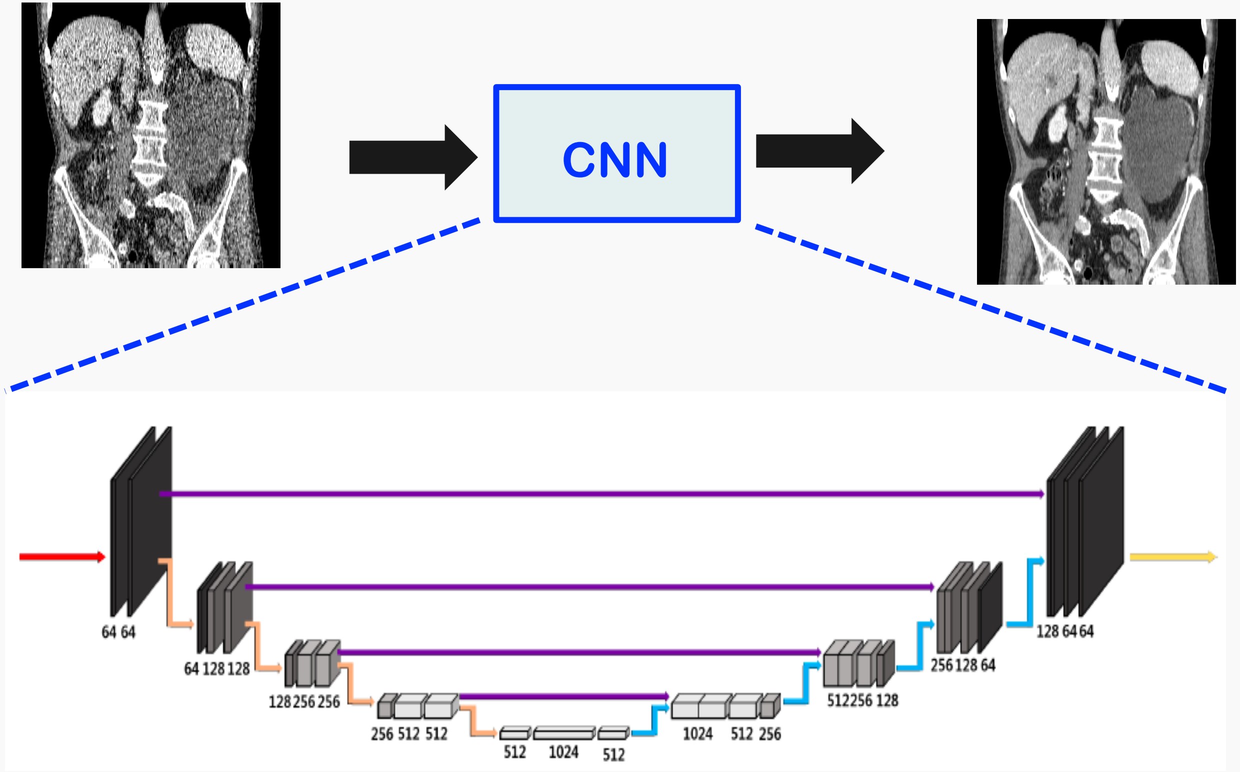
Understanding Geometry of Encoder-Decoder CNNs

(E-D CNNs)

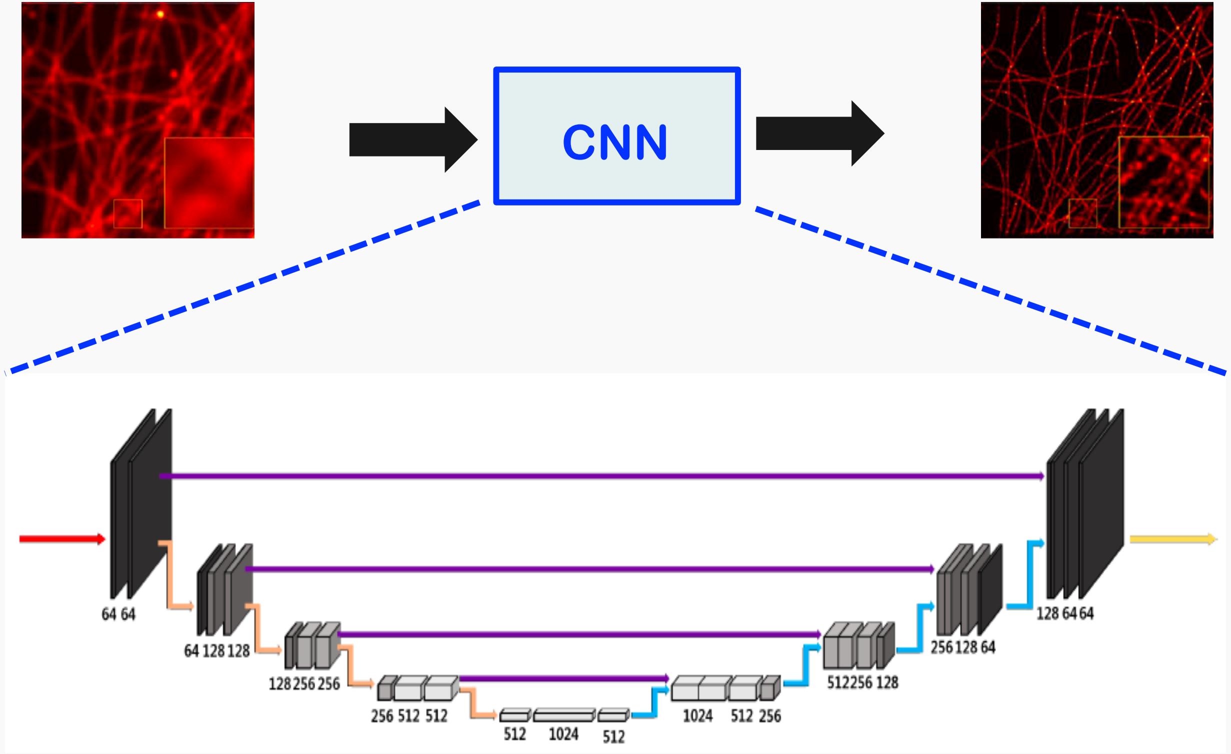
Jong Chul Ye
&
Woon Kyoung Sung

BISPL - BioImaging, Signal Processing and Learning Lab.
Dept. Bio & Brain Engineering
Dept. of Mathematical Sciences
KAIST, Korea

