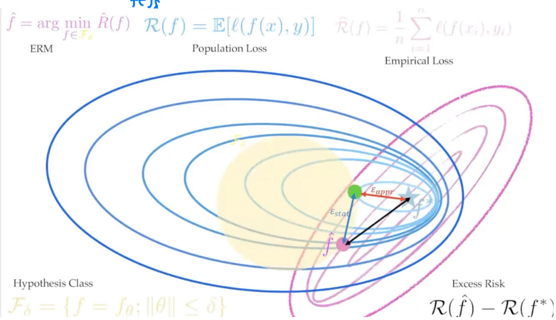
Geometric Learning

Question: Can we reduce the complexity of our hypothesis class without increasing the approximation error?

Answer: Using geometric priors (e.g. symmetry and scale separations), we can decrease the complexity of the hypothesis class while keeping the approximation error constant.

Recap: Error Analysis

$$\leq \varepsilon_{opt} + 2\sup_{f\in F_\delta} |R(f) - \hat{R}(f)| + \varepsilon_{appr} = \varepsilon_{opt} + \varepsilon_{stat} + \varepsilon_{appr}$$



Error Analysis

: Let's say we found $\hat{f} \in F_\delta$ to solve the problem. How good is it?

Mathematically, how small is the population error? : $R(\hat{f}) - \inf_{f\in F} R(f)$

Statistical (ε_{stat}): Gap between $R(\hat{f})$ and $\hat{R}(\hat{f})$

Optimization (ε_{opt}) : How far is $\hat{f}$ from the best $f \in F_\delta$

Approximation (ε_{appr}) : How good is F_δ? How far is the best $f \in F_\delta$ from f^*?

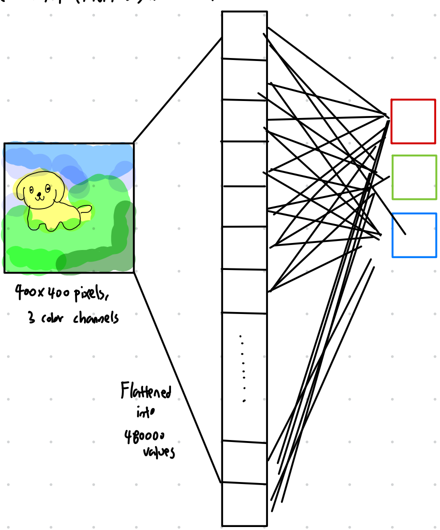
How to reduce ε_{opt}? : Use a better optimization

ε_{appr}? : Use a larger hypothesis class F_δ

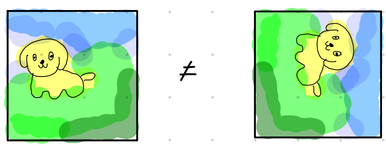
ε_{stat}? : # of data samples,
Complexity of the hypothesis class, F_δ

$$\varepsilon_{stat} = R(\hat{f}) - \hat{R}(\hat{f}) \leq c \cdot \frac{\text{Complexity}}{n^{\frac{1}{d}}}$$

Where MLP (Multi-Layer Perceptron)s fall short?



However,



MLP model cannot distinguish transformed (shifted, rotated, reflected) data, even data is symmetry.

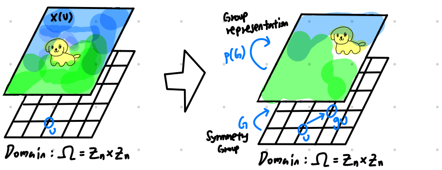
Similarly, changing the indices of the graph won't change the underlying graph.

Geometric Priors

· Geometric prior: The input is a signal defined on a domain with a geometric structure.

