Deep Residual Learning in the JPEG Transform Domain

Max Ehrlich and Larry Davis

maxehr@umiacs.umd.edu lsd@umiacs.umd.edu

https://gitlab.com/Queuecumber/jpeg-domain-resnet

We leverage the JPEG format to achieve a substantial inference speedup over spatial domain networks

JPEG is a linear transform of image pixels, therefore it can be combined with other linear transforms like convolution. Batch normalization likewise has a simple formulation and for ReLu we use an approximation. By defining these operations we create a residual block that can operate on compressed JPEG data.

The JPEG Transform as Tensors

Given an image as a tensor of shape h, w, the JPEG transform can be defined as a linear map as follows

$$B^{hw}_{xymn} = \begin{cases} 1 & h, w \text{ belongs in block } x, y \text{ at offset } m, n \\ 0 & \text{otherwise} \end{cases}$$

$$D^{mn}_{\alpha\beta} = \frac{1}{4} V(\alpha) V(\beta) \cos\left(\frac{(2m+1)\alpha\pi}{16}\right) \cos\left(\frac{(2n+1)\beta\pi}{16}\right)$$

$$Z^{\alpha\beta}_{\gamma} = \begin{cases} 1 & \alpha, \beta \text{ is at } \gamma \text{ under zigzag ordering} \\ 0 & \text{otherwise} \end{cases}$$

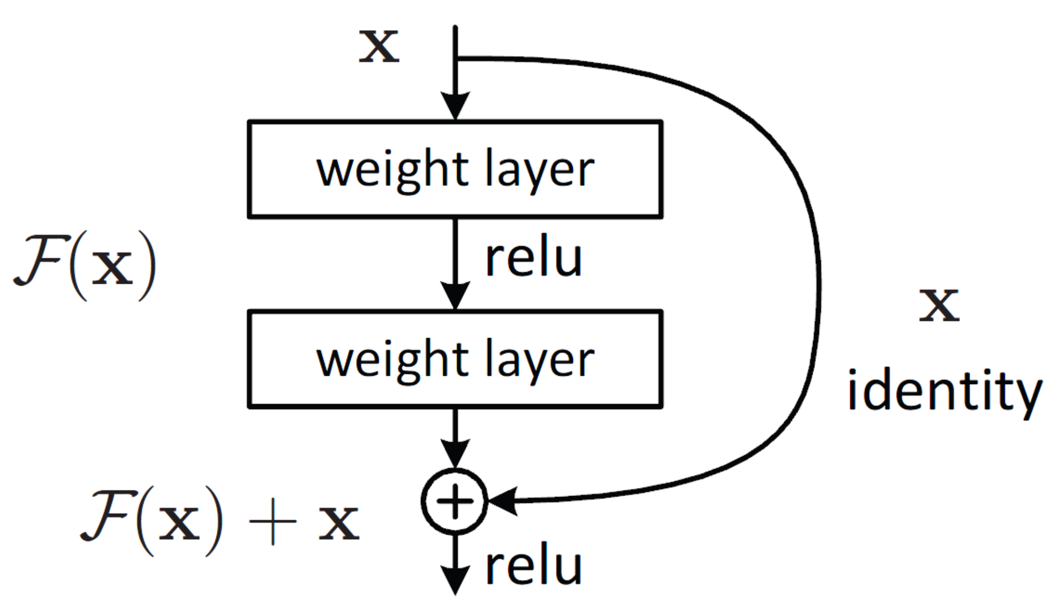
$$S^{\gamma}_{k} = \frac{1}{q_k}$$

$$\widetilde{S}^{k}_{\gamma} = q_k$$

Compression $$J^{hw}_{xyk} = B^{hw}_{xymn} D^{mn}_{\alpha\beta} Z^{\alpha\beta}_{\gamma} S^{\gamma}_{k}$$

Decompression $$\widetilde{J}^{xyk}_{hw} = B^{xymn}_{hw} D^{\alpha\beta}_{mn} Z^{\gamma}_{\alpha\beta} \widetilde{S}^{k}_{\gamma}$$

Residual Block



1. Convolution
2. ReLu
3. Convolution
4. Element-wise Sum
5. ReLu
6. Batch Norm
7. Global Avereage Pooling

By expressing these operations in the JPEG domain, we create a residual block that can process compressed JPEG data