E-D CNN for Inverse Problems

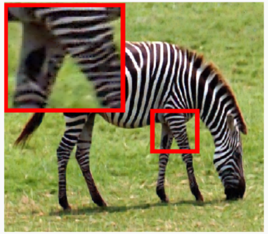


E-D CNN for Inverse Problems



E-D CNN for Inverse Problems

CNN



Successful applications to various inverse problems

Why Same Architecture Works for Different Inverse Problems ?

Classical Methods for Inverse Problems

Step 1: Signal Representation

$$x = \sum_i \langle b_i, x \rangle \tilde{b}_i$$

coefficients

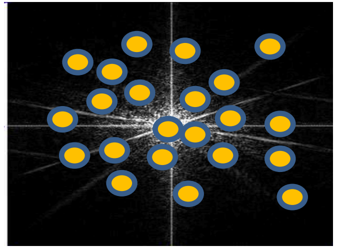
Analysis basis

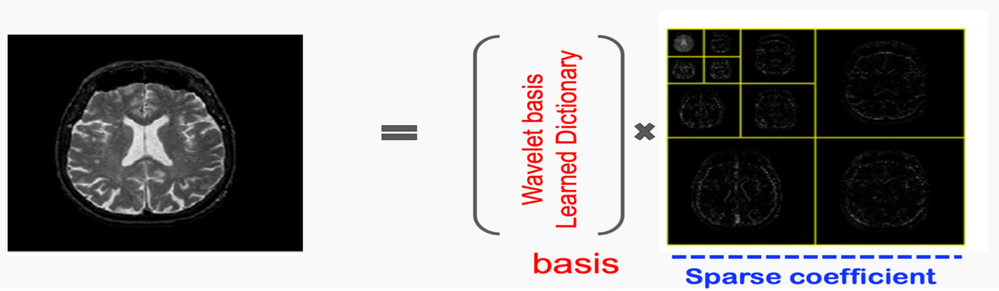
Synthesis basis

Classical Methods for Inverse Problems

Step 2:   Basis Search by Optimization

Eg. Compressed Sensing



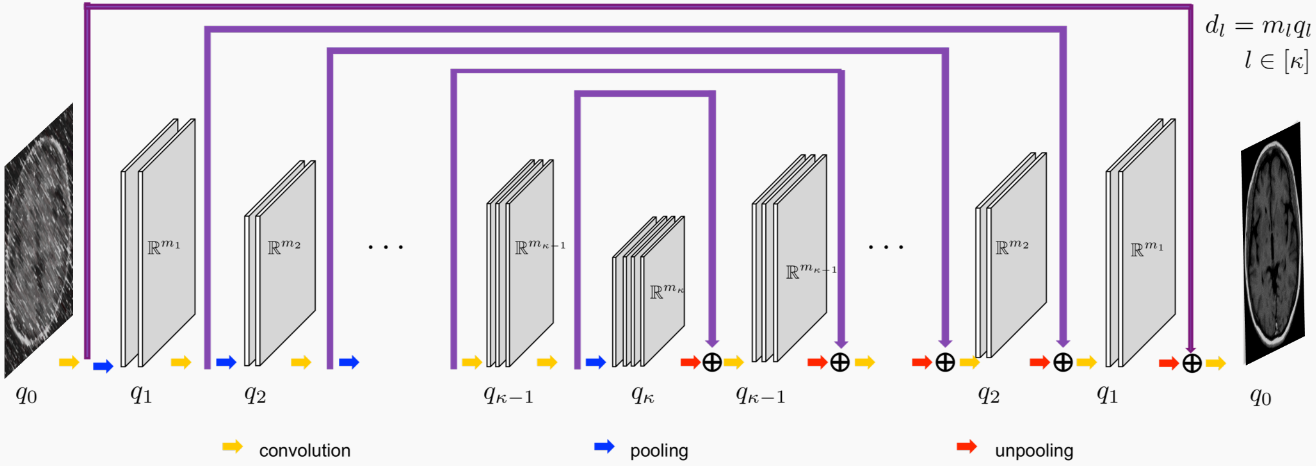


$$x = \sum_i \tilde{b}_i \langle b_i , x \rangle$$

Why do They Look so Different ?

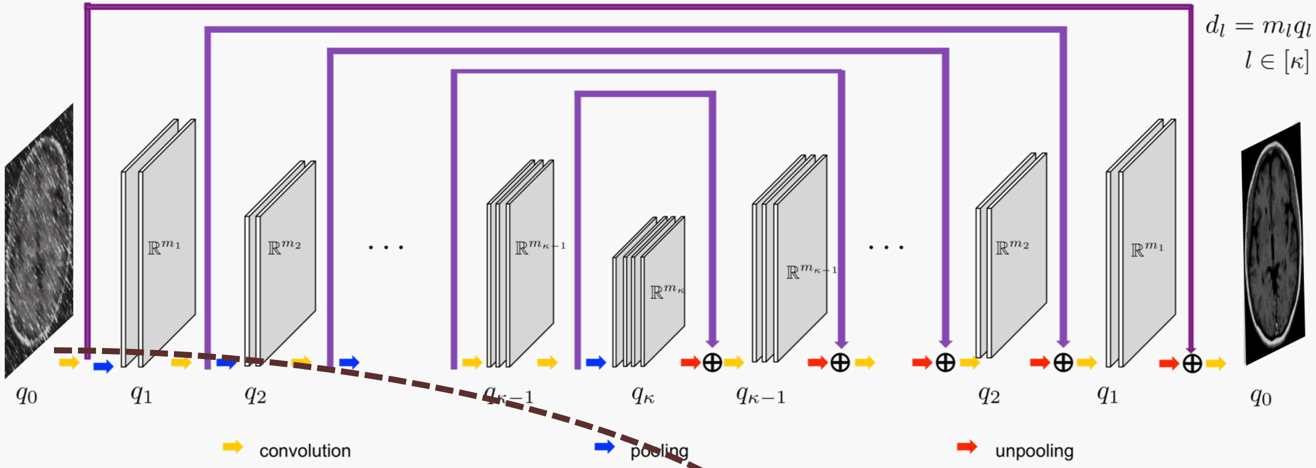
Any Link between Them ?

