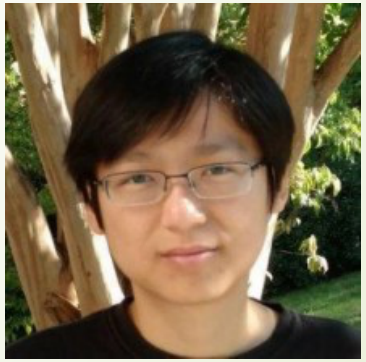
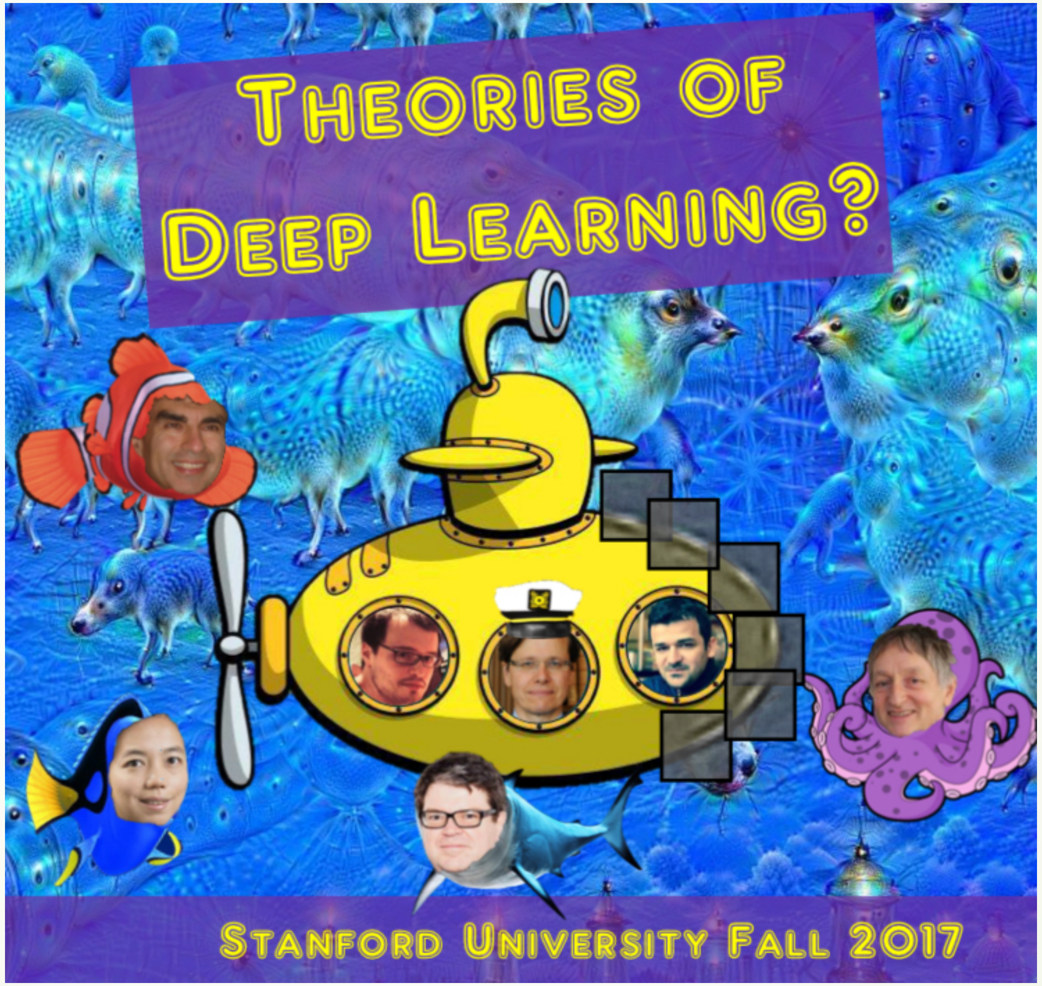
# Symmetry and Network Architectures

Yuan YAO
HKUST
Based on Mallat, Bolcskei, Cheng talks etc.



1

# Acknowledgement



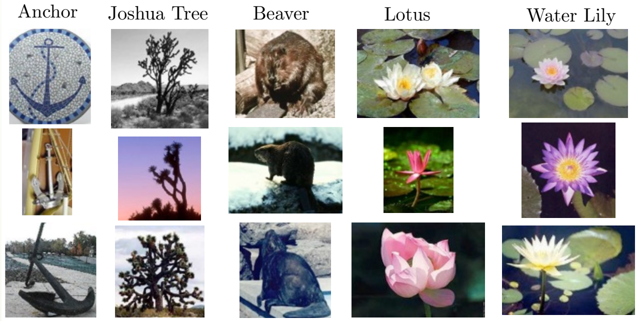

A following-up course at HKUST: https://deeplearning-math.github.io/

# Last time, a good representation learning in classification is:

- Contraction within level set symmetries toward invariance when depth grows (invariants)
- Separation kept between different levels (discriminant)

- High-dimensional $x = (x(1), ..., x(d)) \in \mathbb{R}^d$:
- **Classification:** estimate a class label $f(x)$ given $n$ sample values $\{x_i\,,\, y_i = f(x_i)\}_{i \leq n}$

Image Classification $d = 10^6$




Huge variability inside classes

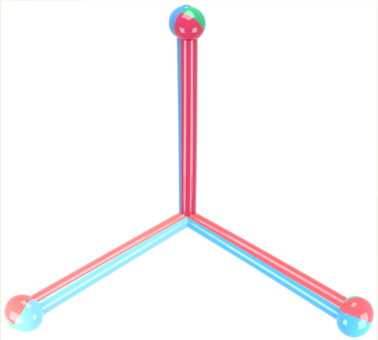
Find invariants

# Prevalence of Neural Collapse during the terminal phase of deep learning training

Papyan, Han, and Donoho (2020), PNAS. arXiv:2008.08186

# **Neural Collapse** phenomena, in post-zero-training-error phase

- **(NC1) Variability collapse:** As training progresses, the within-class variation of the activations becomes negligible as these activations collapse to their class-means.
- **(NC2) Convergence to Simplex ETF:** The vectors of the class-means (after centering by their global-mean) converge to having equal length, forming equal-sized angles between any given pair, and being the maximally pairwise-distanced configuration constrained to the previous two properties. This configuration is identical to a previously studied configuration in the mathematical sciences known as Simplex **Equiangular Tight Frame (ETF).**
- Visualization: https://purl.stanford.edu/br193mh4244



*Definition* 1 (**Simplex ETF**). A *standard* Simplex ETF is a collection of points in $\mathbb{R}^C$ specified by the columns of

$$\boldsymbol{M}^\star = \sqrt{\frac{C}{C-1}}\left(\boldsymbol{I} - \frac{1}{C}\mathbb{1}\mathbb{1}^\top\right), \tag{1}$$

where $\boldsymbol{I} \in \mathbb{R}^{C\times C}$ is the identity matrix, and $\mathbb{1}_C \in \mathbb{R}^C$ is the ones vector. In this paper, we allow other poses, as well as rescaling, so the *general* Simplex ETF consists of the points specified by the columns of $\boldsymbol{M} = \alpha\boldsymbol{U}\boldsymbol{M}^\star \in \mathbb{R}^{p\times C}$, where $\alpha \in \mathbb{R}_+$ is a scale factor, and $\boldsymbol{U} \in \mathbb{R}^{p\times C}$ $(p \geq C)$ is a partial orthogonal matrix $(\boldsymbol{U}^\top\boldsymbol{U} = \boldsymbol{I})$.

# Notations

- Feature layer:

$$\boldsymbol{h} = \boldsymbol{h}_{\boldsymbol{\theta}}(\boldsymbol{x})$$

- Classification layer:

$$\arg\max_{c'} \langle \boldsymbol{w}_{c'}, \boldsymbol{h} \rangle + b_{c'}$$

For a given dataset-network combination, we calculate the train global-mean $\boldsymbol{\mu}_G \in \mathbb{R}^p$:

$$\boldsymbol{\mu}_G \triangleq \operatorname*{Ave}_{i,c}\{\boldsymbol{h}_{i,c}\},$$

and the train class-means $\boldsymbol{\mu}_c \in \mathbb{R}^p$:

$$\boldsymbol{\mu}_c \triangleq \operatorname*{Ave}_{i}\{\boldsymbol{h}_{i,c}\}, \quad c = 1, \dots, C,$$

where Ave is the averaging operator.

Given the train class-means, we calculate the train total covariance $\mathbf{\Sigma}_T \in \mathbb{R}^{p\times p}$,

$$\mathbf{\Sigma}_T \triangleq \operatorname*{Ave}_{i,c}\left\{\left(\boldsymbol{h}_{i,c}-\boldsymbol{\mu}_G\right)\left(\boldsymbol{h}_{i,c}-\boldsymbol{\mu}_G\right)^\top\right\},$$

the between-class covariance, $\mathbf{\Sigma}_B \in \mathbb{R}^{p\times p}$,

$$\mathbf{\Sigma}_B \triangleq \operatorname*{Ave}_{c}\{(\boldsymbol{\mu}_c-\boldsymbol{\mu}_G)(\boldsymbol{\mu}_c-\boldsymbol{\mu}_G)^\top\}, \qquad [3]$$

and the within-class covariance, $\mathbf{\Sigma}_W \in \mathbb{R}^{p\times p}$,

$$\mathbf{\Sigma}_W \triangleq \operatorname*{Ave}_{i,c}\{(\boldsymbol{h}_{i,c}-\boldsymbol{\mu}_c)(\boldsymbol{h}_{i,c}-\boldsymbol{\mu}_c)^\top\}. \qquad [4]$$

# Neural Collapse of Features

**(NC1) Variability collapse:** $\mathbf{\Sigma}_W \rightarrow \mathbf{0}$

**(NC2) Convergence to Simplex ETF:**

$$\left| \|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2 - \|\boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G\|_2 \right| \rightarrow 0 \quad \forall\ c, c'$$

$$\langle \tilde{\boldsymbol{\mu}}_c, \tilde{\boldsymbol{\mu}}_{c'} \rangle \rightarrow \frac{C}{C-1}\delta_{c,c'} - \frac{1}{C-1} \quad \forall\ c, c'.$$

$$\tilde{\boldsymbol{\mu}}_c = (\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)/\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2$$

# Neural Collapse of Classifiers

**(NC3) Convergence to self-duality:**

$$\left\| \frac{\boldsymbol{W}^\top}{\|\boldsymbol{W}\|_F} - \frac{\dot{\boldsymbol{M}}}{\|\dot{\boldsymbol{M}}\|_F} \right\|_F \to 0 \qquad [5]$$

**(NC4): Simplification to NCC:**

$$\arg\max_{c'} \langle \boldsymbol{w}_{c'}, \boldsymbol{h} \rangle + b_{c'} \to \arg\min_{c'} \|\boldsymbol{h} - \boldsymbol{\mu}_{c'}\|_2$$

where $\tilde{\boldsymbol{\mu}}_c = (\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)/\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2$ are the renormalized the class-means, $\dot{\boldsymbol{M}} = [\boldsymbol{\mu}_c - \boldsymbol{\mu}_G, c = 1, \ldots, C] \in \mathbb{R}^{p \times C}$ is the matrix obtained by stacking the class-means into the columns of a matrix, and $\delta_{c,c'}$ is the Kronecker delta symbol.

# 7 Datasets:

- MNIST, FashionMNIST, CI- FAR10, CIFAR100, SVHN, STL10 and ImageNet datasets
- MNIST was sub-sampled to N=5000 examples per class, SVHN to N=4600 examples per class, and ImageNet to N=600 examples per class.
- The remaining datasets are already balanced.
- The images were pre-processed, pixel-wise, by subtracting the mean and dividing by the standard deviation.
- No data augmentation was used.

# 3 Models: VGG/ResNet/DenseNet

- VGG19, ResNet152, and DenseNet201 for ImageNet;
- VGG13, ResNet50, and DenseNet250 for STL10;
- VGG13, ResNet50, and DenseNet250 for CIFAR100;
- VGG13, ResNet18, and DenseNet40 for CIFAR10;
- VGG11, ResNet18, and DenseNet250 for FashionMNIST;
- VGG11, ResNet18, and DenseNet40 for MNIST and SVHN.