Convolution

Convolution is a linear map on image pixels. For multichannel convolutions:

$$I'_{p'h'w'} = C^{phw}_{p'h'w'} I_{phw}$$

To apply this to JPEG data, include compression and decompression with the convolution

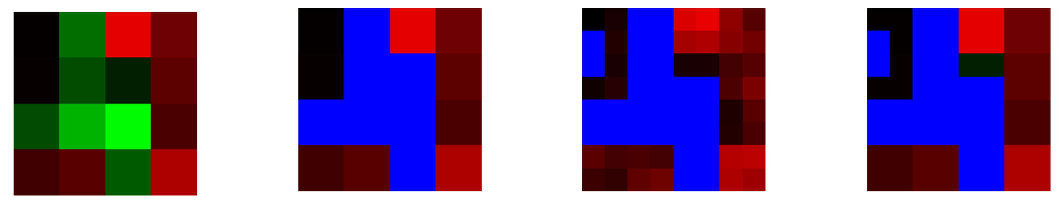
$$\Xi^{pxyk}_{px'y'k'} = \widetilde{J}^{xyk}_{hw} C^{phw}_{p'h'w'} J^{h'w'}_{x'y'k'}$$

then apply with

$$\Phi'_{x'y'k'} = \Xi^{xyk}_{x'y'k'} \Phi_{xyk}$$

ReLu

ReLu is nonlinear, we use a novel approximation technique by partially decompressing each block and masking negative pixels. The mask is then applied to the compressed block using a JPEG to spatial pixelwise multiplication. This preserves positive values which are not preserved by the naive approximation



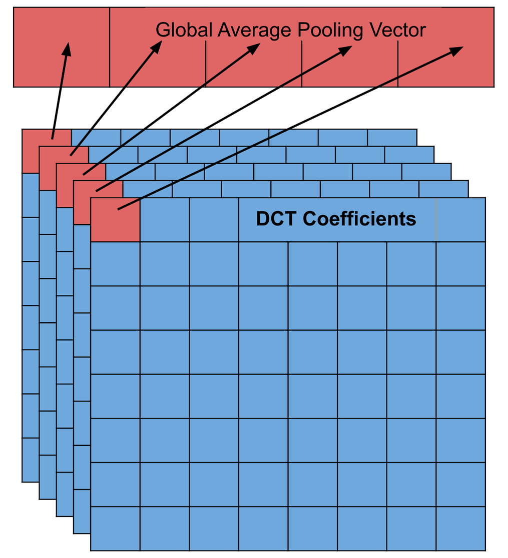
Batch Norm

Mean: The mean of a JPEG block is proportional to the 0th entry, so we can simply extract it for each block to get the per-channel means

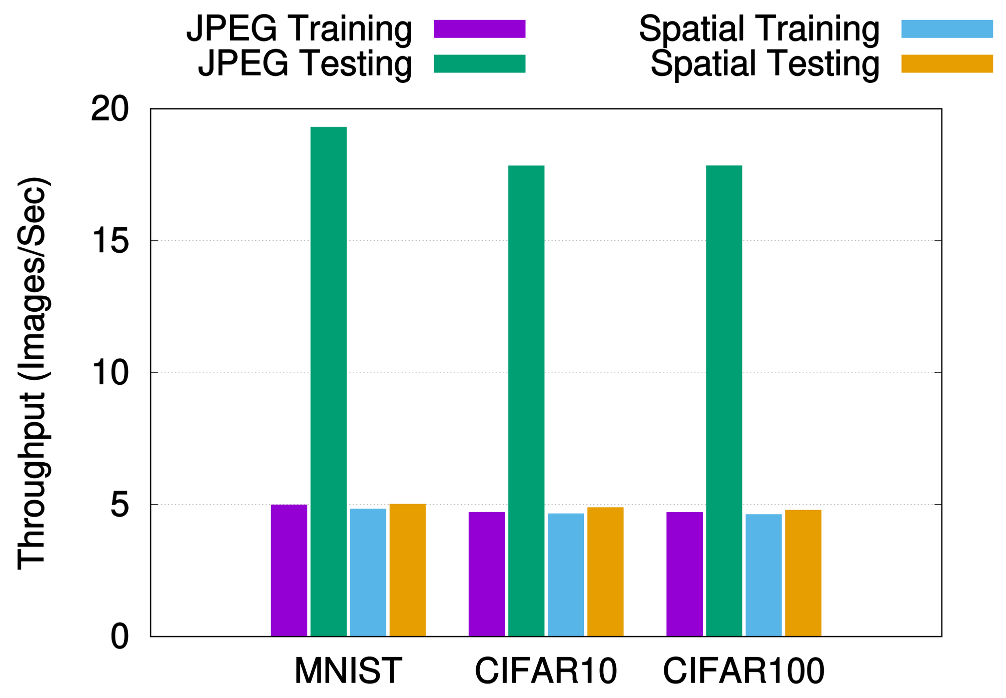
Variance: Variance is equal to the mean of DCT coefficients (proved in this work)