Our Theoretical Findings



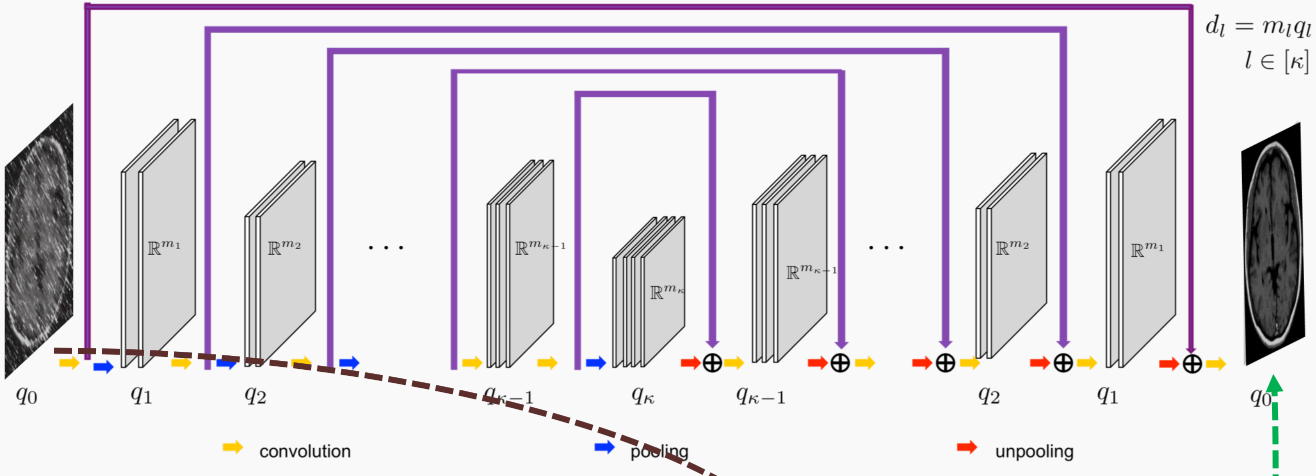
$$y = \sum_i \langle b_i(x), x \rangle \tilde{b}_i(x)$$

Our Theoretical Findings



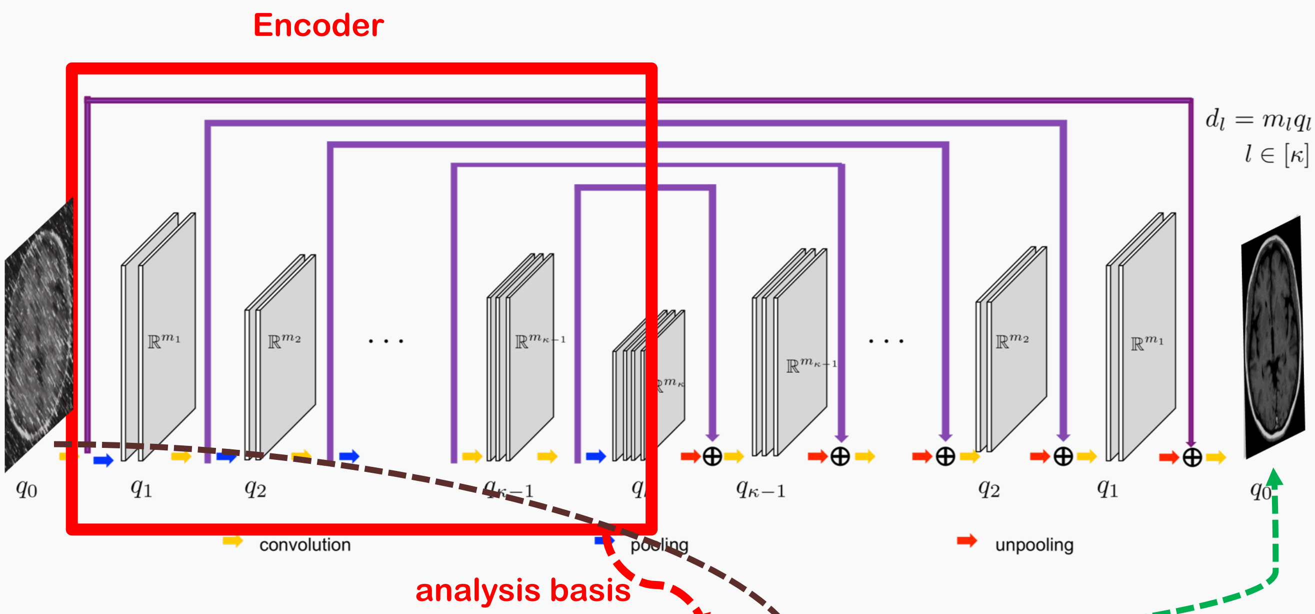
$$y = \sum_i \langle b_i(x), x \rangle \tilde{b}_i(x)$$