Signals $X(\Omega) := \{x \mid x : \Omega \to C\}$

$C = \mathbb{R}^3$ $\quad P(g)x(u) = x(g^{-1}u)$



Definition of Group

: A group is a set G along with a binary operation $\circ : G \times G \to G$

① Associativity: $(g \circ h) \circ f = g \circ (h \circ f) \quad \forall g, h, f \in G$

② Identity: $I \circ g = g \circ I = g \quad \forall g \in G$

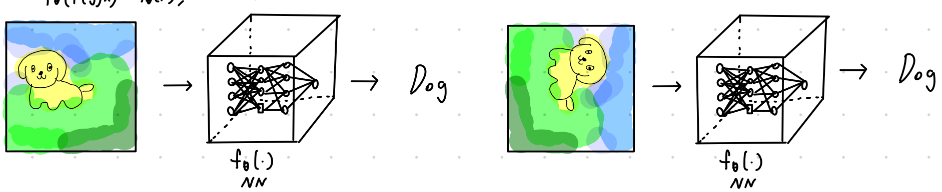
③ Inverse: $g \circ g^{-1} = I = g^{-1} g$

④ Closure: $g \circ h \in G, \ \forall g, h \in G$

For brevity we often write gh instead of $g \circ h$

Geometric Priors: G-invariance

$f_\theta(P(g)x) = f_\theta(x)$, where $P(g)x$. transformed signal



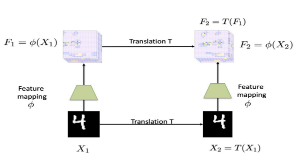
Geometric Priors: G-equivariance

$f_\theta(P(g)x) = P(g) f_\theta(x)$

How can we build equivariant functions?

Meaning of equivariant function: $f_\theta(P(g(x))) = P(g) f_\theta(x)$

Pictorial depiction



From Christian Wolf

$F_1 = \phi(X_1) = f_\theta(x)$

$T = P(g)$

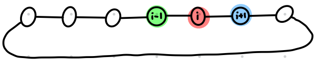
$X_2 = P(g)x = T(X_1)$

Then, as you can see,

$T(F_1) = \phi(X_2)$

$\Rightarrow P(g) f_\theta(x) = f_\theta(P(g)x)$, hence it's equivariant.

Grid : ring graph



local aggregation function

$$f(x_i) = \phi(x_i, \{x_{i-1}, x_{i+1}\}) = \phi(x_{i-1}, x_i, x_{i+1})$$

Linear local aggregation function

$$f(x_i) = ax_{i-1} + bx_i + cx_{i+1}$$

Convolution

vector of parameters θ

$$f(X) = \begin{bmatrix} b & c & 0 & 0 & a \\ a & b & c & 0 & 0 \\ & & \cdots & & \\ 0 & 0 & a & b & c \\ c & 0 & 0 & a & b \end{bmatrix} \begin{bmatrix} X \end{bmatrix}$$

Circulant matrix

= Convolution

ex) image :

$$\begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ x_5 & x_6 & x_7 & x_8 \\ x_9 & x_{10} & x_{11} & x_{12} \\ x_{13} & x_{14} & x_{15} & x_{16} \end{bmatrix}$$

$$\text{Convolution} = \begin{bmatrix} k_1 & k_2 & k_3 & 0 & k_4 & k_5 & k_6 & 0 & k_7 & k_8 & k_9 & 0 & 0 & 0 & 0 & 0 \\ 0 & k_1 & k_2 & X_3 & 0 & k_4 & k_5 & k_6 & 0 & k_7 & k_8 & k_9 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & k_1 & k_2 & k_3 & 0 & k_4 & k_5 & k_6 & 0 & k_7 & k_8 & k_9 & 0 \\ 0 & 0 & 0 & 0 & 0 & k_1 & k_2 & k_3 & 0 & k_4 & k_5 & k_6 & 0 & k_7 & k_8 & k_9 \end{bmatrix}$$

Deriving Convolution from Symmetry

$$\begin{bmatrix} b & c & 0 & 0 & a \\ a & b & c & 0 & 0 \\ 0 & a & b & c & 0 \\ 0 & 0 & a & b & c \\ c & 0 & 0 & a & b \end{bmatrix} \begin{bmatrix} y & z & 0 & 0 & x \\ x & y & z & 0 & 0 \\ 0 & x & y & z & 0 \\ 0 & 0 & x & y & z \\ z & 0 & 0 & x & y \end{bmatrix} = \begin{bmatrix} y & z & 0 & 0 & x \\ x & y & z & 0 & 0 \\ 0 & x & y & z & 0 \\ 0 & 0 & x & y & z \\ z & 0 & 0 & x & y \end{bmatrix} \begin{bmatrix} b & c & 0 & 0 & a \\ a & b & c & 0 & 0 \\ 0 & a & b & c & 0 \\ 0 & 0 & a & b & c \\ c & 0 & 0 & a & b \end{bmatrix}$$

