From Feedforward-Designed Convolutional Neural Networks (FF-CNNs) to Successive Subspace Learning (SSL)

January 30, 2020

C.-C. Jay Kuo
University of Southern California

Introduction

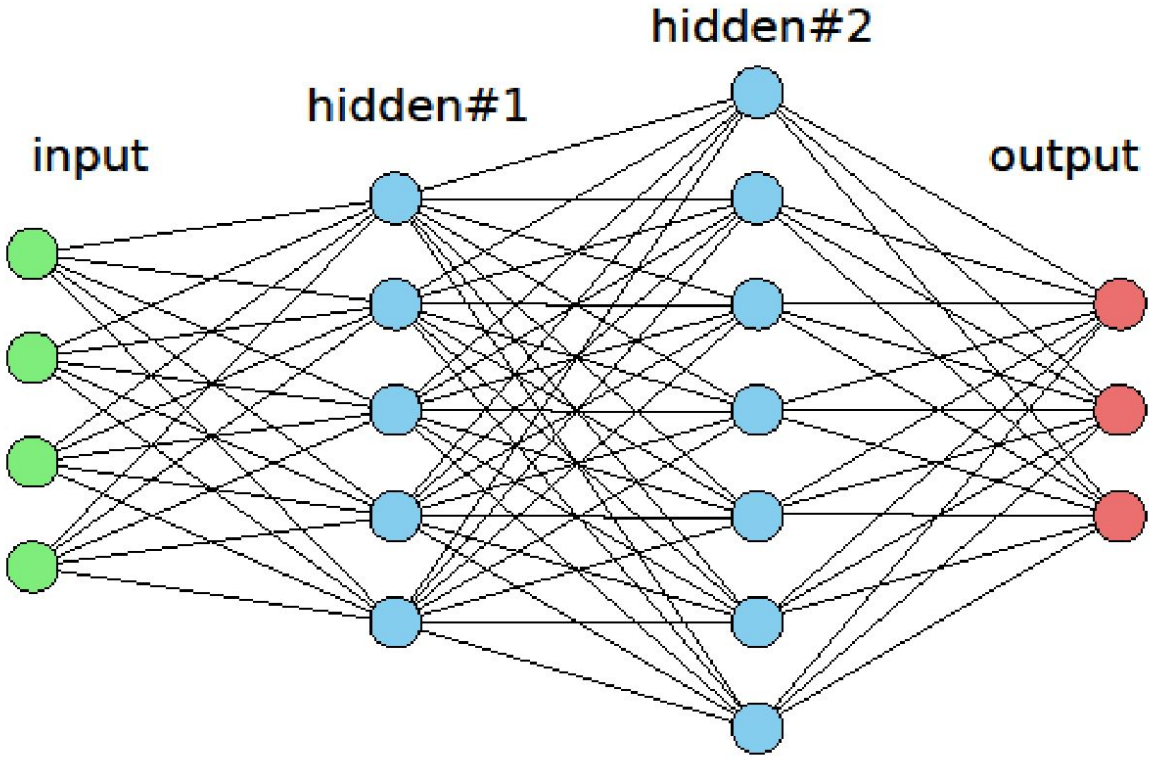
- **Deep Learning provides an effective solution when training data is rich, yet**
  - Lack of interpretability
  - Lack of reliability
    - Vulnerability to adversarial attacks
  - Training complexity
- **An effort towards explainable machine learning**

Evolution of CNNs

- Computational neuron and logic networks
  - McCulloch and Pitts (1943)
  - Why nonlinear activation?
- Multi-Layer Perceptron (MLP)
  - Rosenblatt (1957)
  - Used as “decision networks”
  - Why works?
- Convolutional Neural Networks (CNN)
  - Fukushima (1980) and LeCun et al. (1998)
  - AlexNet (2012)
  - Used as “ feature extraction & decision networks”
  - Why works?