Our Theoretical Findings



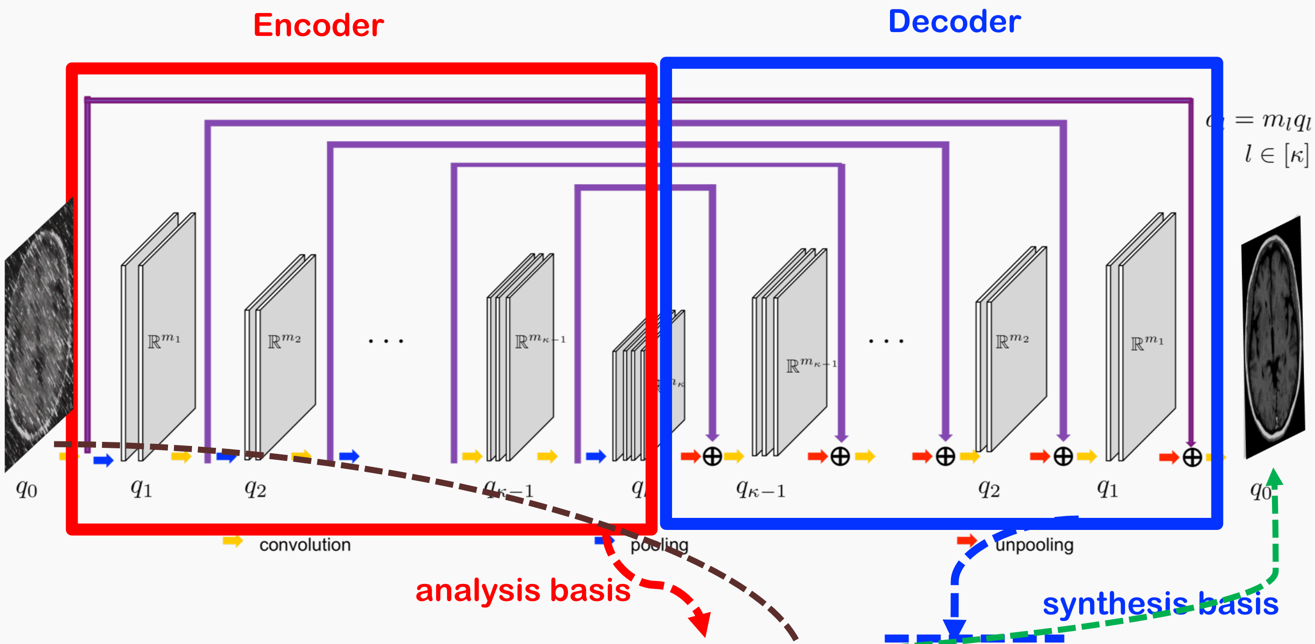
$$y = \sum_i \langle b_i(x), x \rangle \tilde{b}_i(x)$$

Our Theoretical Findings



$$y = \sum_i \langle b_i(x), x \rangle \tilde{b}_i(x)$$

Our Theoretical Findings



$$y = \sum_i \langle b_i(x), x \rangle \tilde{b}_i(x)$$

Linear E-D CNN

$$y = \tilde{B}B^\top x = \sum_i \langle x, b_i \rangle \tilde{b}_i$$

$$\begin{aligned} B &= E^1 E^2 \cdots E^\kappa, \\ \tilde{B} &= D^1 D^2 \cdots D^\kappa \end{aligned}$$

pooling

un-pooling

$$E^l = \begin{bmatrix} \Phi^l \circledast \psi^l_{1,1} & \cdots & \Phi^l \circledast \psi^l_{q_l,1} \\ \vdots & \ddots & \vdots \\ \Phi^l \circledast \psi^l_{1,q_{l-1}} & \cdots & \Phi^l \circledast \psi^l_{q_l,q_{l-1}} \end{bmatrix} \quad D^l = \begin{bmatrix} \tilde{\Phi}^l \circledast \tilde{\psi}^l_{1,1} & \cdots & \tilde{\Phi}^l \circledast \tilde{\psi}^l_{1,q_l} \\ \vdots & \ddots & \vdots \\ \tilde{\Phi}^l \circledast \tilde{\psi}^l_{q_{l-1},1} & \cdots & \tilde{\Phi}^l \circledast \tilde{\psi}^l_{q_{l-1},q_l} \end{bmatrix}$$

Learned filters

Linear E-D CNN w/ Skipped Connection

$$y = \tilde{B} B^\top x = \sum_i \langle x, b_i \rangle \tilde{b}_i$$

$$\begin{aligned} B &= [E^1 \cdots E^\kappa \quad E^1 \cdots E^{\kappa-1} S^\kappa \quad \cdots \quad E^1 S^2 \quad S^1] \\ \tilde{B} &= [D^1 \cdots D^\kappa \quad D^1 \cdots D^{\kappa-1} \tilde{S}^\kappa \quad \cdots \quad D^1 \tilde{S}^2 \quad \tilde{S}^1] \end{aligned}$$

more redundant expression

$$S^l = \begin{bmatrix} I_{m_{l-1}} \circledast \psi^l_{1,1} & \cdots & I_{m_{l-1}} \circledast \psi^l_{q_l,1} \\ \vdots & \ddots & \vdots \\ I_{m_{l-1}} \circledast \psi^l_{1,q_{l-1}} & \cdots & I_{m_{l-1}} \circledast \psi^l_{q_l,q_{l-1}} \end{bmatrix}$$

$$\tilde{S}^l = \begin{bmatrix} I_{m_{l-1}} \circledast \tilde{\psi}^l_{1,1} & \cdots & I_{m_{l-1}} \circledast \tilde{\psi}^l_{1,q_l} \\ \vdots & \ddots & \vdots \\ I_{m_{l-1}} \circledast \tilde{\psi}^l_{q_{l-1},1} & \cdots & I_{m_{l-1}} \circledast \tilde{\psi}^l_{q_{l-1},q_l} \end{bmatrix}$$