# Results

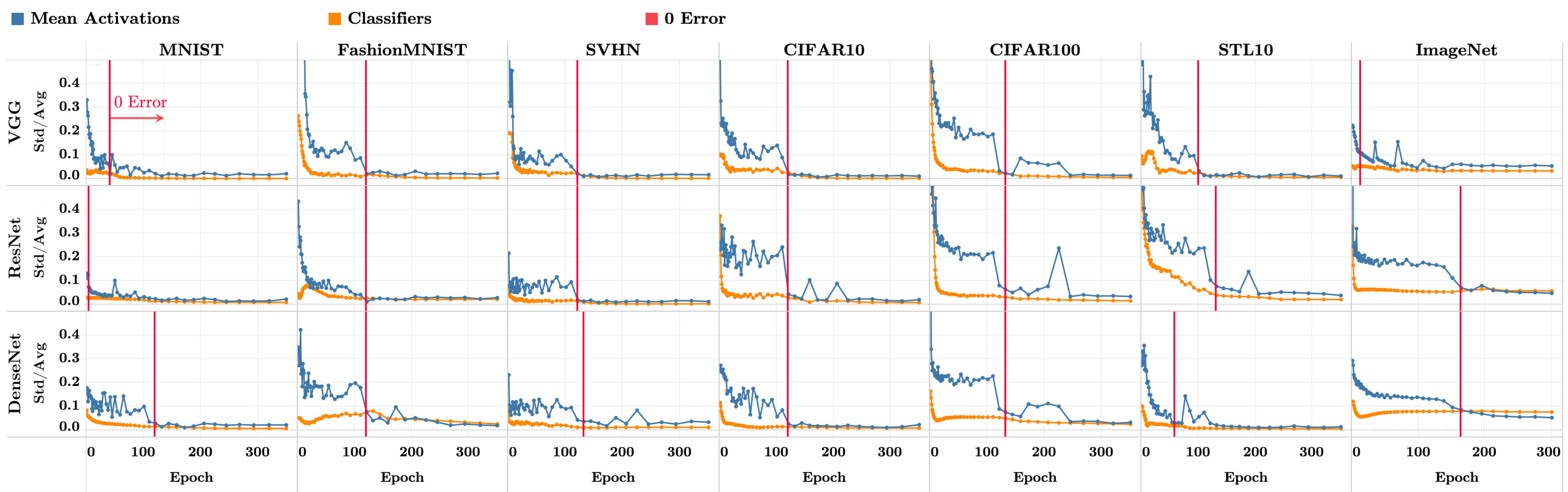



**Fig. 2. Train class-means become equinorm:** The formatting and technical details are as described in Section 3. In each array cell, the vertical axis shows the coefficient of variation of the centered class-mean norms as well as the network classifiers norms. In particular, the blue line shows $\text{Std}_c(\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2)/\text{Avg}_c(\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2)$ where $\{\boldsymbol{\mu}_c\}$ are the class-means of the last-layer activations of the training data and $\boldsymbol{\mu}_G$ is the corresponding train global-mean; the orange line shows $\text{Std}_c(\|\boldsymbol{w}_c\|_2)/\text{Avg}_c(\|\boldsymbol{w}_c\|_2)$ where $\boldsymbol{w}_c$ is the last-layer classifier of the $c$-th class. As training progresses, the coefficients of variation of both class-means and classifiers decreases.

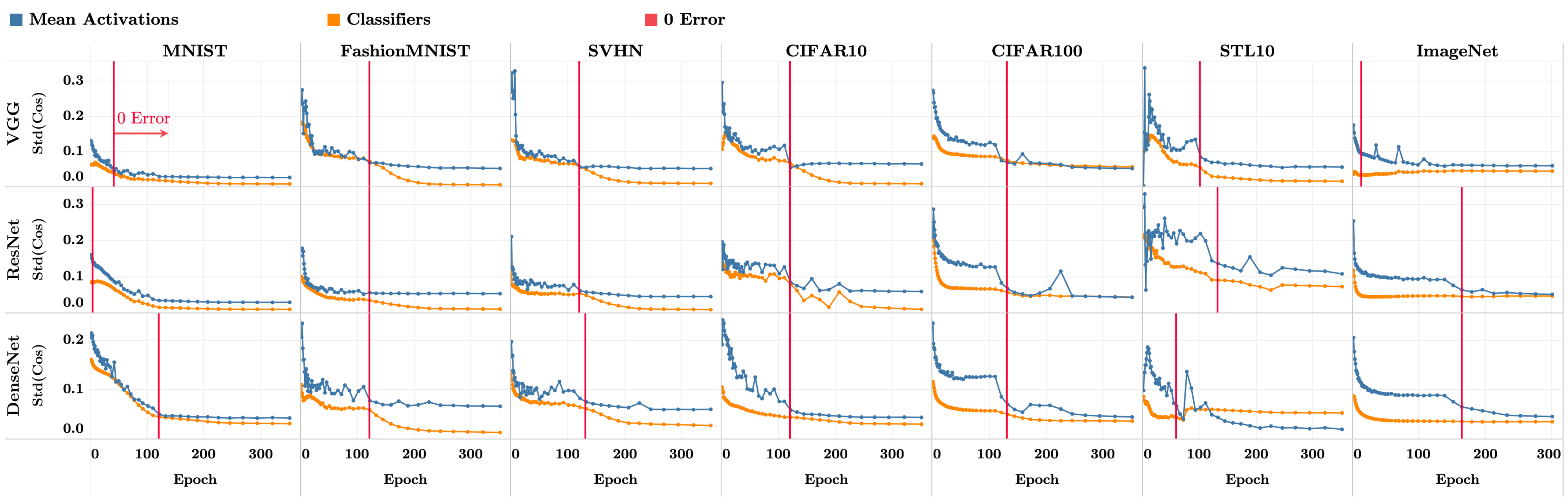



**Fig. 3. Classifiers and train class-means approach equiangularity:** The formatting and technical details are as described in Section 3. In each array cell, the vertical axis shows the standard deviation of the cosines between pairs of centered class-means and classifiers across all distinct pairs of classes $c$ and $c'$. Mathematically, denote $\cos_{\boldsymbol{\mu}}(c, c') = \langle \boldsymbol{\mu}_c - \boldsymbol{\mu}_G, \boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G \rangle / (\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2 \|\boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G\|_2$ and $\cos_{\boldsymbol{w}}(c, c') = \langle \boldsymbol{w}_c, \boldsymbol{w}_{c'} \rangle / (\|\boldsymbol{w}_c\|_2 \|\boldsymbol{w}_{c'}\|_2)$ where $\{\boldsymbol{w}_c\}_{c=1}^C$, $\{\boldsymbol{\mu}_c\}_{c=1}^C$, and $\boldsymbol{\mu}_G$ are as in Figure 2. We measure $\text{Std}_{c,c'\neq c}(\cos_{\boldsymbol{\mu}}(c, c'))$ (**blue**) and $\text{Std}_{c,c'\neq c}(\cos_{\boldsymbol{w}}(c, c'))$ (**orange**). As training progresses, the standard deviations of the cosines approach zero indicating equiangularity.

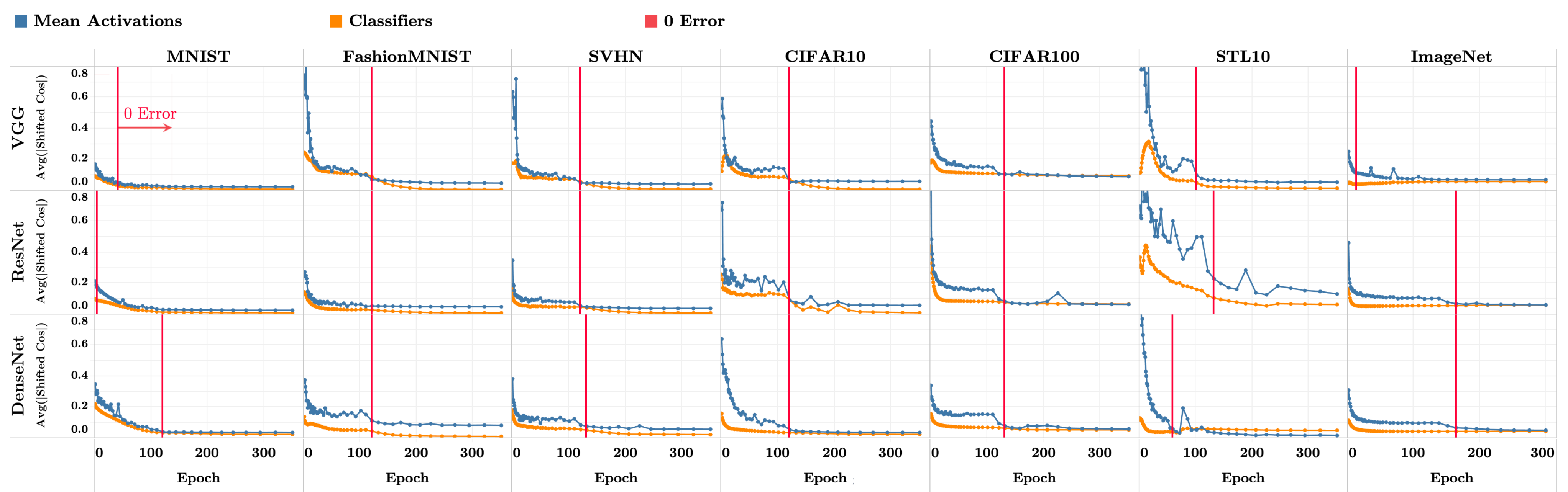



**Fig. 4. Classifiers and train class-means approach maximal-angle equiangularity:** The formatting and technical details are as described in Section 3. We plot in the vertical axis of each cell the quantities $\text{Avg}_{c,c'} |\cos_{\boldsymbol{\mu}}(c,c') + 1/(C-1)|$ (**blue**) and $\text{Avg}_{c,c'} |\cos_{\boldsymbol{w}}(c,c') + 1/(C-1)|$ (**orange**), where $\cos_{\boldsymbol{\mu}}(c,c')$ and $\cos_{\boldsymbol{w}}(c,c')$ are as in Figure 3. As training progresses, the convergence of these values to zero implies that all cosines converge to $-1/(C-1)$. This corresponds to the maximum separation possible for globally centered, equiangular vectors.

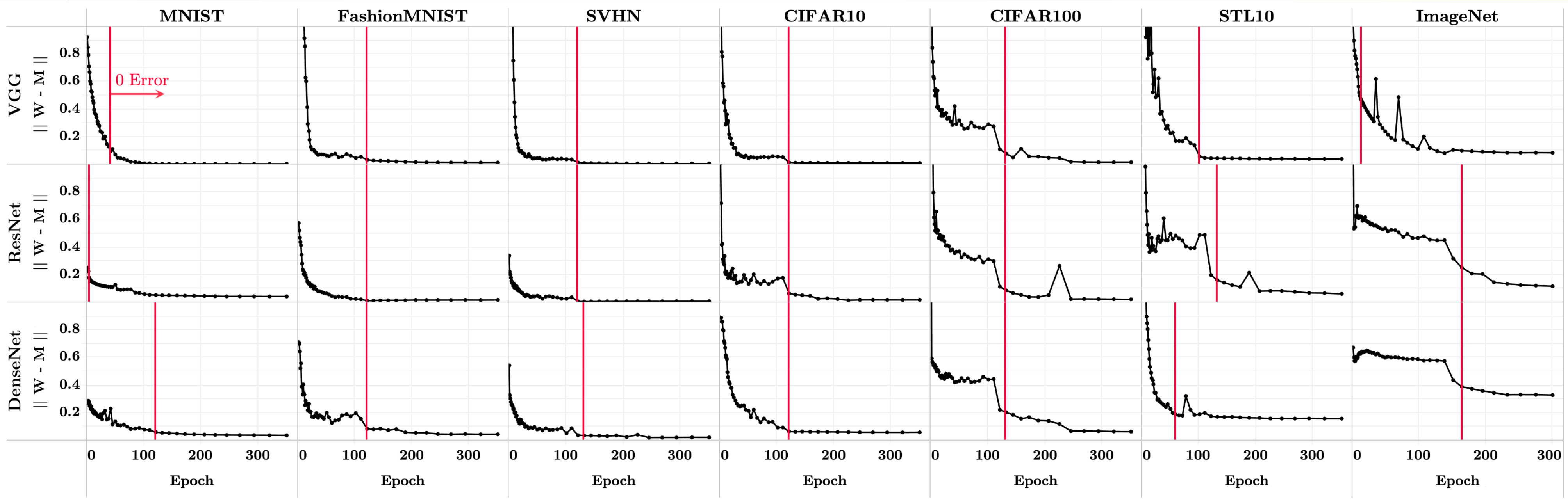



**Fig. 5. Classifier converges to train class-means:** The formatting and technical details are as described in Section 3. In the vertical axis of each cell, we measure the distance between the classifiers and the centered class-means, both rescaled to unit-norm. Mathematically, denote $\widetilde{\boldsymbol{M}} = \dot{\boldsymbol{M}}/\|\dot{\boldsymbol{M}}\|_F$ where $\dot{\boldsymbol{M}} = [\boldsymbol{\mu}_c - \boldsymbol{\mu}_G : c = 1, \ldots, C] \in \mathbb{R}^{p \times C}$ is the matrix whose columns consist of the centered train class-means; denote $\widetilde{\boldsymbol{W}} = \boldsymbol{W}/\|\boldsymbol{W}\|_F$ where $\boldsymbol{W} \in \mathbb{R}^{C \times p}$ is the last-layer classifier of the network. We plot the quantity $\|\widetilde{\boldsymbol{W}}^\top - \widetilde{\boldsymbol{M}}\|_F^2$ on the vertical axis. This value decreases as a function of training, indicating the network classifier and the centered-means matrices become proportional to each other (self-duality).

