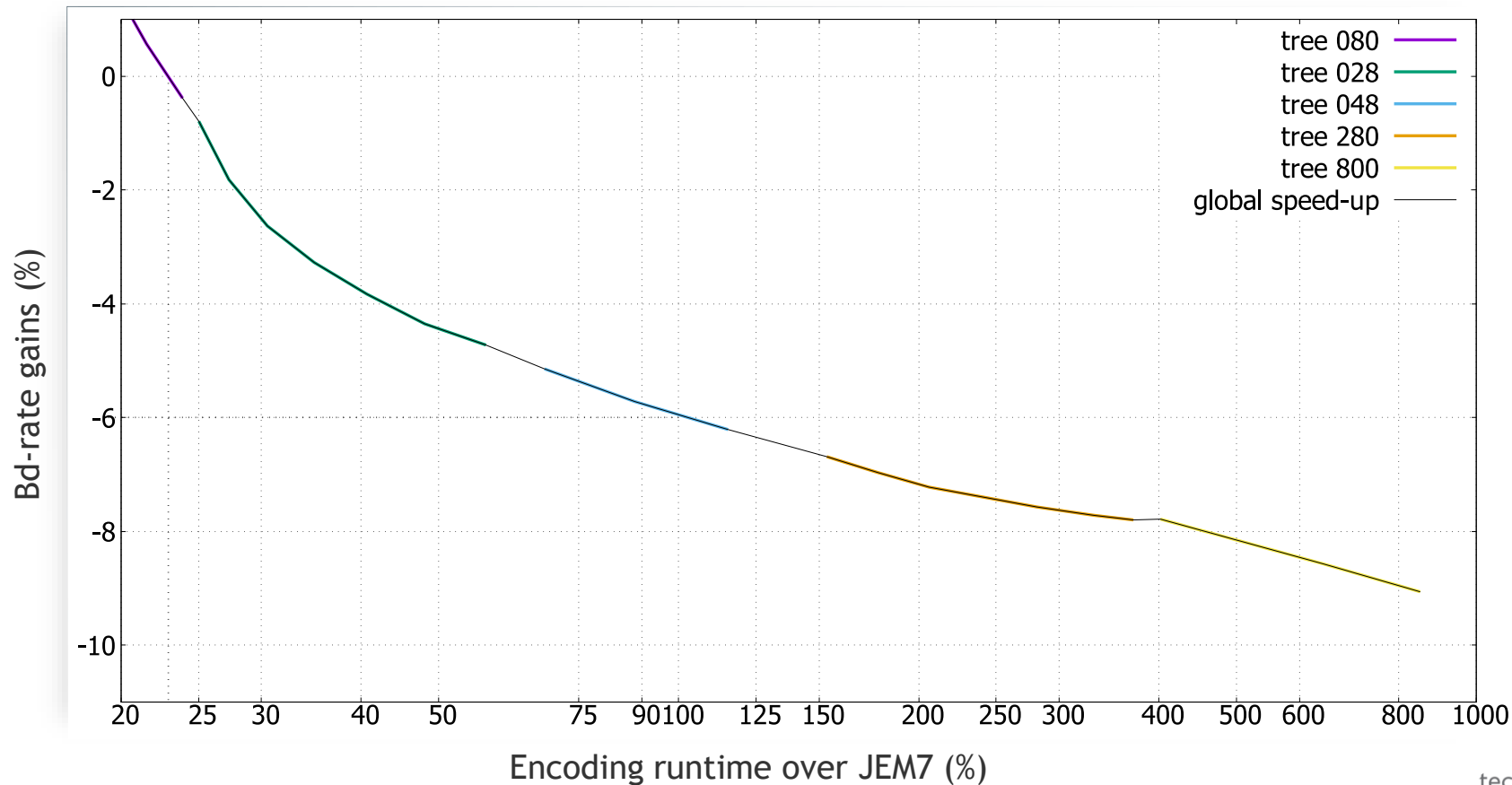
J0022 vs. JEM7, All Intra configuration



J0022 vs. JEM7, All Intra configuration

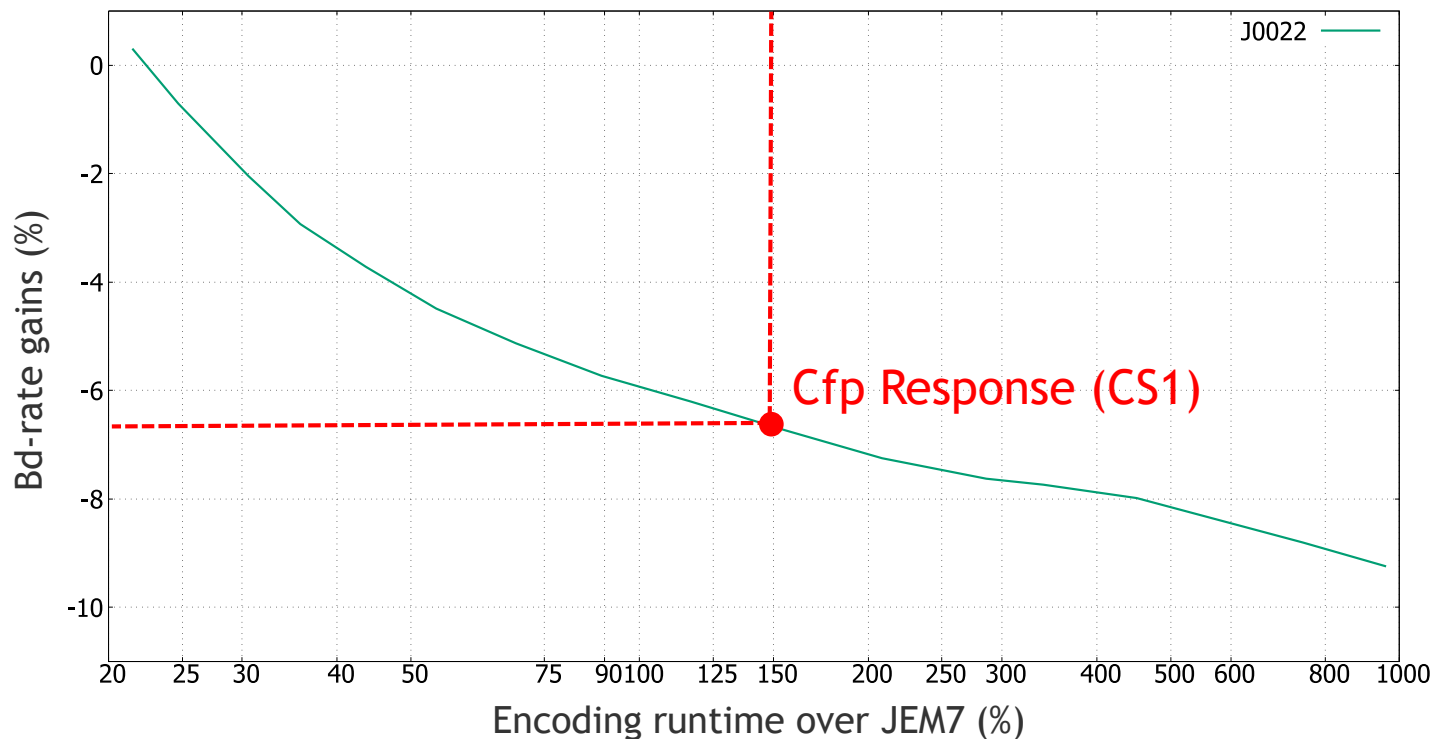


J0022 vs. JEM7, All Intra configuration



Results of J0022 in All Intra, compared to JEM7

- At same BD-rate as JEM7: encoder runtime sped up by a factor 4.3
- At same encoding runtime as JEM7: ~6% BD-rate gain



A deep-learning based partitioning speedup is proposed

It enables to leverage gains from complex partitioning schemes

Future encoders will include such tool

C++ source code is provided, included in the MTT software (J0022)