Learned filters

Deep Convolutional Framelets

Perfect reconstruction

$$x = \tilde{B} B^\top x = \sum_i \langle x, b_i \rangle \tilde{b}_i$$

Frame conditions

w/o skipped connection

$$\tilde{\Phi}^l \Phi^{l\top} = \alpha I_{m_{l-1}}, \; \Psi^l \tilde{\Psi}^{l\top} = \frac{1}{r\alpha} I_{rq_{l-1}}$$

w skipped connection

$$\tilde{\Phi}^l \Phi^{l\top} = \alpha I_{m_{l-1}}, \; \Psi^l \tilde{\Psi}^{l\top} = \frac{1}{r(\alpha+1)} I_{rq_{l-1}}$$

Ye et al, SIAM J. Imaging Science, 2018

Role of ReLUs?
Generator for Multiple Expressions

$$y = \tilde{B}(x) B(x)^\top x = \sum_i \langle x, b_i(x) \rangle \tilde{b}_i(x)$$

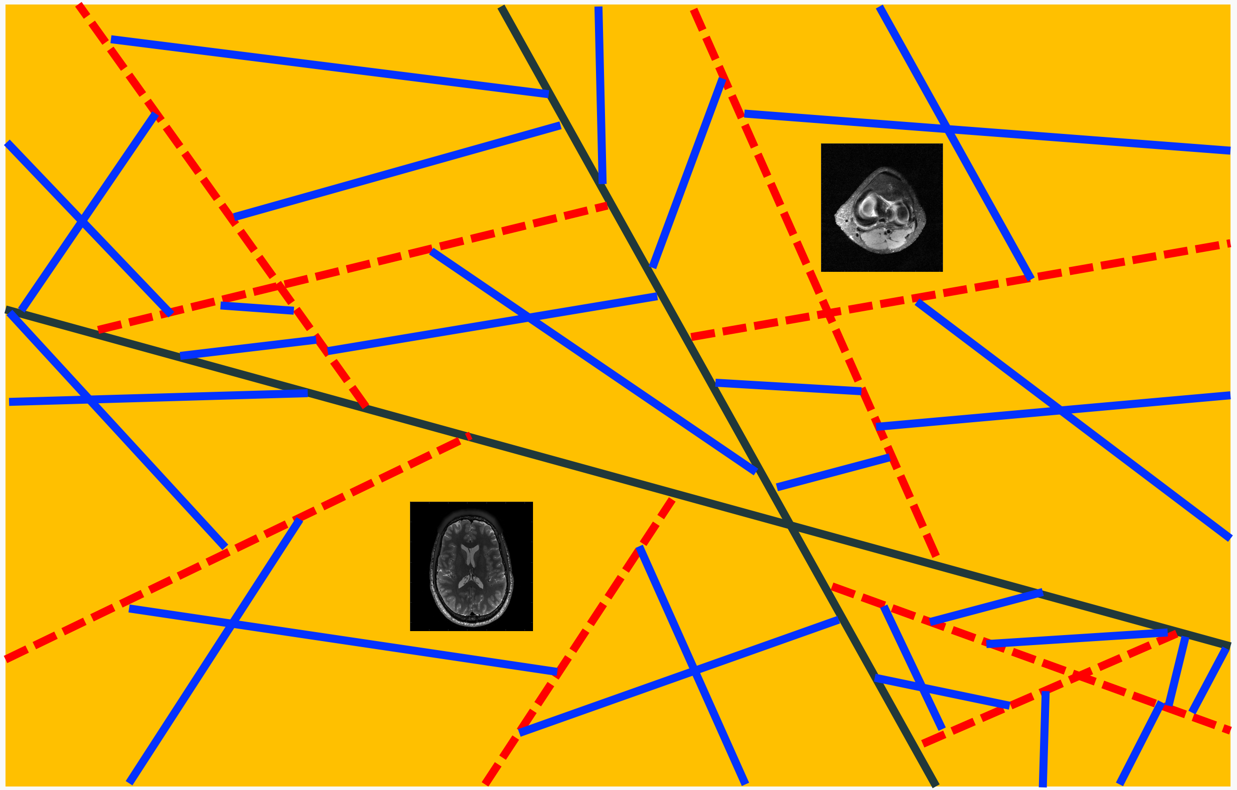
$$\begin{aligned} B(x) &= E^1 \Sigma^1(x) E^2 \cdots \Sigma^{\kappa-1}(x) E^\kappa, \\ \tilde{B}(x) &= D^1 \tilde{\Sigma}^1(x) D^2 \cdots \tilde{\Sigma}^{\kappa-1}(x) D^\kappa \end{aligned}$$

$$\Sigma^l(x) = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_{m_l} \end{bmatrix}$$