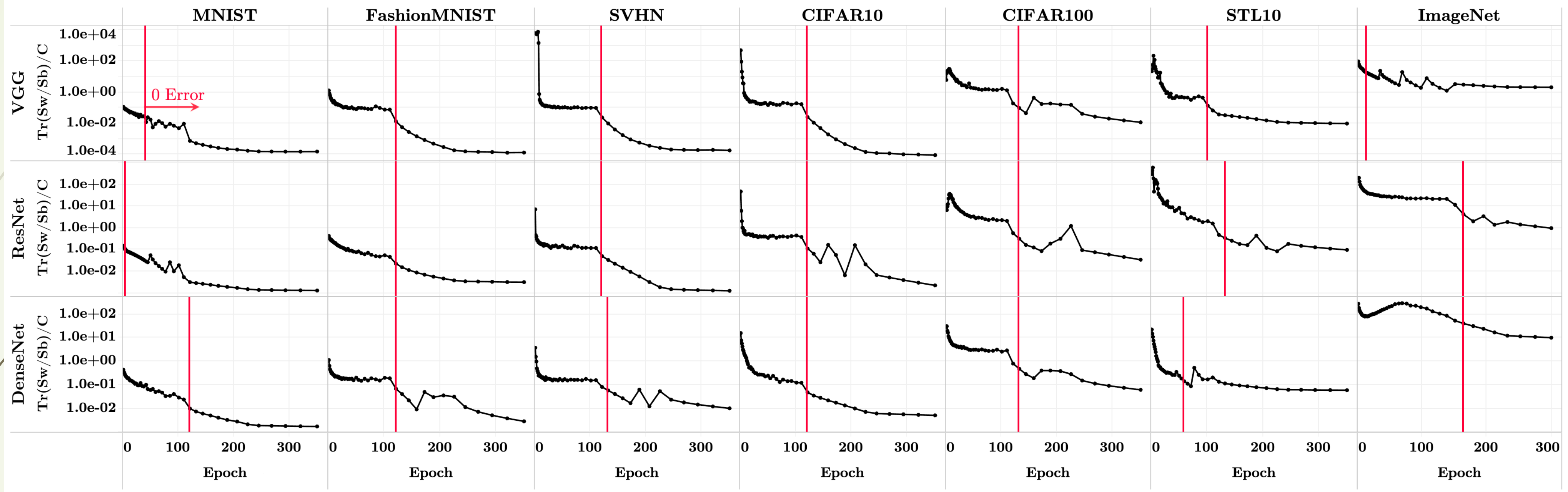



**Fig. 6. Training within-class variation collapses:** The formatting and technical details are as described in Section 3. In each array cell, the vertical axis (log-scaled) shows the magnitude of the between-class covariance compared to the within-class covariance of the train activations . Mathematically, this is represented by $\mathrm{Tr}\left\{\boldsymbol{\Sigma}_W \boldsymbol{\Sigma}_B^\dagger\right\}/C$ where $\mathrm{Tr}\{\cdot\}$ is the trace operator, $\boldsymbol{\Sigma}_W$ is the within-class covariance of the last-layer activations of the training data, $\boldsymbol{\Sigma}_B$ is the corresponding between-class covariance, $C$ is the total number of classes, and $[\cdot]^\dagger$ is Moore-Penrose pseudoinverse. This value decreases as a function of training – indicating collapse of within-class variation.

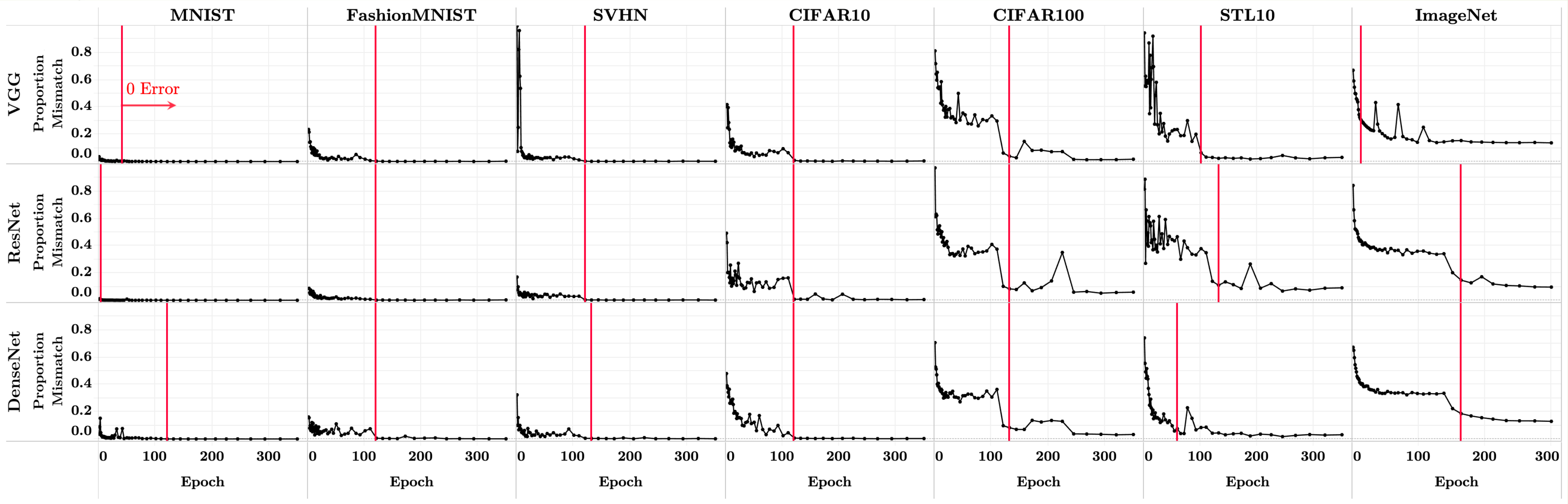



**Fig. 7. Classifier behavior approaches that of Nearest Class-Center:** The formatting and technical details are as described in Section 3. In each array cell, we plot the proportion of examples (vertical axis) in the *testing* set where network classifier disagrees with the result that would have been obtained by choosing $\arg\min_c \|\boldsymbol{h} - \boldsymbol{\mu}_c\|_2$ where $\boldsymbol{h}$ is a last-layer test activation, and $\{\boldsymbol{\mu}_c\}_{c=1}^{C}$ are the class-means of the last-layer train activations. As training progresses, the disagreement tends to zero, showing the classifier's behavioral simplification to the nearest train class-mean decision rule.

# Propositions

- LDA:
  - NC1 +
  - NC2 +
  - Linear Discriminant Analysis (LDA)

→ NC3 + NC4 (nearest neighbor classifier)

- Max-Margin classifier:
  - NC1 +
  - NC2 +
  - Max-Margin Classifier

→ NC3 + NC4 (nearest neighbor classifier)

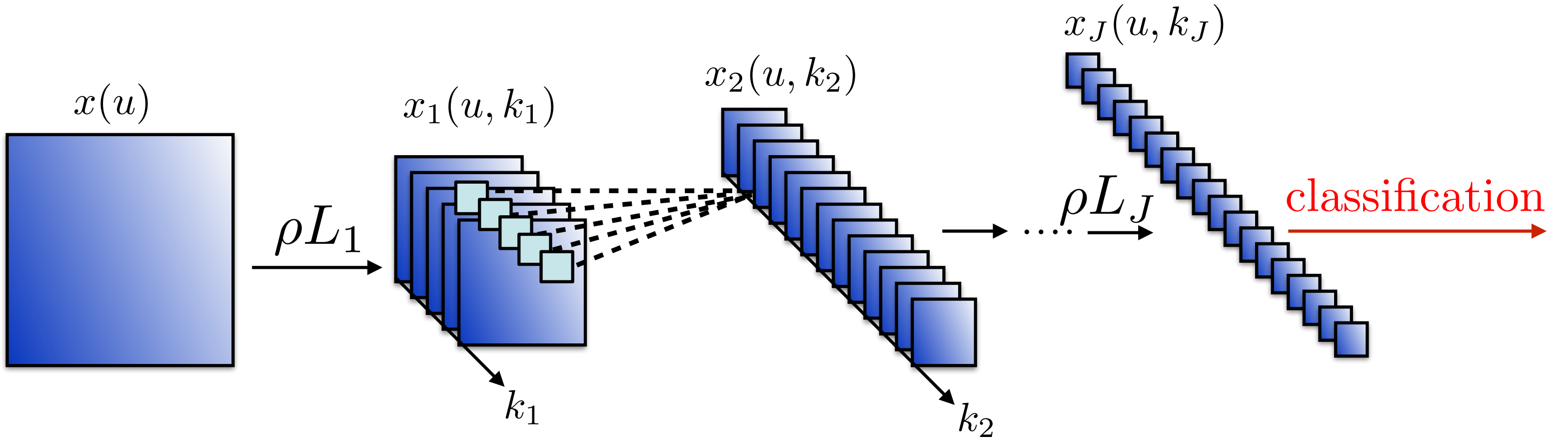
# Summary

- Contraction within class
- Separation between class
- After the zero-training-error (terminal phase of training),
  - Feature representation approaches the regular simplex of C vertices
  - Classifier converges to the nearest neighbor rule (LDA)

# Translation and Deformation Invariances in CNN

Stephane Mallat et al. Wavelet Scattering Networks

# Deep Convolutional Networks




$$x_j = \rho \, L_j \, x_{j-1}$$

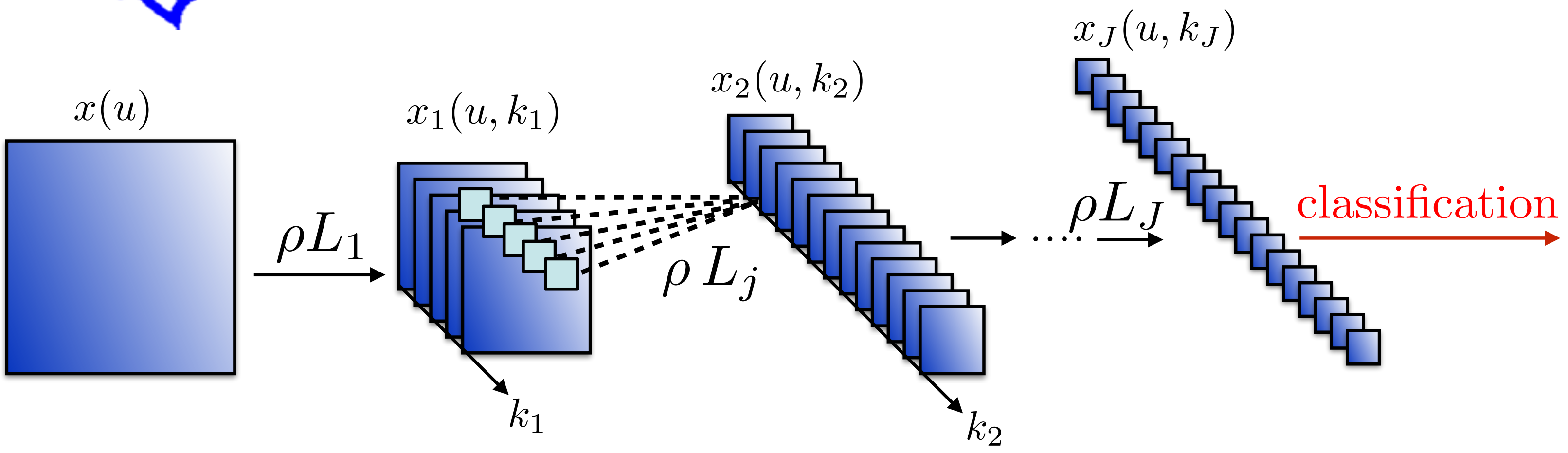
- $L_j$ is a linear combination of convolutions and subsampling:

$$x_j(u, k_j) = \rho\Big( \underbrace{\sum_k}_{\text{sum across channels}} x_{j-1}(\cdot, k) \star h_{k_j,k}(u) \Big)$$

- $\rho$ is contractive: $|\rho(u) - \rho(u')| \leq |u - u'|$

$$\rho(u) = \max(u, 0) \text{ or } \rho(u) = |u|$$

# Many Questions




- Why convolutions ? Translation covariance.
- Why no overfitting ? Contractions, dimension reduction

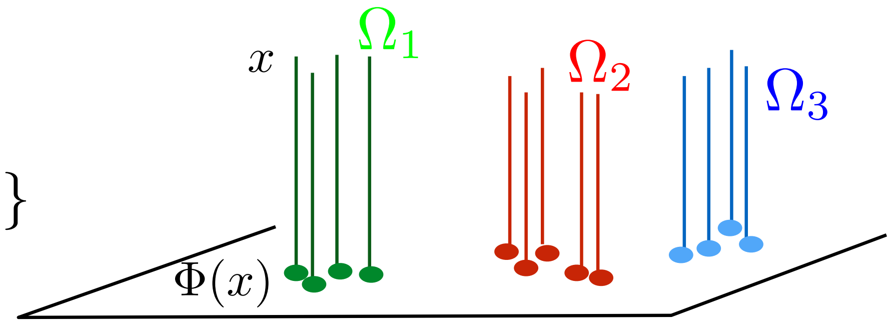
- Why hierarchical cascade ?
- Why introducing non-linearities ?
- How and what to linearise ?
- What are the roles of the multiple channels in each layer ?

