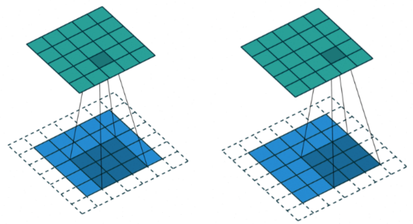
# Factorized Convolution Kernels in Image Processing

Alex Bergman, David Lindell
Stanford University

## Motivation

- Image denoising can be done with convolutional neural networks (CNNs) [3]
- Convolution kernel of size $D_K \times D_K \times S \times T$ operates on input $U$ of size $D_F \times D_F \times S$ and outputs $V$ of size $D_F \times D_F \times T$.

$$V(x,y,t) = \sum_{i=x-\delta}^{x+\delta} \sum_{j=y-\delta}^{y+\delta} \sum_{s=1}^{S} K(i-x+\delta, j-y+\delta, s, t)\, U(i,j,s)$$

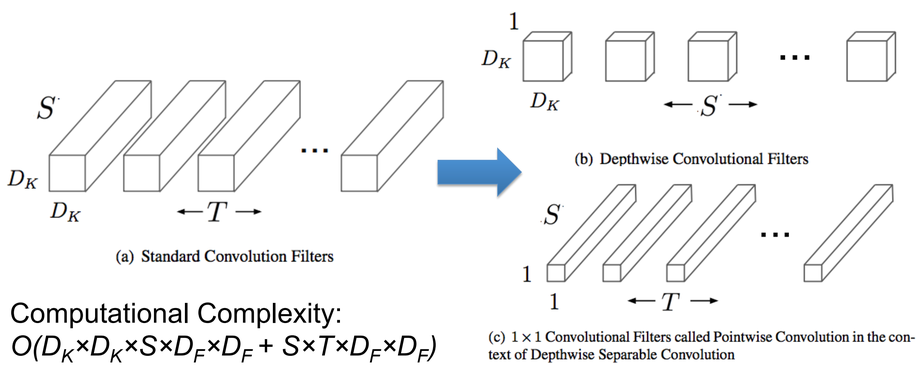


Where:
- $2\partial = D_K$
- x,y in $D_F \times D_F$

Computational Complexity:
$O(D_K \times D_K \times S \times T \times D_F \times D_F)$

## Related Work

- Approximate architecture with depthwise separable convolutions [1]:



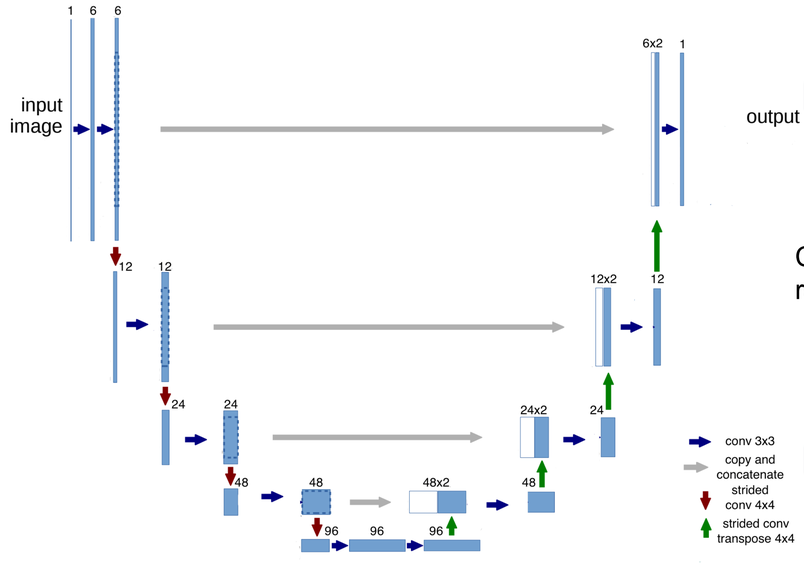

Computational Complexity:
$O(D_K \times D_K \times S \times D_F \times D_F + S \times T \times D_F \times D_F)$