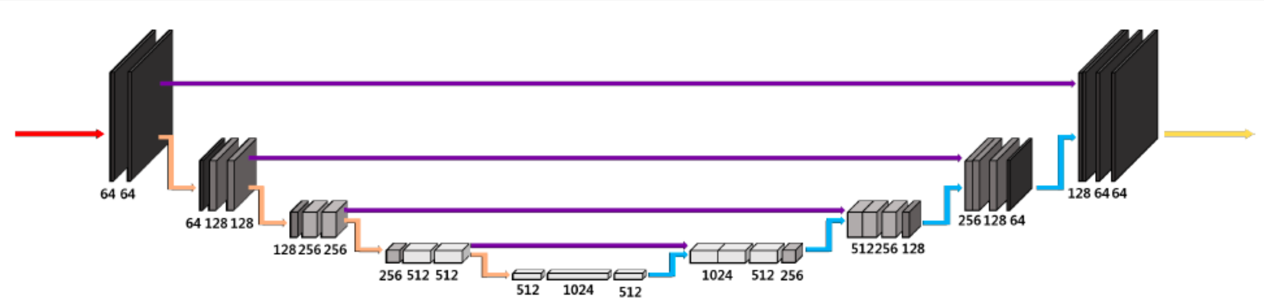
Input dependent {0,1} matrix

--> Input adaptivity

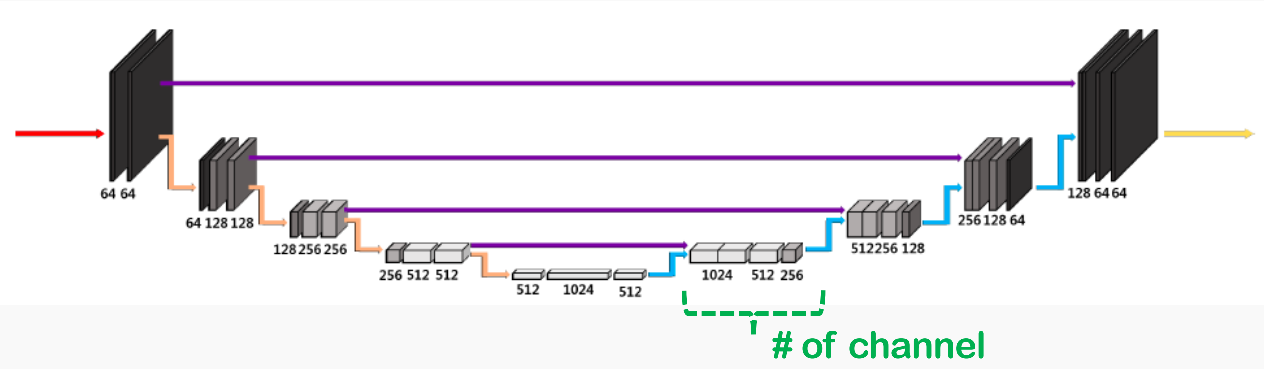
Input Space Partitioning for Multiple Expressions