Global Average Pooling

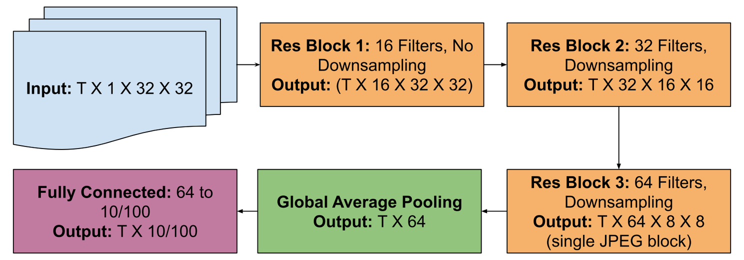
Extract 0th entry from each block as in batch norm



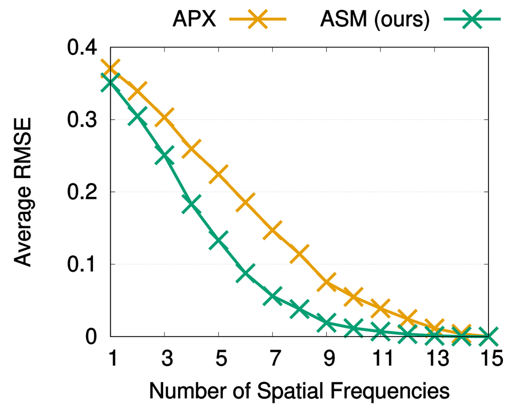
Results

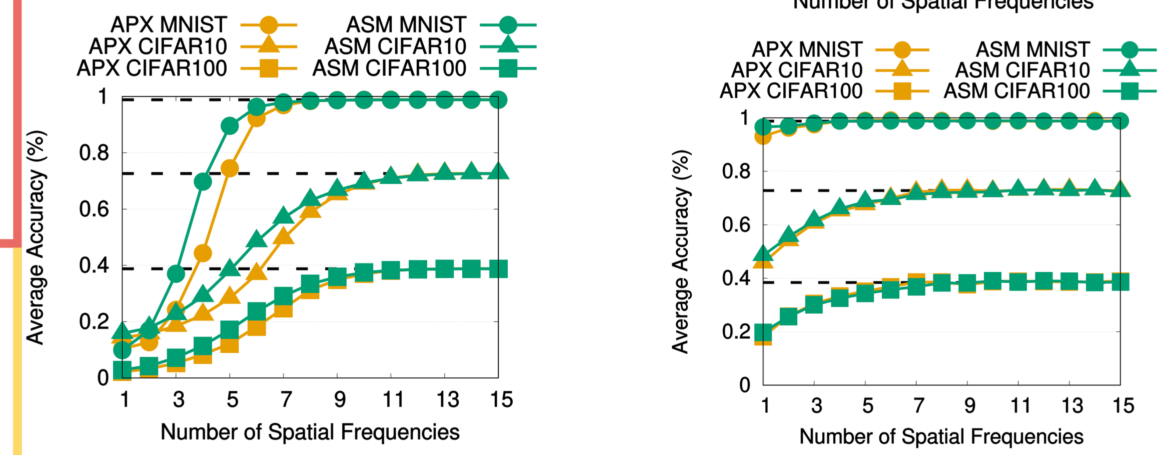


Network



ReLu Approximation Accuracy





Network accuracy converting spatial model (left) training JPEG model (right)

Acknowledgement

This research was partially funded by Facebook AI Research. We especially thank Dr. Ser-Nam Lim and his team at Facebook for their continued support of our work.