Multilayer Perceptron (MLP)

- Full connection between every two adjacent layers
- No connection between neurons at the same layer
- Highly parallelism
- Supervised learning by backpropagation (BP)



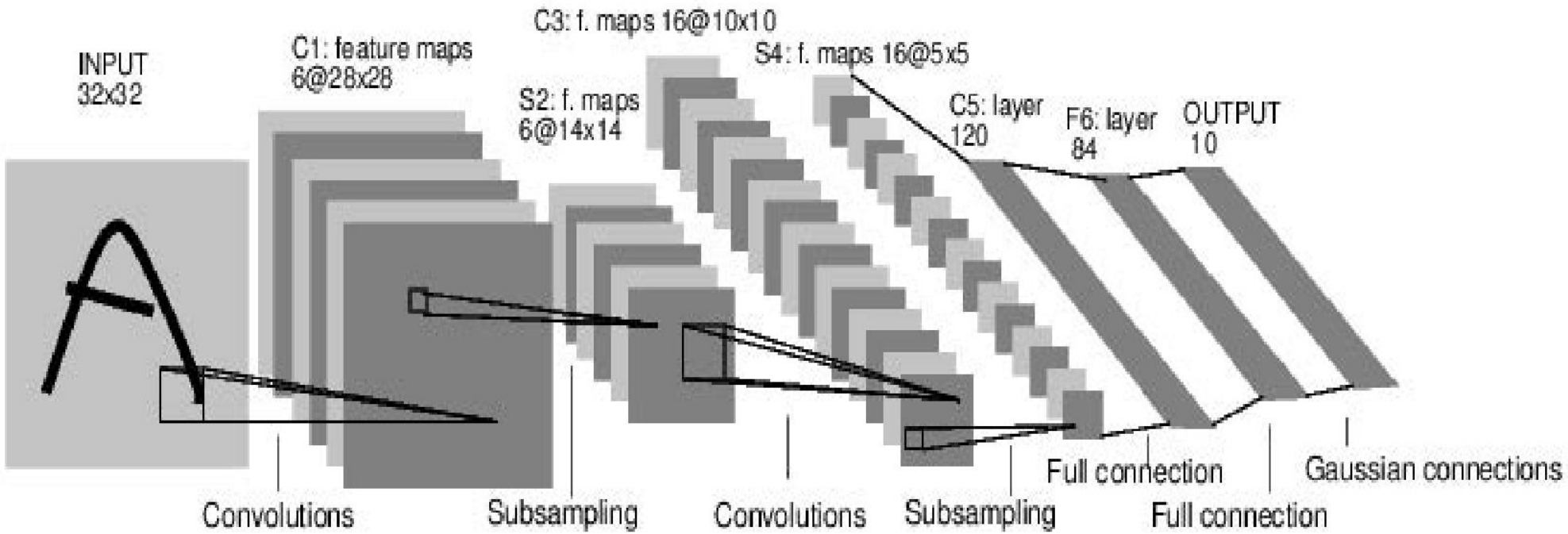
Classic 2-Hidden Layer MLP

Competitions and Limitations

- **MLPs were hot in 80's and early 90's**
  - Use the n-D feature vector as the input
  - One feature per input node (n nodes in total)
- **Competitive solutions exist**
  - SVM
  - Random Forest
- **What happens if the input is the source data?**
  (e.g. an image of size 32x32 = 1024)

Convolutional Neural Network (CNN)

- LeNet-5



- Can handle a large image by partitioning it into small blocks
- Convolutional layers -> feature extraction module
- Fully connected layers -> decision module
- Two modules are back-to-back connected

CNN Design via Backpropagation (BP)

- **Three human design choices**
  - CNN architecture (hyper-parameters)
  - Cost function at the output
  - Training dataset (input data and output label)
- **Network parameters are determined by the end-to-end optimization algorithm -> backpropagation (BP)**
  - Non-convex optimization
  - Few theoretical results
    - Universal approximation (one-hidden layer)
    - Local minima are as good as global minimum