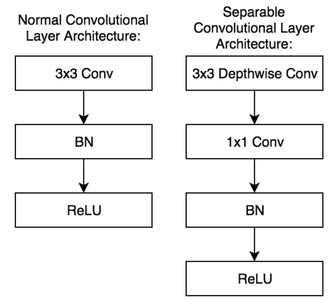
- Kernel factorization of pre-trained CNNs with CP-decomposition [2]

## References

[1] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." arXiv:1704.04861, 2017.
[2] V. Lebedev, Y. Ganin, M. Rakhuba, I. Oseledets, and V. Lempitsky. "Speeding-up convolutional neural networks using fine-tuned cp-decomposition". arXiv:1412.6553, 2014.
[3] Burger, H.C., Schuler, C.J., Harmeling, S., "Image denoising: Can plain neural networks compete with BM3D?", CVPR 2012, pp. 2392—2399

## New Technique



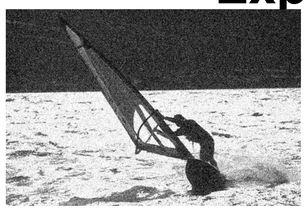

Change architecture to reflect separable convolutions, retrain on data vs. decompose an already trained model

Algorithm 1 CP-decomposition convolutional layer

Require: Rank: $r$, $conv2D$ layer to factor: $layer$
1: $pw_{in}, pw_{out}, dw_{vertical}, dw_{horizontal} = CP(layer.weight, rank = r)$
2: Create $conv2D$ layers with kernel sizes as follows:
$L1 = conv2D(size(pw_{in}))$, $L1.weight = pw_{in}$
$L2 = conv2D(size(dw_{vertical}))$, $L2.weight = dw_{vertical}$
$L3 = conv2D(size(dw_{horizontal}))$, $L3.weight = dw_{horizontal}$
$L4 = conv2D(size(pw_{out}))$, $L4.weight = pw_{out}$
3: $layer_{CP} = sequential([L1, L2, L3, L4])$

## Experimental Results

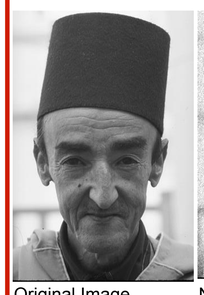

Original Image

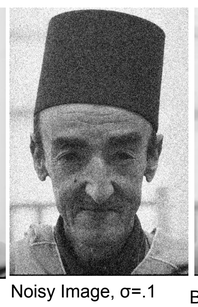

Noisy Image, σ=.1


BM3D denoised image
PSNR = 26.3638

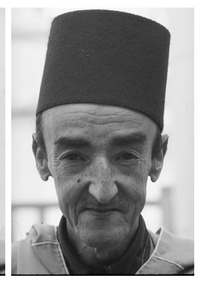

UNet denoised image
PSNR: 26.5669

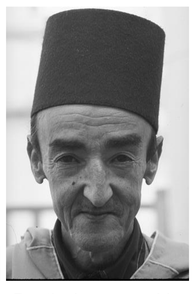

Depthwise separable conv UNet
PSNR: 25.8227


Original Image

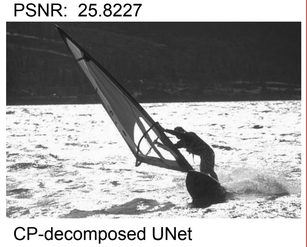

Noisy Image, σ=.1


BM3D denoised image
PSNR = 32.6021


UNet denoised image
PSNR: 31.7033


Depthwise separable conv UNet
PSNR: 29.6242


CP-decomposed UNet
PSNR: 28.1682


CP-decomposed UNet
PSNR: 25.6650

UNet: Operations: $178.487045 \times 10^9$, Parameters: 811100
Separable: Operations: $26.765184 \times 10^9$, Parameters: 76968
CP-decomposed: Operations: $127.800331 \times 10^9$, Parameters: 351651