# Linear Dimension Reduction

*Classes*

*Level sets of* $f(x)$

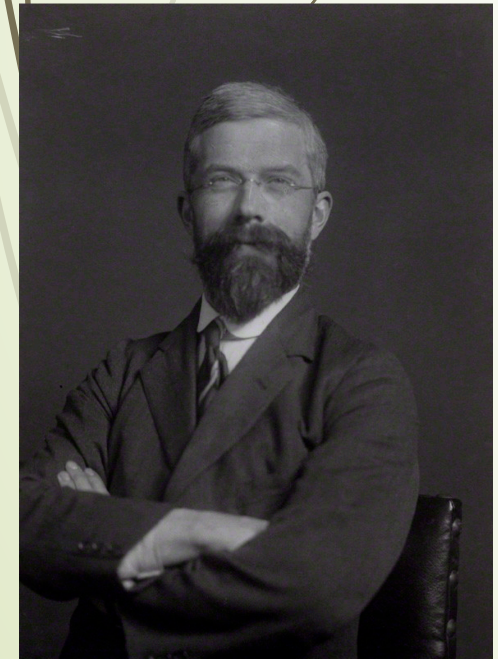
$$\Omega_t = \{x \;:\; f(x) = t\}$$




If level sets (classes) are parallel to a linear space then variables are eliminated by linear projections: *invariants*.

$$\Phi(x) = \alpha \hat{\Sigma}_W^{-1} (\hat{\mu}_1 - \hat{\mu}_0)$$

$$\hat{\mu}_k = \frac{1}{|C_k|} \sum_{i \in C_k} x_i \qquad \hat{\Sigma}_W = \sum_k \sum_{i \in C_k} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T$$

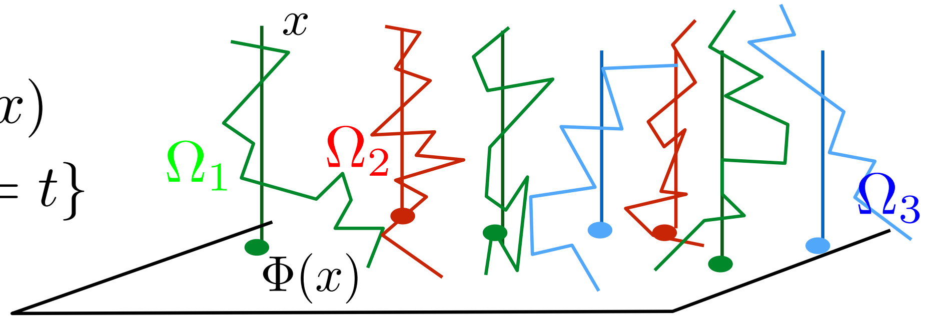


# Linearise for Dimensionality Reduction

*Classes*

*Level sets of* $f(x)$

$$\Omega_t = \{x \ : \ f(x) = t\}$$



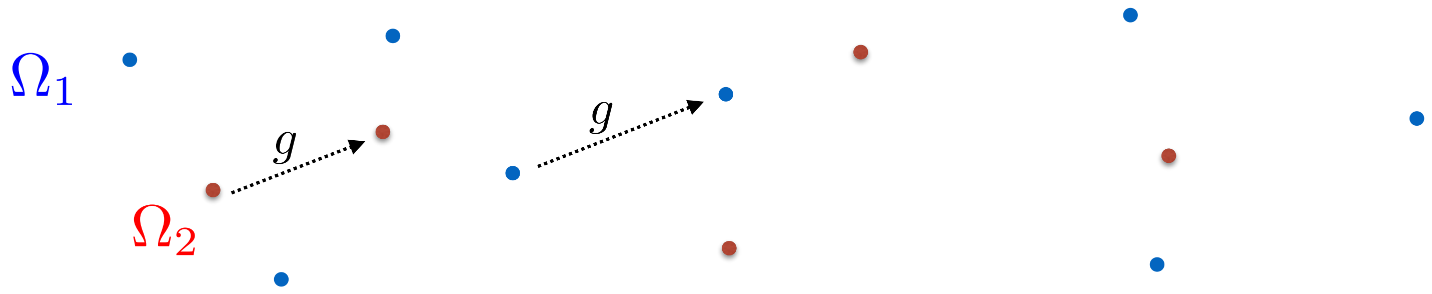

- If level sets $\Omega_t$ are not parallel to a linear space
  - Linearise them with a change of variable $\Phi(x)$
  - Then reduce dimension with linear projections

- Difficult because $\Omega_t$ are high-dimensional, irregular, known on few samples.

# Level Set Geometry: Symmetries

- Curse of dimensionality $\Rightarrow$ not local but global geometry

  Level sets: classes, characterised by their global symmetries.




- A symmetry is an operator $g$ which preserves level sets:

$$\forall x \quad , \quad f(g.x) = f(x) \ : \ \text{global}$$

If $g_1$ and $g_2$ are symmetries then $g_1.g_2$ is also a symmetry

$$f(g_1.g_2.x) = f(g_2.x) = f(x)$$

# Groups of symmetries

- $G = \{$ all symmetries $\}$ is a group: unknown

$$\forall (g, g') \in G^2 \quad \Rightarrow g.g' \in G$$

Inverse: $\quad \forall g \in G \;\; , \;\; g^{-1} \in G$

Associative: $\quad (g.g').g'' = g.(g'.g'')$

If commutative $\; g.g' = g'.g \;$ : Abelian group.

- Group of dimension $n$ if it has $n$ generators:

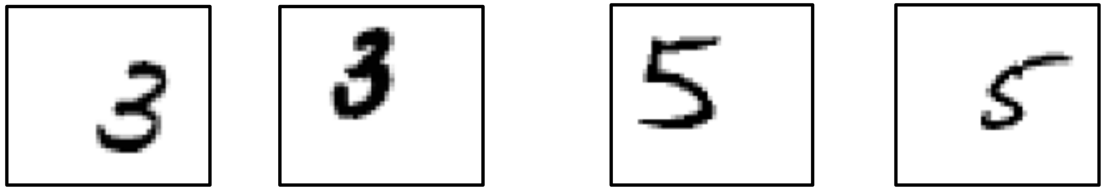
$$g = g_1^{p_1} \, g_2^{p_2} \ldots g_n^{p_n}$$

- Lie group: infinitely small generators (Lie Algebra)

# Translation and Deformations

- Digit classification:

$x(u)$ $x'(u)$



- Globally invariant to the translation group
- Locally invariant to small diffeomorphisms

Linearize small diffeomorphisms:
$\Rightarrow$ Lipschitz regular

*Video of Philipp Scott Johnson*

https://www.youtube.com/watch?v=nUDIoN-_Hxs

# Translations and Deformations

- Invariance to translations:

$$g.x(u) = x(u-c) \quad \Rightarrow \quad \Phi(g.x) = \Phi(x)\ .$$

- Small diffeomorphisms: $g.x(u) = x(u - \tau(u))$

Metric: $\|g\| = \|\nabla\tau\|_\infty$ maximum scaling

Linearisation by Lipschitz continuity

$$\|\Phi(x) - \Phi(g.x)\| \leq C\, \|\nabla\tau\|_\infty\ .$$

- Discriminative change of variable:

$$\|\Phi(x) - \Phi(x')\| \geq C^{-1}\, |f(x) - f(x')|$$

# Fourier Deformation Instability

- Fourier transform $\hat{x}(\omega) = \int x(t)\, e^{-i\omega t}\, dt$

$$x_c(t) = x(t-c) \quad \Rightarrow \quad \hat{x}_c(\omega) = e^{-ic\omega}\, \hat{x}(\omega)$$

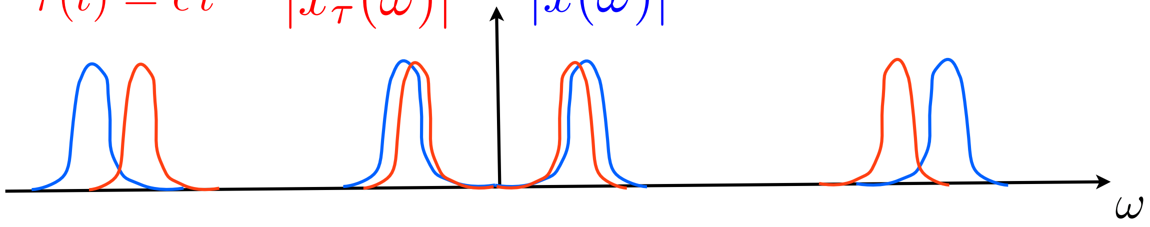
The modulus is invariant to translations:

$$\Phi(x) = |\hat{x}| = |\hat{x}_c|$$

- Instabilites to small deformations $x_\tau(t) = x(t - \tau(t))$ :

$\big|\, |\hat{x}_\tau(\omega)| - |\hat{x}(\omega)| \,\big|$ is big at high frequencies

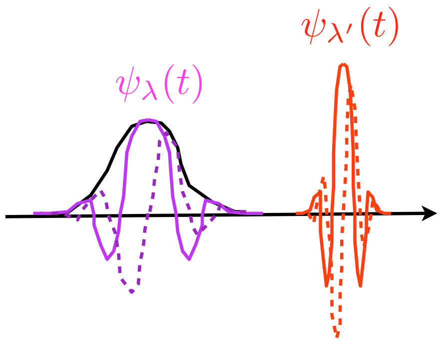
$\tau(t) = \epsilon\, t$ $\quad |\widehat{x}_\tau(\omega)|$ $\quad |\widehat{x}(\omega)|$

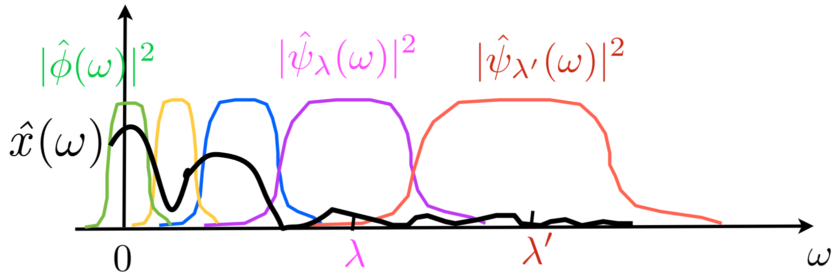



$$\Rightarrow \quad \big\| |\hat{x}| - |\hat{x}_\tau| \big\| \gg \|\nabla \tau\|_\infty \, \|x\|$$

# Wavelet Transform

- Complex wavelet: $\psi(t) = \psi^a(t) + i\,\psi^b(t)$
- Dilated: $\psi_\lambda(t) = 2^{-j}\,\psi(2^{-j}t)$ with $\lambda = 2^{-j}$ .




- Wavelet transform: $x \star \psi_\lambda(t) = \int x(u)\,\psi_\lambda(t-u)\,du$

$$Wx = \begin{pmatrix} x \star \phi(t) \\ x \star \psi_\lambda(t) \end{pmatrix}_{t,\lambda}$$

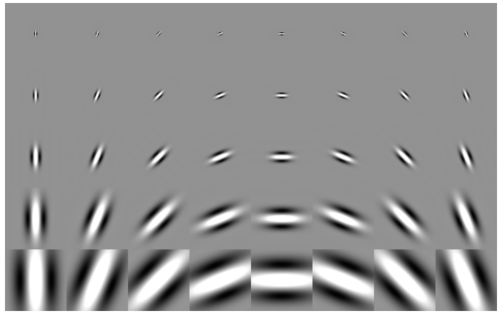
Unitary: $\|Wx\|^2 = \|x\|^2$ .

# Image Wavelet Transform

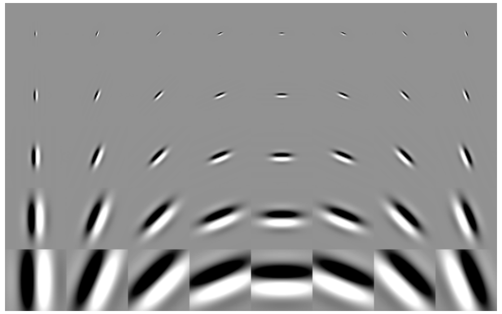
- Complex wavelet: $\psi(t) = \psi^a(t) + i\,\psi^b(t)$ , $t = (t_1, t_2)$

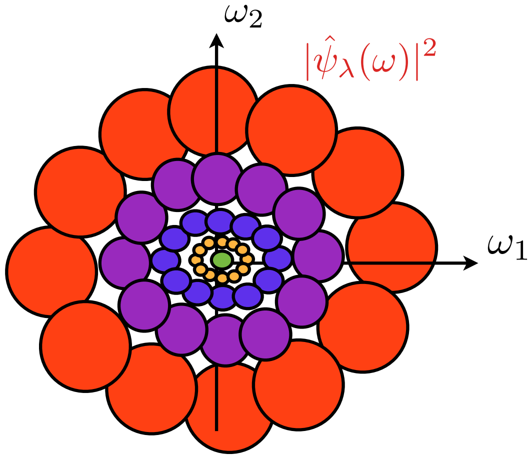
  rotated and dilated: $\psi_\lambda(t) = 2^{-j}\,\psi(2^{-j} r t)$ with $\lambda = (2^j, r)$

real parts



imaginary parts






- Wavelet transform: $$W x = \begin{pmatrix} x \star \phi(t) \\ x \star \psi_\lambda(t) \end{pmatrix}_{t,\lambda}$$

  Unitary: $\|Wx\|^2 = \|x\|^2$ .