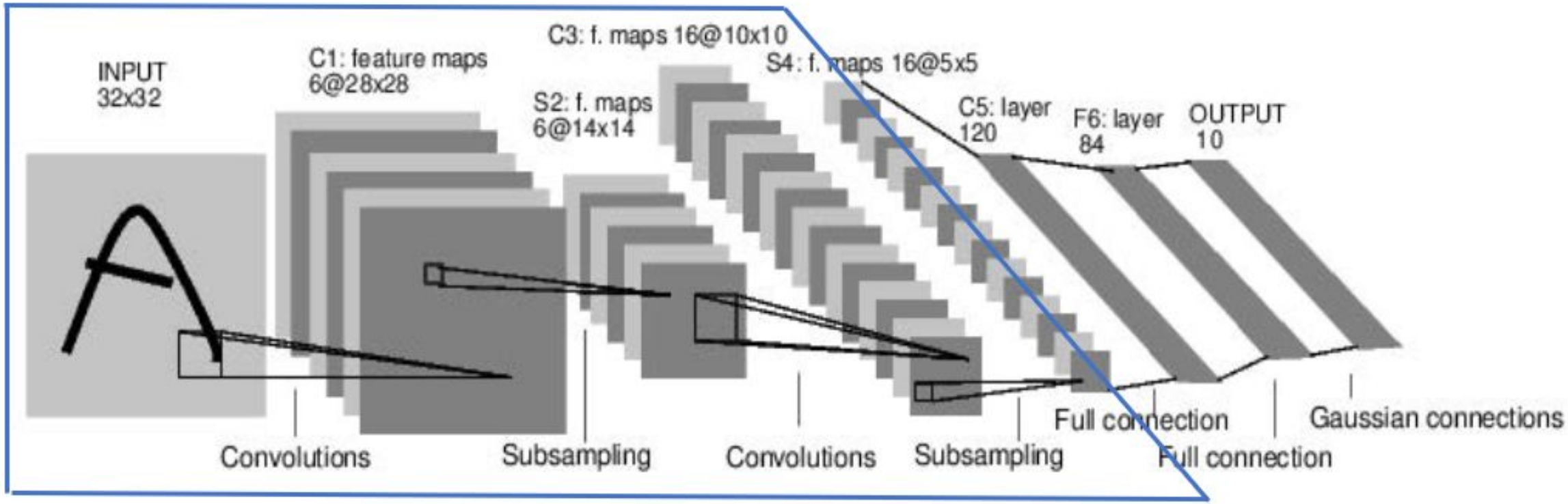
Feedforward-Designed Convolutional Neural Networks (FF-CNNs)

Feedforward (FF) Design

- Given a CNN architecture, how to design model parameters in a feedforward manner?
- New viewpoint:
  - Vectors in high-dimensional spaces
    - Example: Classification of CIFAR-10 color images of spatial size 32x32 to 10 classes
      - Input space of dimension 32x32x3=3,072
      - Output space of dimension 10
      - Intermediate layers: vector spaces of various dimensions
    - A unified framework of image representations, features and