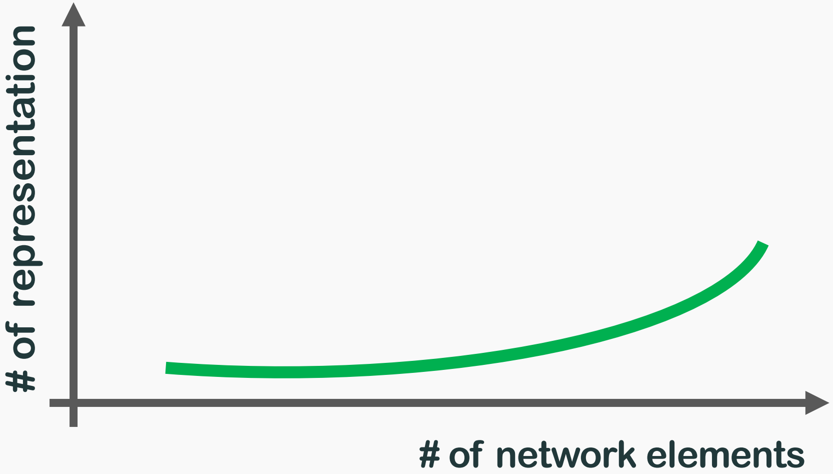


Expressivity of E-D CNN

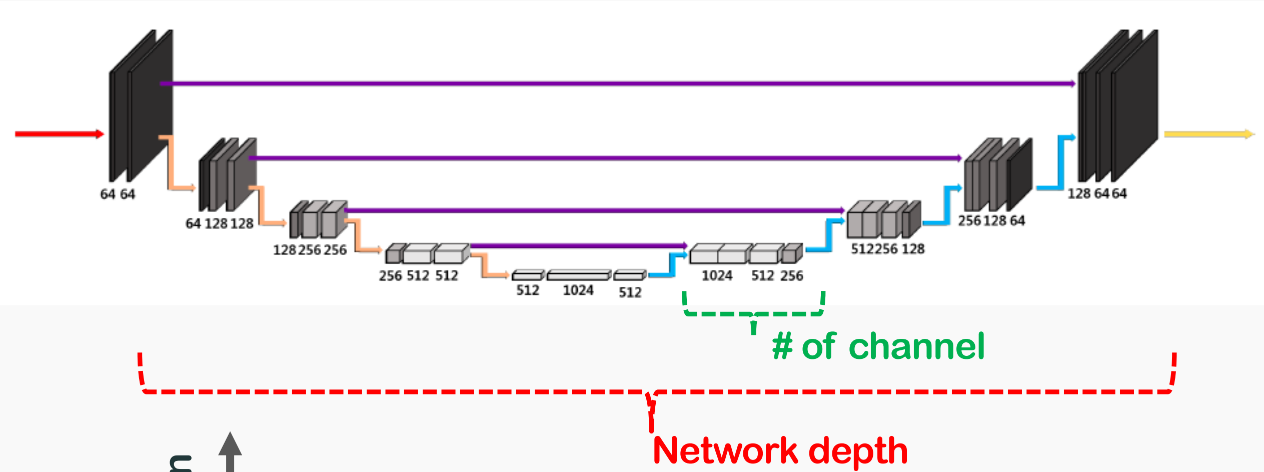


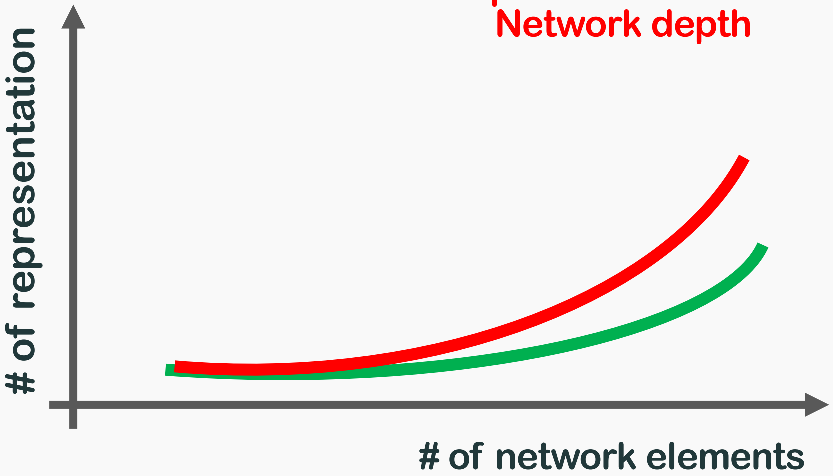
Expressivity of E-D CNN



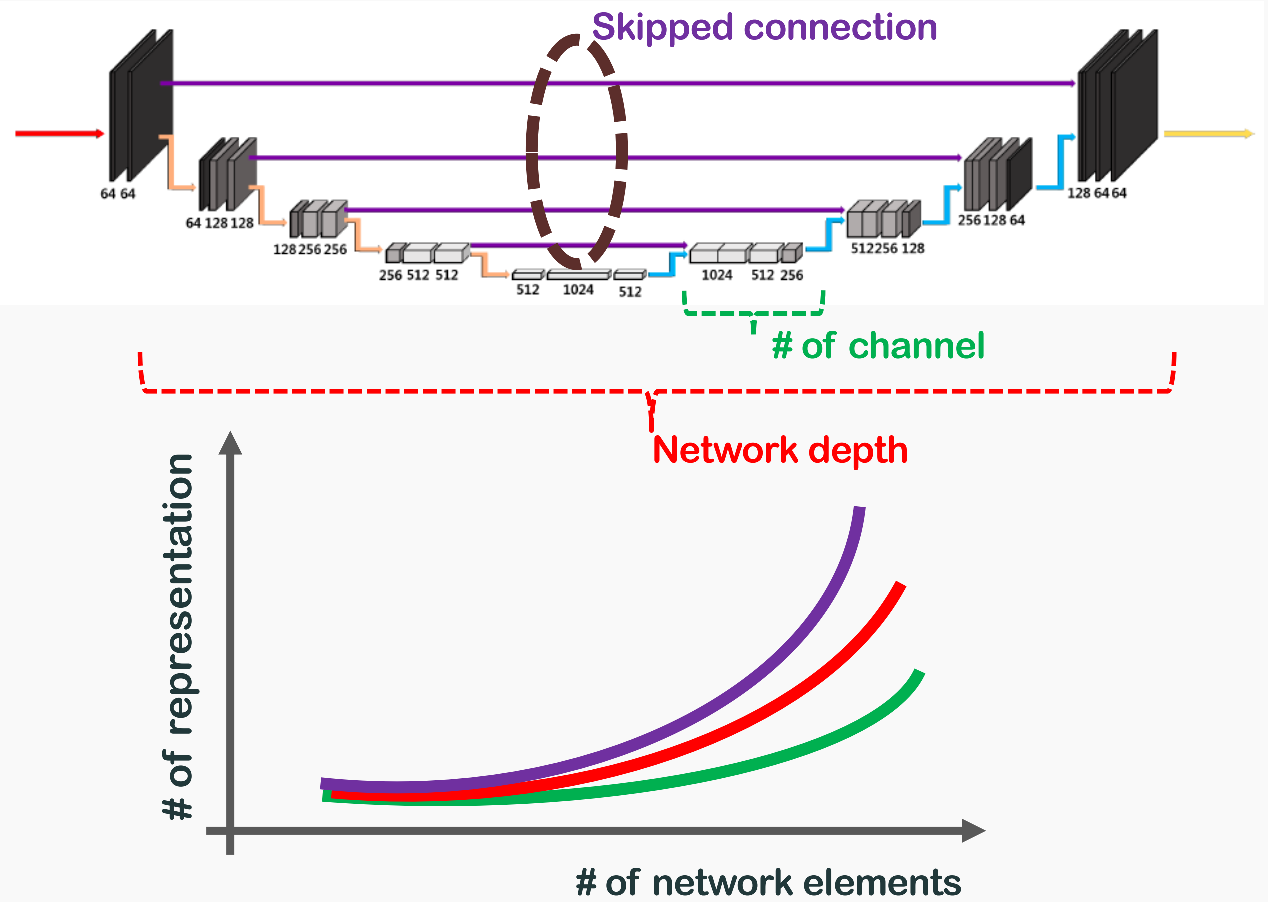


Expressivity of E-D CNN





Expressivity of E-D CNN



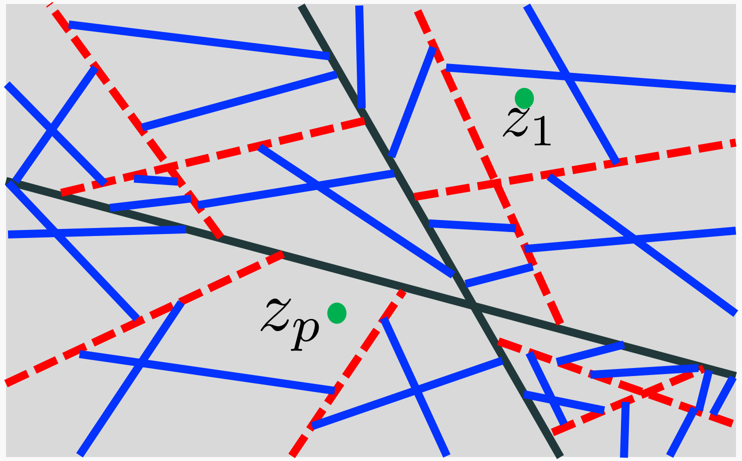
Lipschitz Continuity

Related to the generalizability

$$\|F(\mathbf{W}, x^{(1)}) - F(\mathbf{W}, x^{(2)})\|_2 \leq K\|x^{(1)} - x^{(2)}\|_2$$