# Why Wavelets?

- Complex band limited Wavelets are uniformly **stable to deformations**

$$\text{if } \psi_{\lambda,\tau}(t) = \psi_\lambda(t - \tau(t)) \text{ then}$$

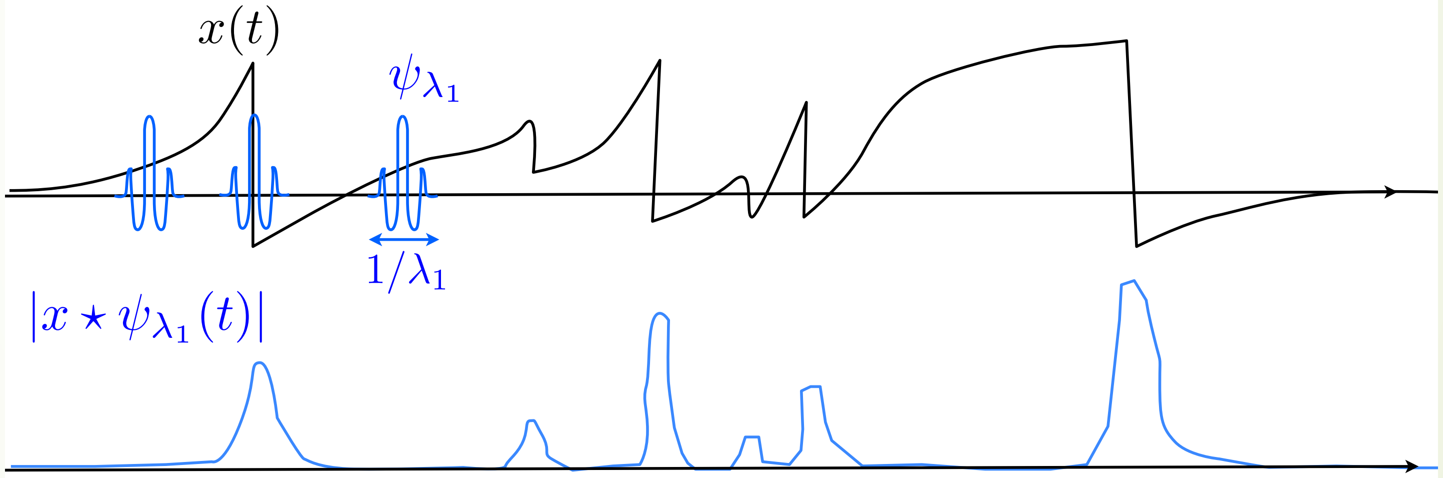
$$\|\psi_\lambda - \psi_{\lambda,\tau}\| \leq C \sup_t |\nabla \tau(t)| \ .$$

- Wavelets are **sparse** representations of functions
- Wavelets separate **multiscale** information
- Wavelets can be locally **translation invariant**

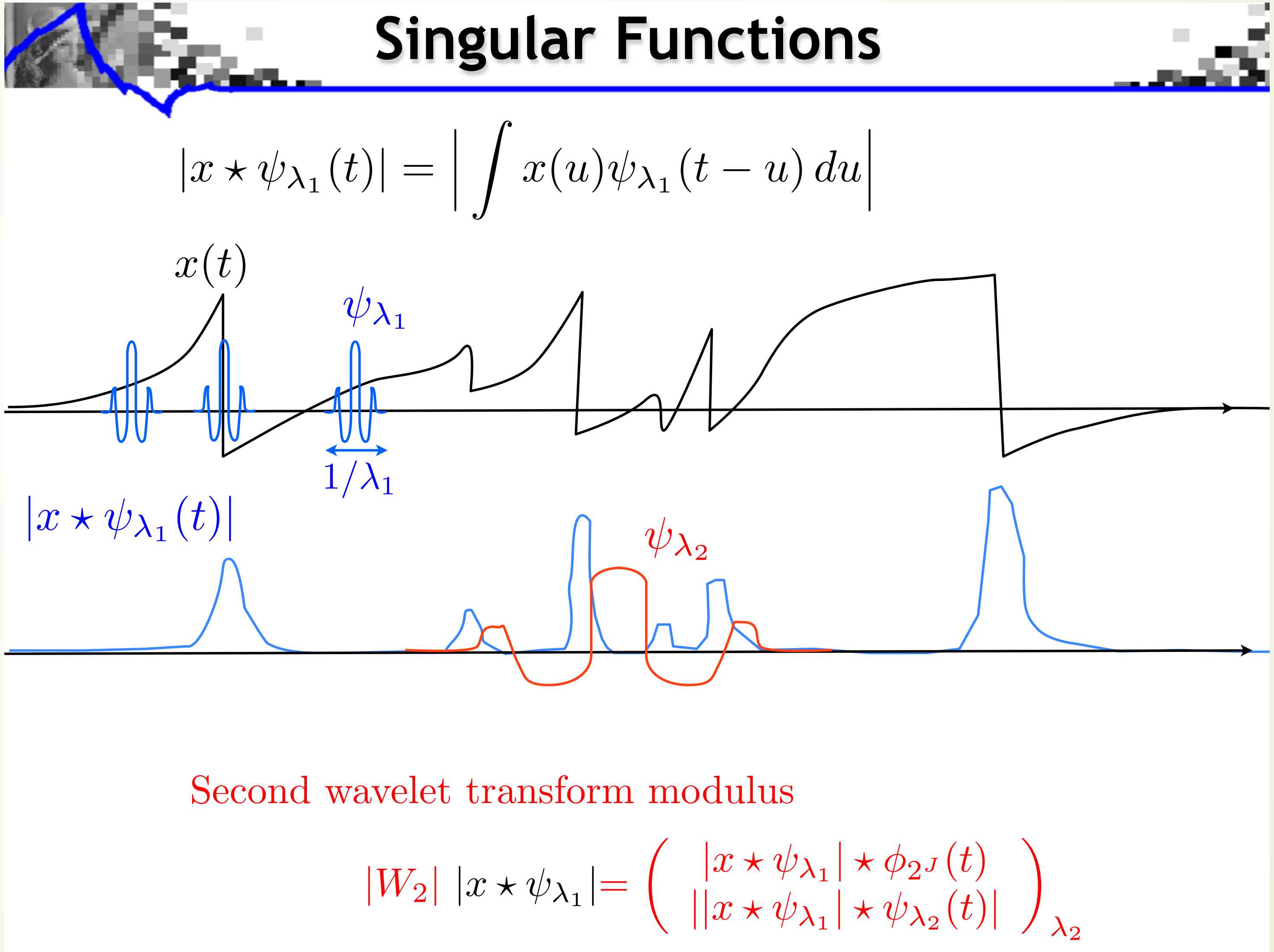
# Sparsity of Wavelet Transforms

## Singular Functions

$$|x \star \psi_{\lambda_1}(t)| = \left| \int x(u)\psi_{\lambda_1}(t-u)\, du \right|$$

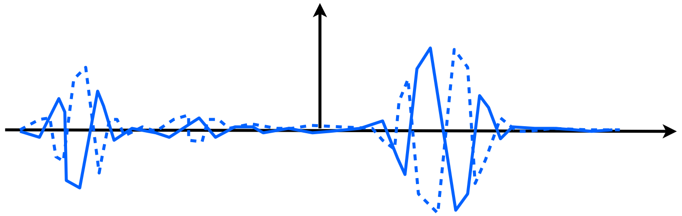


# Singularity is preserved in multiscale transform



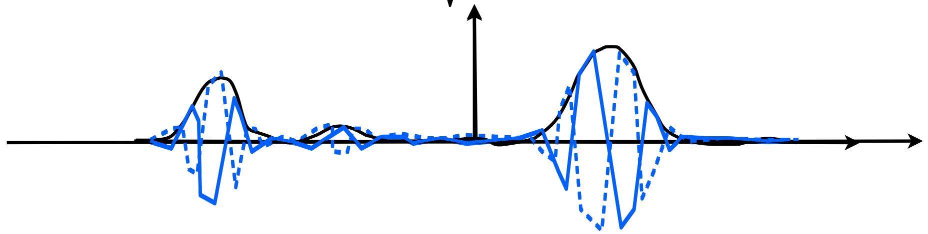
# Wavelet Translation Invariance

$$x \star \psi_{\lambda_1}(t) = x \star \psi^a_{\lambda_1}(t) + i\, x \star \psi^b_{\lambda_1}(t)$$



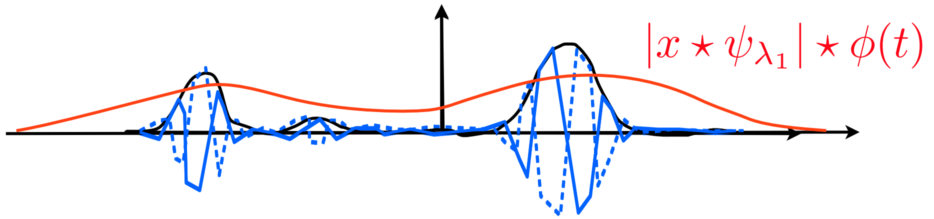
# Wavelet Translation Invariance

$$|x \star \psi_{\lambda_1}(t)| = \sqrt{|x \star \psi^a_{\lambda_1}(t)|^2 + |x \star \psi^b_{\lambda_1}(t)|^2} \quad \text{pooling}$$



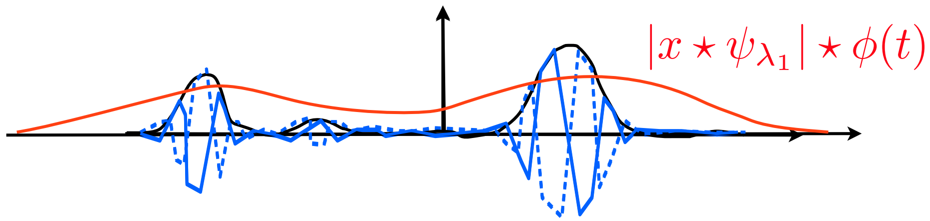
- The modulus $|x \star \psi_{\lambda_1}|$ is a regular envelop

# Wavelet Translation Invariance




- The modulus $|x \star \psi_{\lambda_1}|$ is a regular envelop
- The average $|x \star \psi_{\lambda_1}| \star \phi(t)$ is invariant to small translations relatively to the support of $\phi$.

# Wavelet Translation Invariance

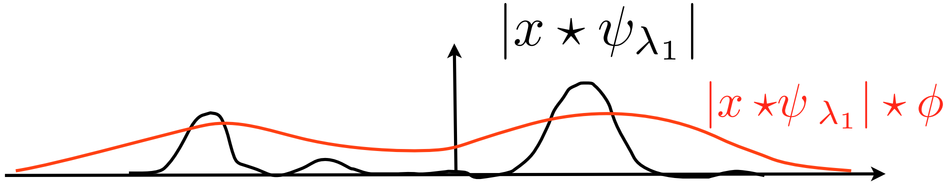



- The modulus $|x \star \psi_{\lambda_1}|$ is a regular envelop
- The average $|x \star \psi_{\lambda_1}| \star \phi(t)$ is invariant to small translations relatively to the support of $\phi$.
- Full translation invariance at the limit:

$$\lim_{\phi \to 1} |x \star \psi_{\lambda_1}| \star \phi(t) = \int |x \star \psi_{\lambda_1}(u)|\, du = \|x \star \psi_{\lambda_1}\|_1$$

but few invariants.