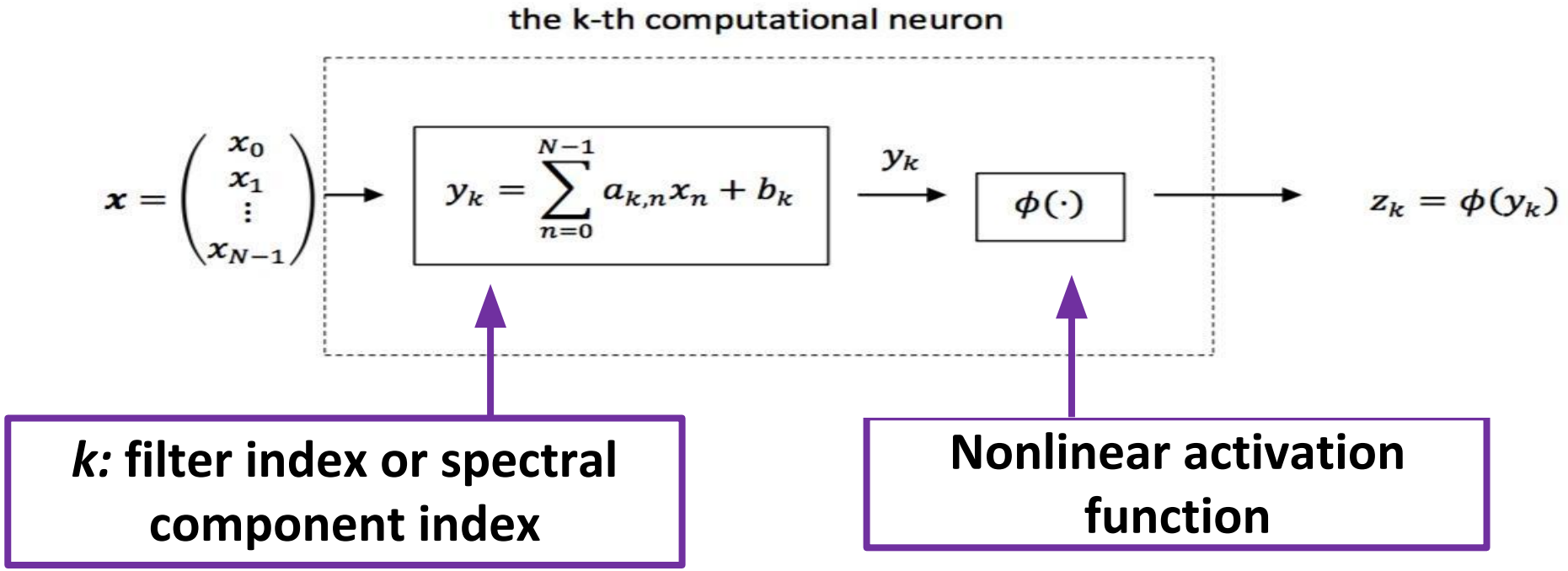
Selecting Parameters in Conv Layers

Exemplary network: LeNet-5



2 convolutional layers + 2 FC layers + 1 output layer

Convolutional Filter



Two challenges:

- Nonlinear activation – difficult to analyze
- A multi-stage affine system is complex

Three Ideas in Parameters Selection

1st viewpoint (training process, BP)

- Parameters to optimize in large nonlinear networks
- Backpropagation – SGD

2nd viewpoint (testing process)

- Filter weights are fixed (called anchor vectors)
- Inner product of input and filter weights -> matched filters
- k-means clustering

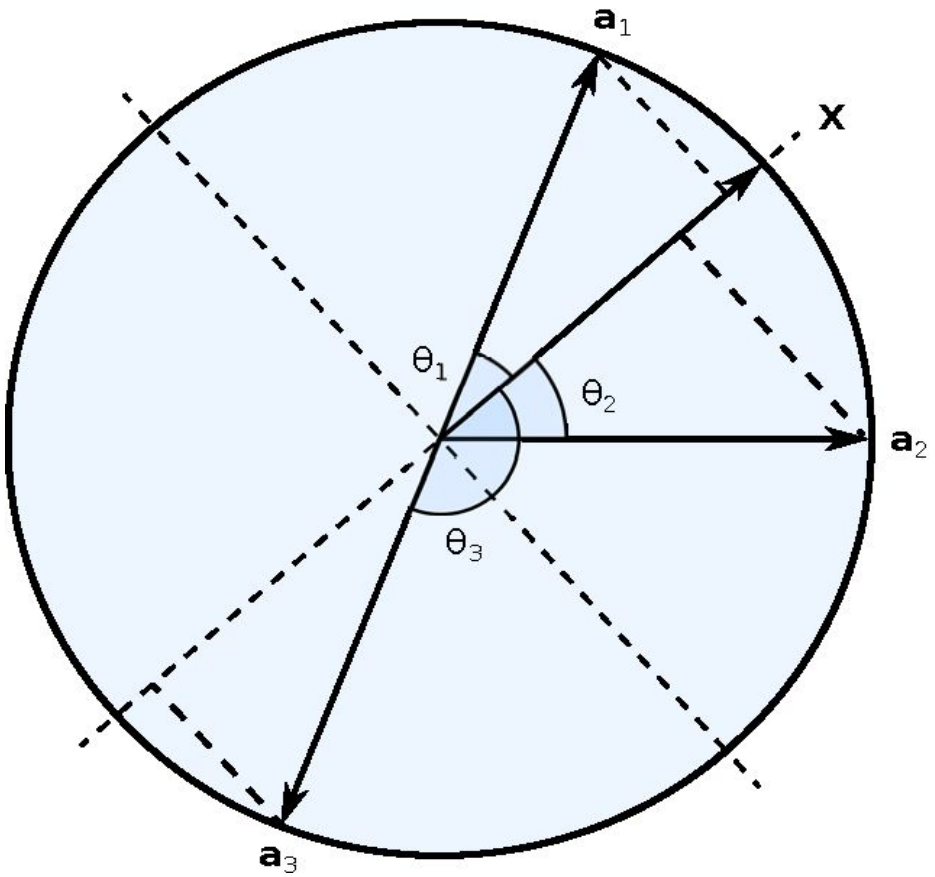
3rd viewpoint (testing process, FF)