circulant matrices commute

Ex)

$$\underbrace{\begin{bmatrix} b & c & 0 & 0 & a \\ a & b & c & 0 & 0 \\ 0 & a & b & c & 0 \\ 0 & 0 & a & b & c \\ c & 0 & 0 & a & b \end{bmatrix}}_{\text{Convolution}} \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}}_{g:\ \text{shifting}} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} b & c & 0 & 0 & a \\ a & b & c & 0 & 0 \\ 0 & a & b & c & 0 \\ 0 & 0 & a & b & c \\ c & 0 & 0 & a & b \end{bmatrix}$$

$$\Rightarrow \underbrace{\begin{bmatrix} a & b & c & 0 & 0 \\ 0 & a & b & c & 0 \\ 0 & 0 & a & b & c \\ c & 0 & 0 & a & b \\ b & c & 0 & 0 & a \end{bmatrix}}_{CS} = \underbrace{\begin{bmatrix} a & b & c & 0 & 0 \\ 0 & a & b & c & 0 \\ 0 & 0 & a & b & c \\ c & 0 & 0 & a & b \\ b & c & 0 & 0 & a \end{bmatrix}}_{SC}$$

Hence, $CS = SC$ iff Convolution is circulant matrix.

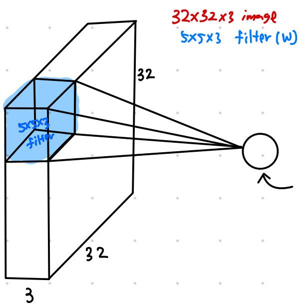
Recap: Geometric Priors: G-equivariance

$$f_\theta(\rho(g)x) = \rho(g) f_\theta(x)$$

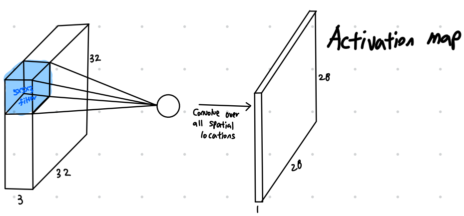
Therefore, Convolution matrix (circulant matrix) is shift-equivariant.

Convolution $\Leftrightarrow$ shift-equivariant.

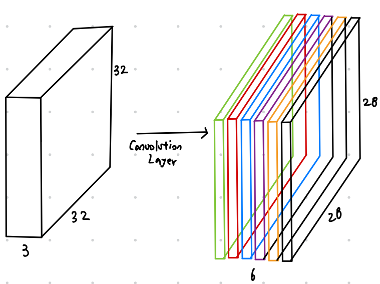
Convolution Operator



1 number:
the result of taking a dot product between
the filter and a small 5x5x3 chunk of the image : $w^T x + b$
bias

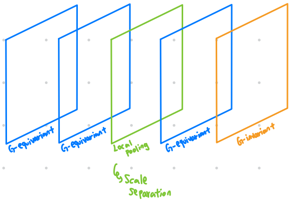


For example, if we had 6 5x5 filters, we will get 6 separate activation maps:

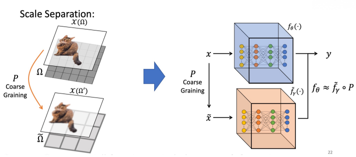


We stack these up to get a new image of size 28x28x6

Geometric Deep Learning - A Blueprint

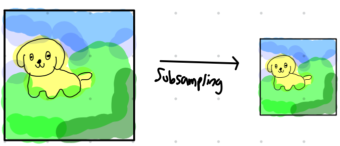


Scale Separation:



Why pooling?

: Subsampling pixels will not change the object



- We can subsample the pixels to make image smaller, fewer parameters to characterize the image

Max Pooling

$$\begin{bmatrix} 3 & -1 & -3 & -1 \\ -3 & 1 & 0 & -3 \\ -3 & -3 & 0 & 1 \\ 3 & -2 & -2 & -1 \end{bmatrix} \rightarrow \begin{bmatrix} 3 & 0 \\ 3 & 1 \end{bmatrix}$$