# Recovering Lost Information




- The high frequencies of $|x \star \psi_{\lambda_1}|$ are in wavelet coefficients:

$$W|x \star \psi_{\lambda_1}| = \begin{pmatrix} |x \star \psi_{\lambda_1}| \star \phi(t) \\ |x \star \psi_{\lambda_1}| \star \psi_{\lambda_2}(t) \end{pmatrix}_{t,\lambda_2}$$

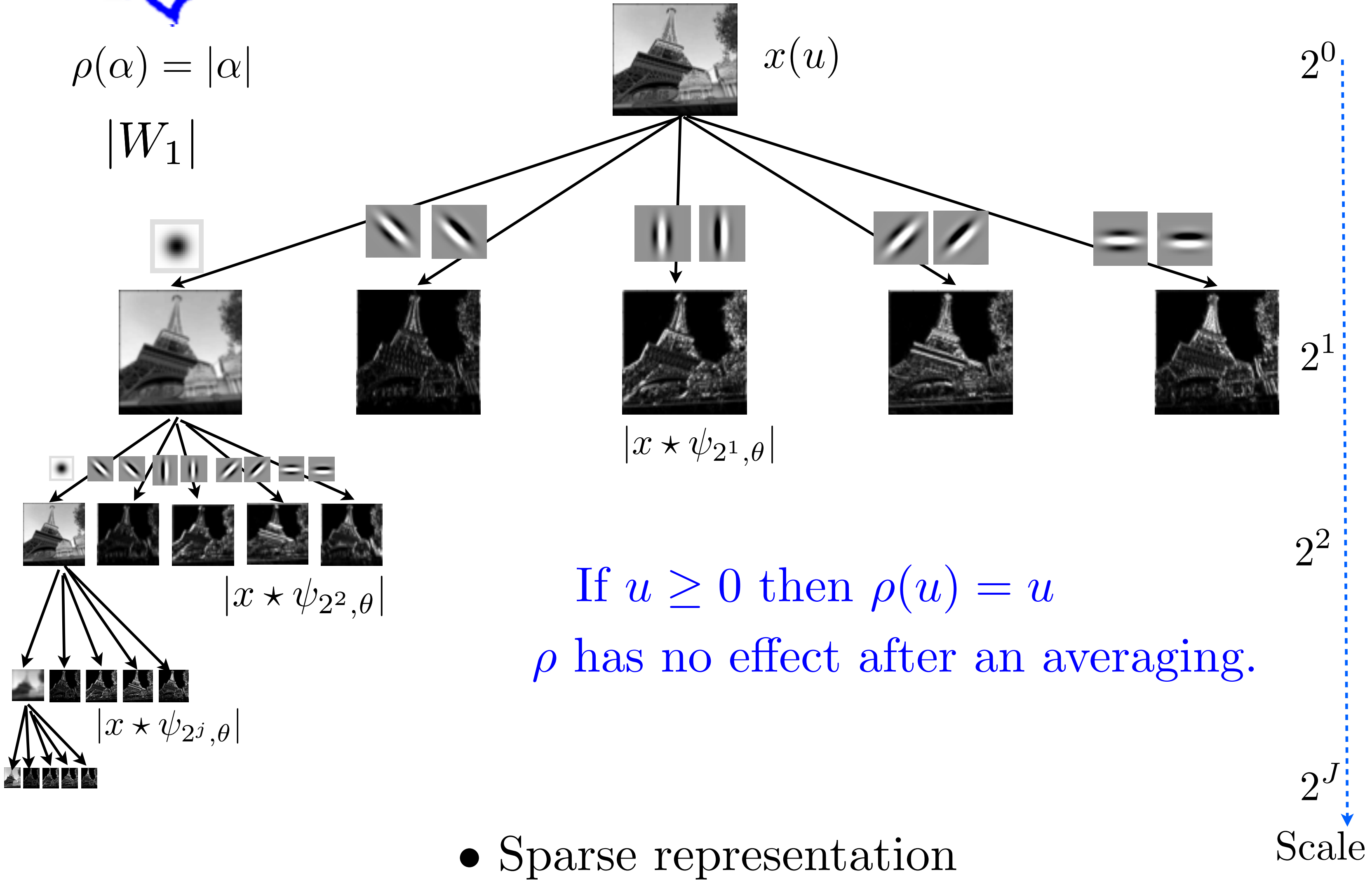
- Translation invariance by time averaging the amplitude:

$$\forall \lambda_1, \lambda_2, \quad ||x \star \psi_{\lambda_1}| \star \psi_{\lambda_2}| \star \phi(t)$$

# Wavelet Filter Bank

$\rho(\alpha) = |\alpha|$

$|W_1|$




If $u \geq 0$ then $\rho(u) = u$

$\rho$ has no effect after an averaging.

- Sparse representation

# Contraction

$$Wx = \begin{pmatrix} x \star \phi(t) \\ x \star \psi_\lambda(t) \end{pmatrix}_{t,\lambda}$$ is linear and $\|Wx\| = \|x\|$

$$\rho(u) = |u|$$

$$|W|x = \begin{pmatrix} x \star \phi(t) \\ |x \star \psi_\lambda(t)| \end{pmatrix}_{t,\lambda}$$ is non-linear

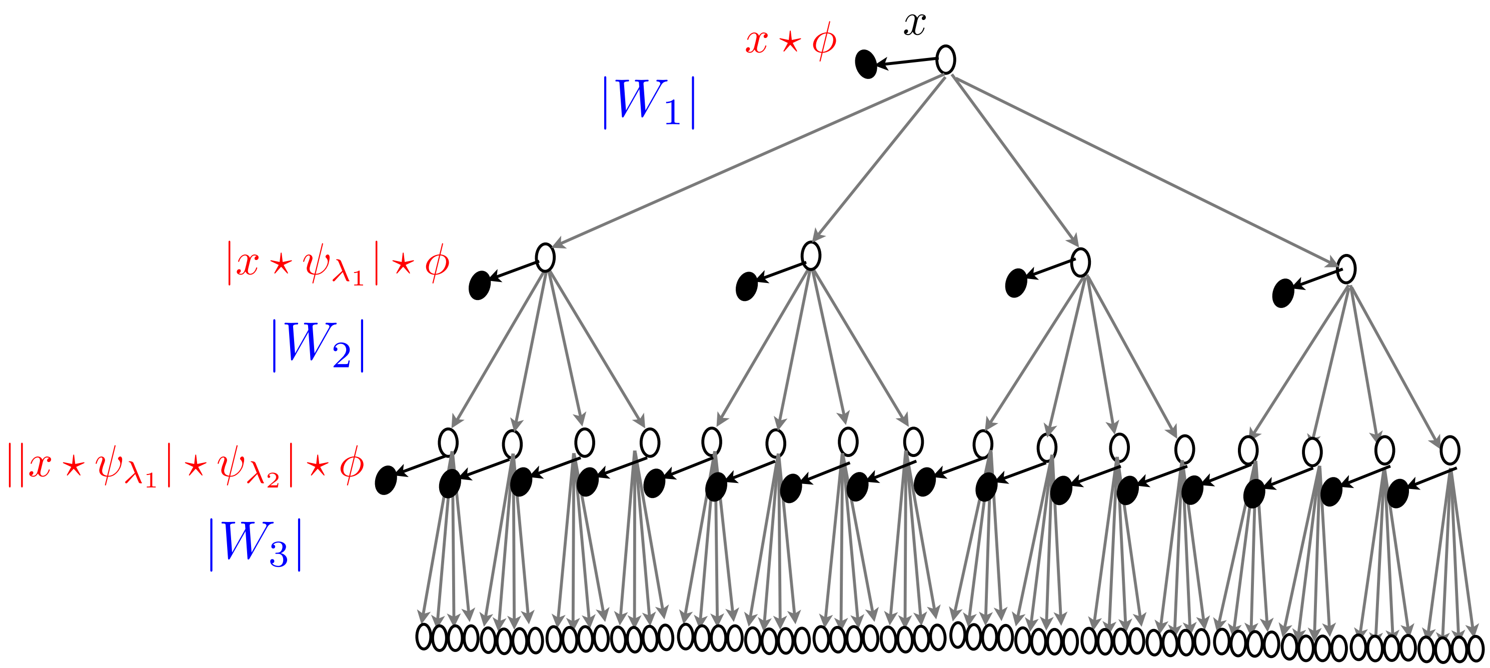
- it is contractive $\| |W|x - |W|y \| \leq \|x - y\|$

because for $(a, b) \in \mathbb{C}^2$ $||a| - |b|| \leq |a - b|$

- it preserves the norm $\| |W|x \| = \|x\|$

# Wavelet Scattering Network

## Cascade of Contractions




- Cascade of contractive operators

$$\| |W_k| x - |W_k| x' \| \leq \| x - x' \| \quad \text{with} \quad \| |W_k| x \| = \| x \| \ .$$

# Stability of Wavelet Scattering Transform

## Scattering Properties

$$Sx = \begin{pmatrix} x \star \phi\,(u) \\ |x \star \psi_{\lambda_1}| \star \phi(u) \\ ||x \star \psi_{\lambda_1}| \star \psi_{\lambda_2}| \star \phi(u) \\ |||x \star \psi_{\lambda_2}| \star \psi_{\lambda_2}| \star \psi_{\lambda_3}| \star \phi(u) \\ \cdots \end{pmatrix}_{u,\lambda_1,\lambda_2,\lambda_3,\ldots}$$

**Theorem**: *For appropriate wavelets, a scattering is*

*contractive* $\|Sx - Sy\| \leq \|x - y\|$

*preserves norms* $\|Sx\| = \|x\|$

*stable to deformations* $x_\tau(t) = x(t - \tau(t))$

$$\|Sx - Sx_\tau\| \leq C \sup_t |\nabla\tau(t)|\,\|x\|$$

$\Rightarrow$ linear discriminative classification from $\Phi x = Sx$

# Summary: Wavelet Scattering Net

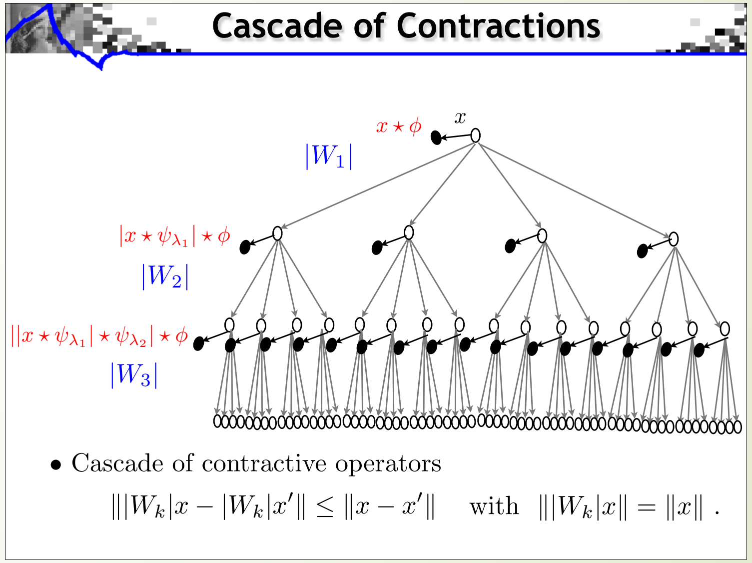
- Architechture:
  - Convolutional filters: band-limited wavelets
  - Nonlinear activation: modulus (Lipschitz)
  - Pooling: L1 norm as averaging
- Properties:
  - A Multiscale Sparse Representation
  - Norm Preservation (Parseval's identity):

$$\|Sx\| = \|x\|$$

  - Contraction:

$$\|Sx - Sy\| \leq \|x - y\|$$

$$Sx = \left( \begin{array}{c} x \star \phi\,(u) \\ |x \star \psi_{\lambda_1}| \star \phi(u) \\ ||x \star \psi_{\lambda_1}| \star \psi_{\lambda_2}| \star \phi(u) \\ |||x \star \psi_{\lambda_2}| \star \psi_{\lambda_2}| \star \psi_{\lambda_3}| \star \phi(u) \\ \cdots \end{array} \right)_{u,\lambda_1,\lambda_2,\lambda_3,\ldots}$$



## What is in between?

Scattering ⟵ - - - - - - - - - - ⟶ CNN

**Scattering**

- No training until the classifier
- No parameters in the convolutional layers
- Most “control” of regularity and robustness
- Strong performance and explainable features

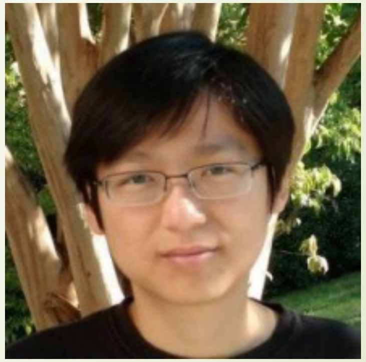
**CNN**

- Fully trained by large volume of data
- Lots of parameters (largest model capacity)
- Least “control” of regularity and robustness
- Best performance but not explainable

# Decomposed Convolutional Filters (DCF)

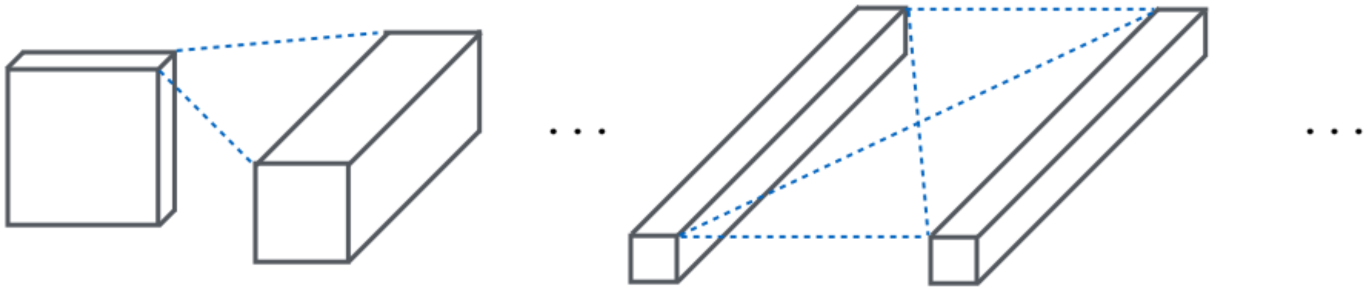
Xiuyuan Cheng et al.