- Bases (or kernels) for a linear space
- Subspace approximation

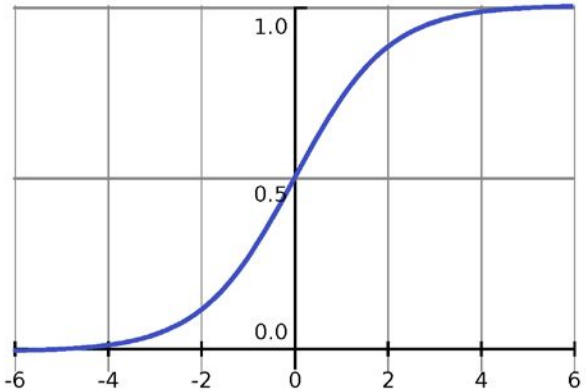
3rd Viewpoint: Subspace Approximation

Assumption: $b_k = 0$ for all k

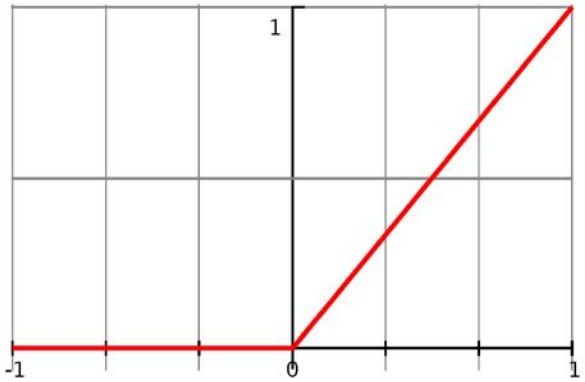
- $\boldsymbol{a}_k$ is an anchor vector of unit length in a hypersphere
- $\boldsymbol{x}$ is an input vector of arbitrary length
- $y_k = < \boldsymbol{a}_k , \boldsymbol{x} >$
- The space spanned by $\boldsymbol{a}_k$, $1 \leq k \leq K$, is a subspace of the input space $\boldsymbol{x} \in \boldsymbol{R}^N$ (if $K < N$)
- What happens to negative correlation?



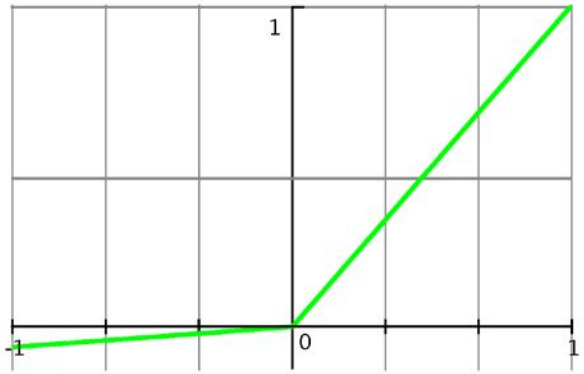
Nonlinear Activation (1)