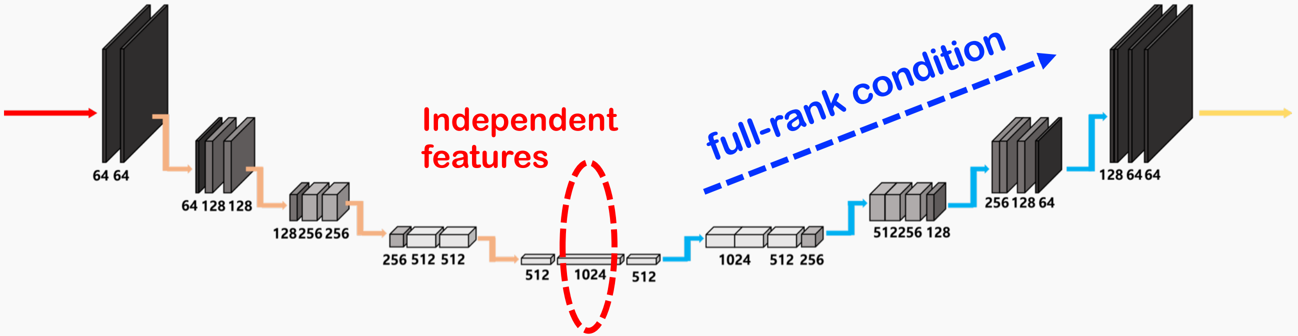
$$K = \max_p K_p, \quad K_p = \|\tilde{B}(z_p)B(z_p)^\top\|_2$$

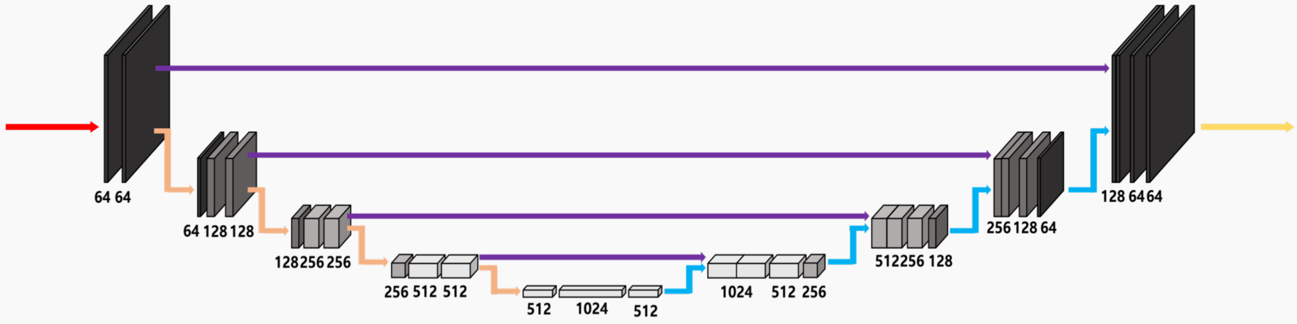
Dependent on the Local Lipschitz



Benign Optimization Landscape

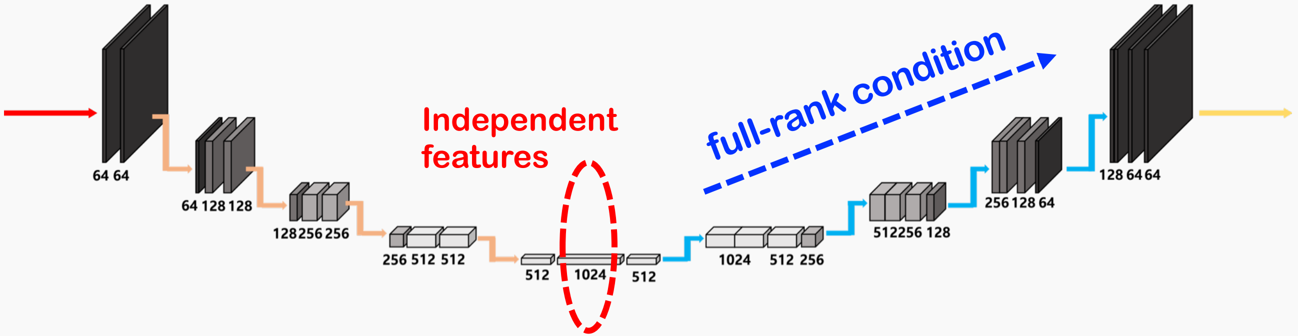
Nguyen, et al, ICML, 2018



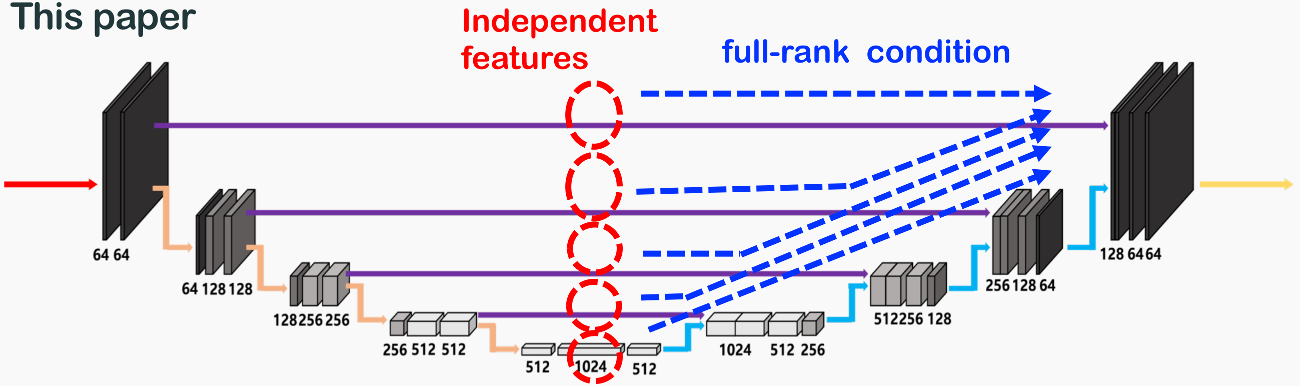


Benign Optimization Landscape

Nguyen, et al, ICML, 2018



This paper



Summary

- Deep learning is a novel signal representation using combinatorial framelets
- ReLUs generate multiple linear representation by partitioning the input space
- Local Lipschitz controls the global Liptschiz continuity
- Skipped connection improves the optimization landscape

Poster #99: 06:30 -- 09:00 PM @ Pacific Ballroom