https://arxiv.org/abs/1802.04145



# Decomposition of Convolutional Filters

$$x^{(0)} \mapsto x^{(1)} \mapsto \cdots \mapsto x^{(l-1)} \mapsto x^{(l)} \mapsto \cdots$$



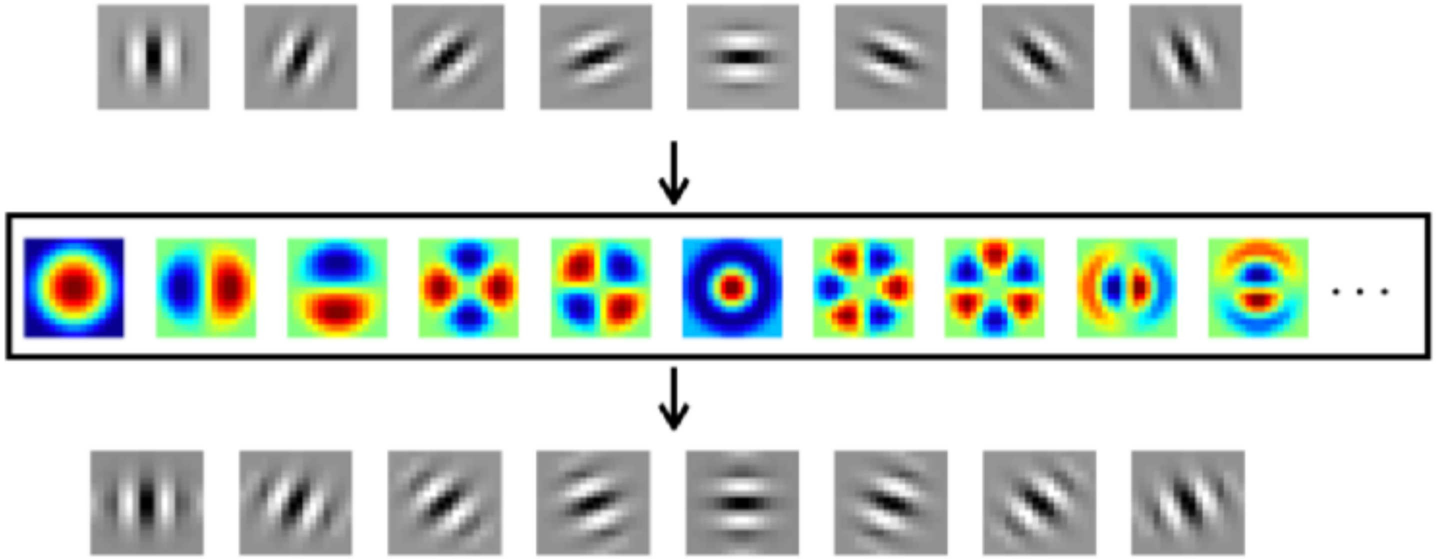

The mapping in a convolutional layer

$$x^{(l)}(u, \lambda) = \sigma\left(\sum_{\lambda'} \int W^{(l)}_{\lambda',\lambda}(v') x^{(l-1)}(u + v', \lambda') dv' + b^{(l)}(\lambda)\right)$$

# Decomposition of Convolutional Filters

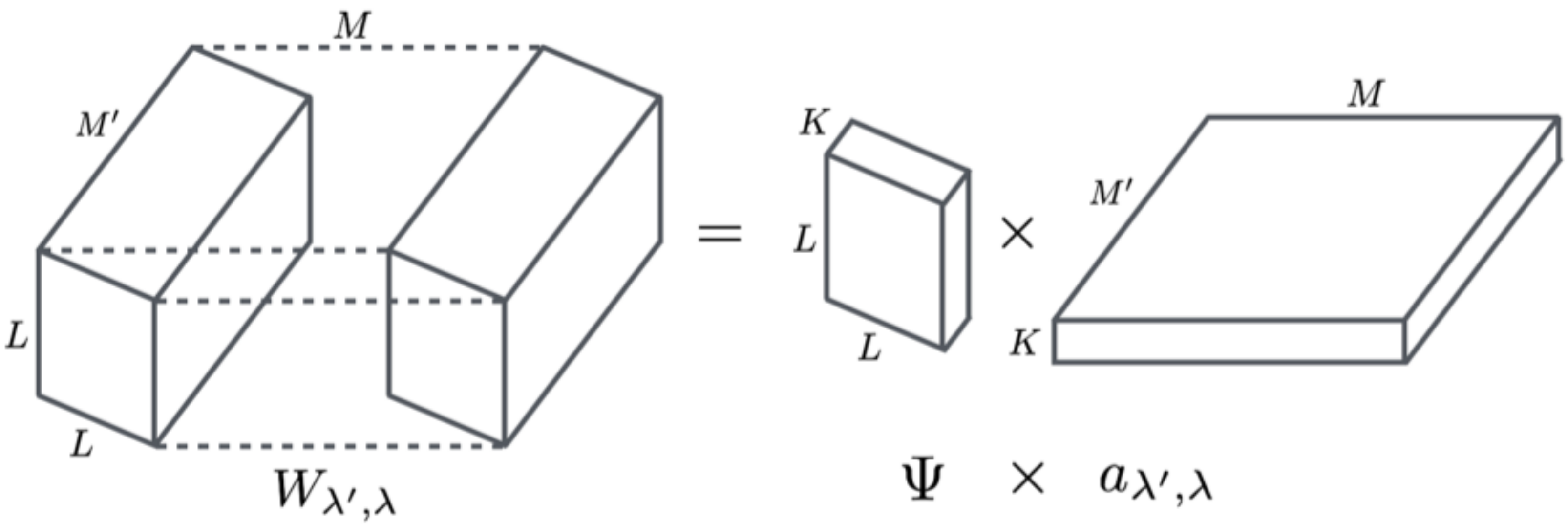
Introducing bases $\psi_k$

$$W_{\lambda',\lambda}(u) = \sum_{k=1}^{K} (a_{\lambda',\lambda})_k \psi_k(u).$$



# Decomposition of Convolutional Filters

- Filters viewed in tensors




$[L, L, M', M]$ $[L, L, 1, K]$ $[1, 1, KM', M]$

- Psi prefixed, a trained from data

# Reduction in the Number of Parameters

- Number of parameters
  - Regular conv layer: $L \times L \times M' \times M$
  - DCF layer: $K \times M' \times M$
- Forward-pass computation
  - Regular conv layer: $M'W^2 \cdot M(1 + 2L^2)$
  - DCF layer: $M'W^2 \cdot 2K(L^2 + M)$

*A factor of* $\frac{K}{L^2}$ *!*

# Applications and extensions:

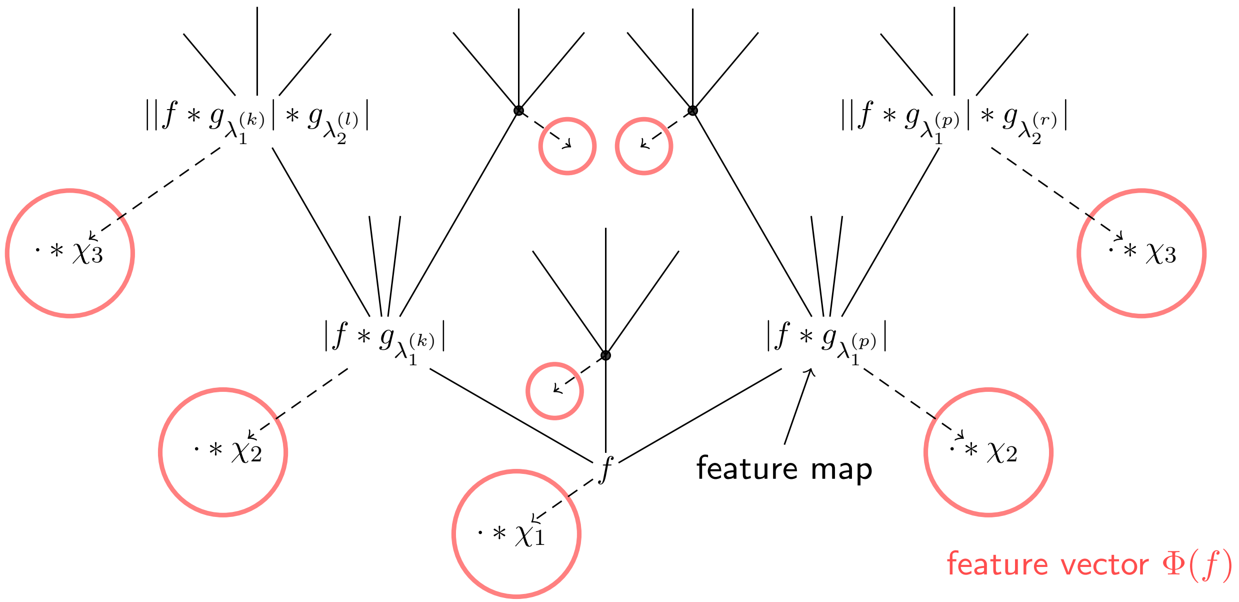
- Invertibility/completeness of representation [Waldspurger et al. ’12]
- Extension to signals on graphs [Chen et al. ’14] [Cheng et al. ’16]
- With general family of filters [Bolcskei et al. ’15] [Czaja et al. ’15]

# Wiatowski-Bolcskei'15

- Scattering Net by Mallat et al. so far
  - Wavelet Linear filter
  - Nonlinear activation by modulus
  - Average pooling
- Generalization by Wiatowski-Bolcskei'15
  - Filters as frames
  - Lipschitz continuous Nonlinearities
  - General Pooling: Max/Average/Nonlinear, etc.

# Generalization of Wiatowski-Bolcskei'15

**Scattering networks ([*Mallat, 2012*], [*Wiatowski and HB, 2015*])**




General scattering networks guarantee [*Wiatowski & HB, 2015*]

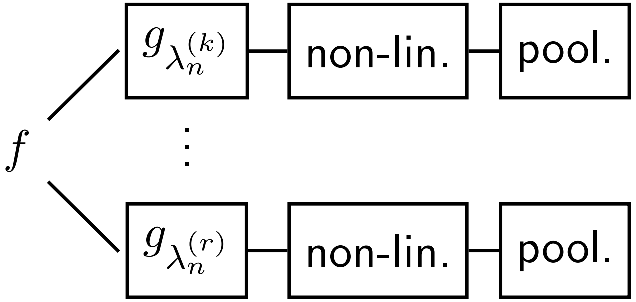
- (vertical) **translation invariance**
- **small deformation sensitivity**

essentially irrespective of filters, non-linearities, and poolings!

# Wavelet basis -> filter frame

## Building blocks

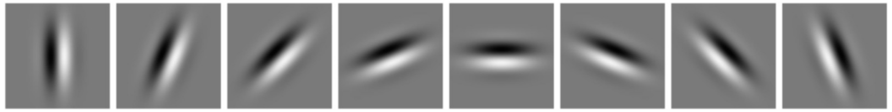
**Basic operations in the $n$-th network layer**




**Filters**: Semi-discrete frame $\Psi_n := \{\chi_n\} \cup \{g_{\lambda_n}\}_{\lambda_n \in \Lambda_n}$

$$A_n \|f\|_2^2 \leq \|f * \chi_n\|_2^2 + \sum_{\lambda_n \in \Lambda_n} \|f * g_{\lambda_n}\|^2 \leq B_n \|f\|_2^2, \quad \forall f \in L^2(\mathbb{R}^d)$$

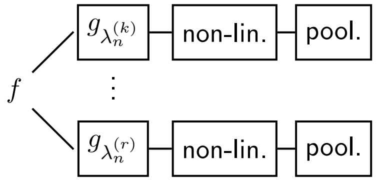
e.g.: Structured filters



# Frames: random or learned filters

## Building blocks

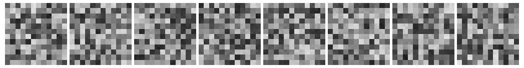
**Basic operations in the $n$-th network layer**




**Filters**: Semi-discrete frame $\Psi_n := \{\chi_n\} \cup \{g_{\lambda_n}\}_{\lambda_n \in \Lambda_n}$

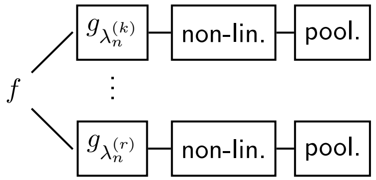
$$A_n \|f\|_2^2 \leq \|f * \chi_n\|_2^2 + \sum_{\lambda_n \in \Lambda_n} \|f * g_{\lambda_n}\|^2 \leq B_n \|f\|_2^2, \quad \forall f \in L^2(\mathbb{R}^d)$$

e.g.: Unstructured filters



## Building blocks

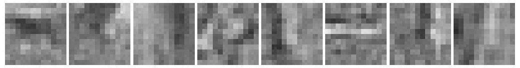
**Basic operations in the $n$-th network layer**




**Filters**: Semi-discrete frame $\Psi_n := \{\chi_n\} \cup \{g_{\lambda_n}\}_{\lambda_n \in \Lambda_n}$

$$A_n \|f\|_2^2 \leq \|f * \chi_n\|_2^2 + \sum_{\lambda_n \in \Lambda_n} \|f * g_{\lambda_n}\|^2 \leq B_n \|f\|_2^2, \quad \forall f \in L^2(\mathbb{R}^d)$$

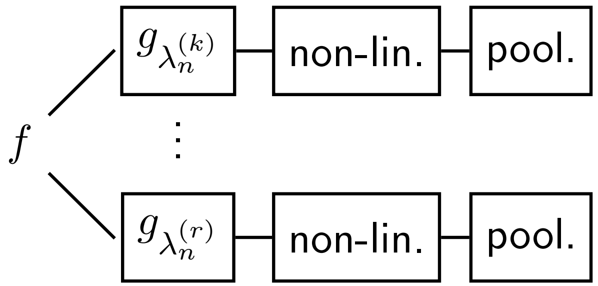
e.g.: Learned filters



# Nonlinear activations

## Building blocks

**Basic operations in the $n$-th network layer**




**Non-linearities**: Point-wise and Lipschitz-continuous

$$\|M_n(f) - M_n(h)\|_2 \leq L_n \|f - h\|_2, \quad \forall\, f, h \in L^2(\mathbb{R}^d)$$

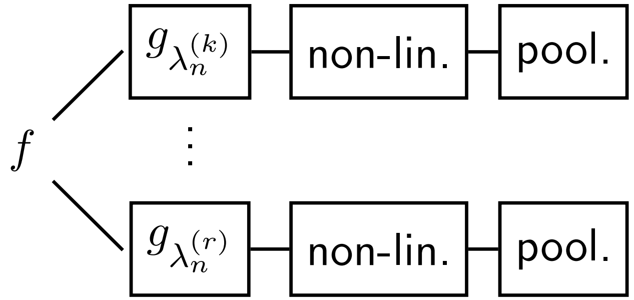
$\Rightarrow$ Satisfied by virtually **all** non-linearities used in the **deep learning literature**!

ReLU: $L_n = 1$; modulus: $L_n = 1$; logistic sigmoid: $L_n = \frac{1}{4}$; ...

# Pooling

## Building blocks

**Basic operations in the $n$-th network layer**




**Pooling**: In continuous-time according to

$$f \mapsto S_n^{d/2} P_n(f)(S_n \cdot),$$

where $S_n \geq 1$ is the **pooling factor** and $P_n : L^2(\mathbb{R}^d) \to L^2(\mathbb{R}^d)$ is $R_n$-Lipschitz-continuous

$\Rightarrow$ **Emulates** most **poolings** used in the **deep learning literature**!

e.g.: Pooling by **sub-sampling** $P_n(f) = f$ with $R_n = 1$

e.g.: Pooling by **averaging** $P_n(f) = f * \phi_n$ with $R_n = \|\phi_n\|_1$

# Vertical translation invariance

**Theorem (Wiatowski and HB, 2015)**

*Assume that the filters, non-linearities, and poolings satisfy*

$$B_n \leq \min\{1, L_n^{-2} R_n^{-2}\}, \quad \forall\, n \in \mathbb{N}.$$

*Let the pooling factors be $S_n \geq 1$, $n \in \mathbb{N}$. Then,*

$$|||\Phi^n(T_t f) - \Phi^n(f)||| = \mathcal{O}\left(\frac{\|t\|}{S_1 \ldots S_n}\right),$$

*for all $f \in L^2(\mathbb{R}^d)$, $t \in \mathbb{R}^d$, $n \in \mathbb{N}$.*

The condition

$$B_n \leq \min\{1, L_n^{-2} R_n^{-2}\}, \quad \forall\, n \in \mathbb{N},$$

is **easily satisfied** by **normalizing** the filters $\{g_{\lambda_n}\}_{\lambda_n \in \Lambda_n}$.

## Vertical translation invariance

**Theorem (Wiatowski and HB, 2015)**

*Assume that the filters, non-linearities, and poolings satisfy*

$$B_n \leq \min\{1, L_n^{-2} R_n^{-2}\}, \quad \forall\, n \in \mathbb{N}.$$

*Let the pooling factors be $S_n \geq 1$, $n \in \mathbb{N}$. Then,*

$$|||\Phi^n(T_t f) - \Phi^n(f)||| = \mathcal{O}\left(\frac{\|t\|}{S_1 \ldots S_n}\right),$$

*for all $f \in L^2(\mathbb{R}^d)$, $t \in \mathbb{R}^d$, $n \in \mathbb{N}$.*

$\Rightarrow$ Features become **more invariant** with **increasing** network **depth**!

# Vertical translation invariance

**Theorem (Wiatowski and HB, 2015)**

*Assume that the filters, non-linearities, and poolings satisfy*

$$B_n \leq \min\{1, L_n^{-2} R_n^{-2}\}, \quad \forall\, n \in \mathbb{N}.$$

*Let the pooling factors be $S_n \geq 1$, $n \in \mathbb{N}$. Then,*

$$|||\Phi^n(T_t f) - \Phi^n(f)||| = \mathcal{O}\left(\frac{\|t\|}{S_1 \ldots S_n}\right),$$

*for all $f \in L^2(\mathbb{R}^d)$, $t \in \mathbb{R}^d$, $n \in \mathbb{N}$.*

**Full translation invariance**: If $\lim_{n\to\infty} S_1 \cdot S_2 \cdot \ldots \cdot S_n = \infty$, then

$$\lim_{n\to\infty} |||\Phi^n(T_t f) - \Phi^n(f)||| = 0$$

## Philosophy behind invariance results

Mallat's "horizontal" translation invariance [*Mallat, 2012*]:

$$\lim_{J\to\infty} |||\Phi_W(T_t f) - \Phi_W(f)||| = 0, \quad \forall f \in L^2(\mathbb{R}^d),\ \forall t \in \mathbb{R}^d$$

- features become invariant in every network layer, but needs $J \to \infty$
- applies to wavelet transform and modulus non-linearity without pooling

"Vertical" translation invariance:

$$\lim_{n\to\infty} |||\Phi^n(T_t f) - \Phi^n(f)||| = 0, \quad \forall f \in L^2(\mathbb{R}^d), \forall t \in \mathbb{R}^d$$

- features become more invariant with increasing network depth
- applies to general filters, general non-linearities, and general poolings

# Group Invariant and Equivariant Networks

Cohen, Welling, https://arxiv.org/abs/1602.07576

Sannai, Takai, Cordonnier, https://arxiv.org/abs/1903.01939v2

**Definition 2.1.** Let $G$ be a group and $X$ and $Y$ two sets. We assume that $G$ acts on $X$ (resp. $Y$) by $g \cdot x$ (resp. $g * y$) for $g \in G$ and $x \in X$ (resp. $y \in Y$) . We say that a map $f\colon X \to Y$ is

- *$G$-invariant* if $f(g \cdot x) = f(x)$ for any $g \in G$ and any $x \in X$,
- *$G$-equivariant* if $f(g \cdot x) = g * f(x)$ for any $g \in G$ and any $x \in X$.

# Group Convolution Neural Network

[Cohen, Welling, https://arxiv.org/abs/1602.07576]

$$[f * \psi^i](x) = \sum_{y \in \mathbb{Z}^2} \sum_{k=1}^{K^l} f_k(y) \psi_k^i(x - y)$$

$$[f \star \psi](g) = \sum_{h \in G} \sum_{k} f_k(h) \psi_k(g^{-1}h).$$

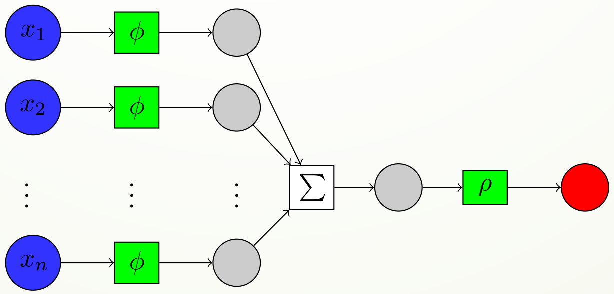
# Permutation Invariant Functions

When $G = S_n$ and the actions are induced by permutation, we call $G$-invariant (resp. $G$-equivariant) functions as *permutation invariant* (resp. *permutation equivariant*) functions.

**Theorem 3.1** ([28] Kolmogorov-Arnold's representation theorem for permutation actions)**.** *Let $K \subset \mathbb{R}^n$ be a compact set. Then, any continuous $S_n$-invariant function $f\colon K \longmapsto \mathbb{R}$ can be represented as*

$$f(x_1, \ldots, x_n) = \rho\left(\sum_{i=1}^{n} \phi(x_i)\right) \tag{1}$$

*for some continuous function $\rho\colon \mathbb{R}^{n+1} \to \mathbb{R}$. Here, $\phi\colon \mathbb{R} \to \mathbb{R}^{n+1}; x \mapsto (1, x, x^2, \ldots, x^n)^\top$.*



# Permutation Equivariant Functions

**Proposition 4.1.** *A map $F\colon \mathbb{R}^n \to \mathbb{R}^n$ is $S_n$-equivariant if and only if there is a* $\mathrm{Stab}(1)$*-invariant function $f\colon \mathbb{R}^n \to \mathbb{R}$ satisfying $F = (f, f \circ (1\ 2), \ldots, f \circ (1\ n))^\top$. Here, $(1\ i) \in S_n$ is the transposition between $1$ and $i$.*

**Corollary 4.1** (Representation of $\mathrm{Stab}(1)$-invariant function)**.** *Let $K \subset \mathbb{R}^n$ be a compact set, let $f\colon K \longrightarrow \mathbb{R}$ be a continuous and* $\mathrm{Stab}(1)$*-invariant function. Then, $f(\boldsymbol{x})$ can be represented as*

$$f(\boldsymbol{x}) = f(x_1, \ldots, x_n) = \rho\left(x_1, \sum_{i=2}^{n} \phi(x_i)\right),$$

*for some continuous function $\rho\colon \mathbb{R}^{n+1} \longrightarrow \mathbb{R}$. Here, $\phi\colon \mathbb{R} \to \mathbb{R}^n$ is similar as in Theorem 3.1.*

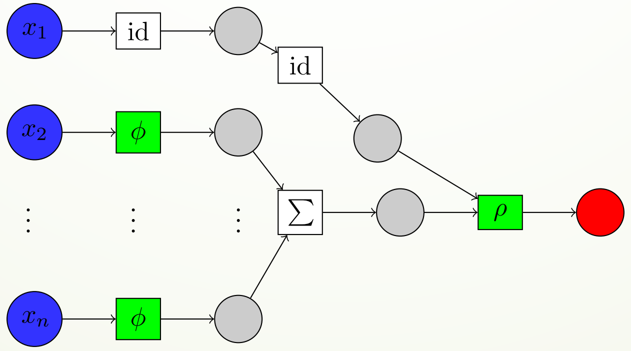



Diagram 3: A neural network approximating the $\mathrm{Stab}(1)$-invariant function $f$

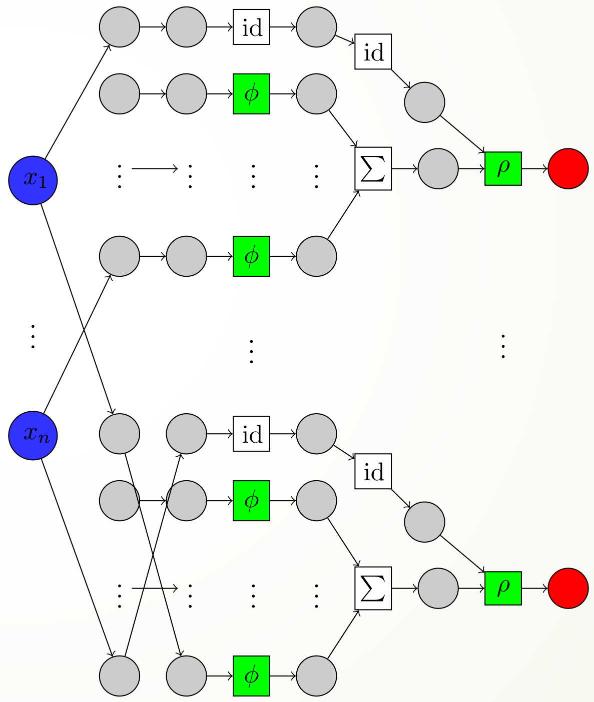



Diagram 2: A neural network approximating $S_n$-equivariant map $F$

# Thank you!