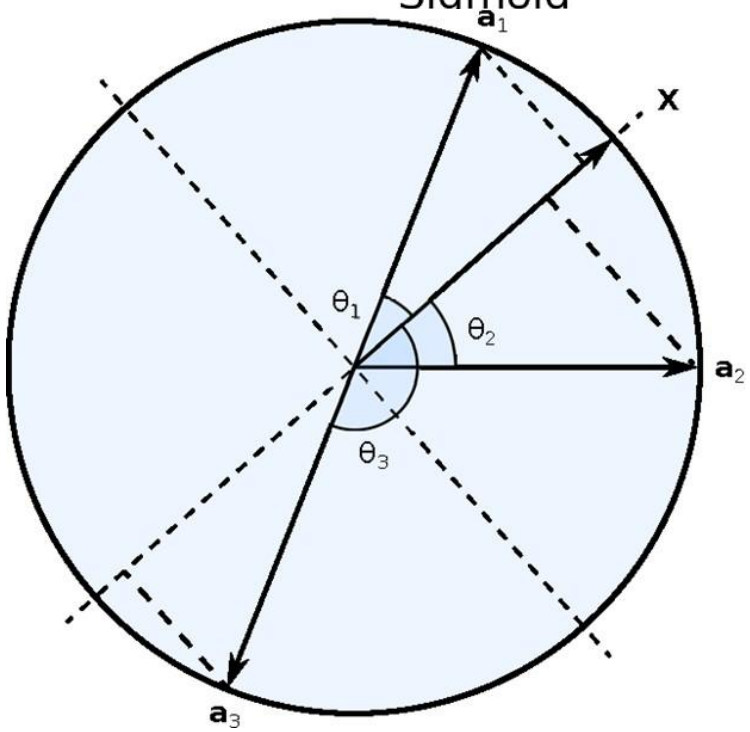
Sigmoid

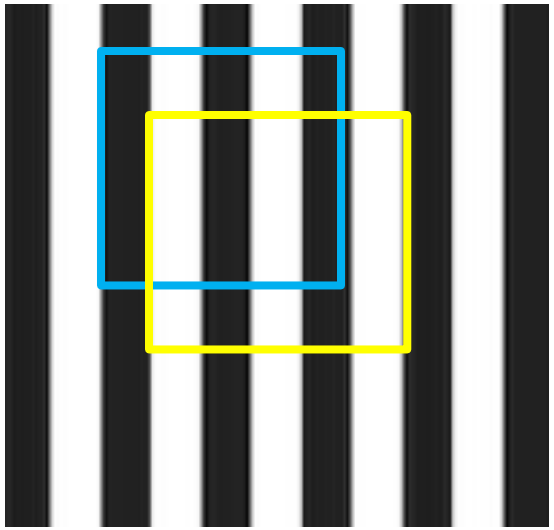


ReLU



Leaky ReLU





a : Original 5 by 5 vertical stripe pattern

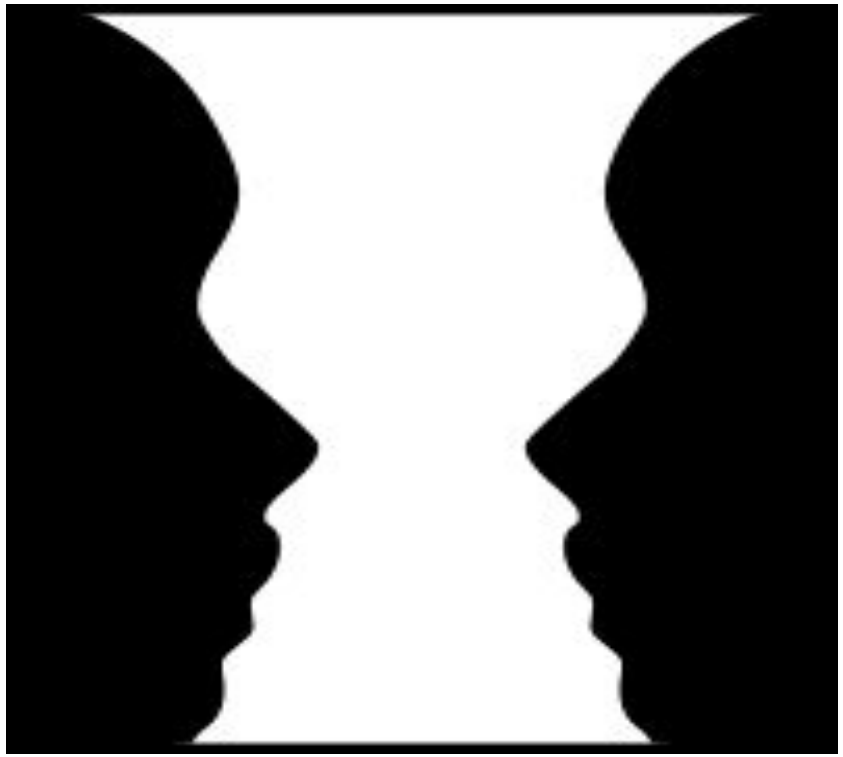
$-a$: Negative 5 by 5 vertical stripe pattern

Nonlinear Activation (2)

- **The sign confusion problem**
  - When two convolutional filters are in cascade, the system is not able to differentiate the following scenarios:
  - **Confusing Case #1**
    a. A positive correlation followed by a positive outgoing filter weight
    b. A negative correlation followed by a negative outgoing filter weight
  - **Confusing Case #2**
    a. A positive correlation followed by a negative outgoing filter weight
    b. A negative correlation followed by a positive outgoing filter weight
- **Solution**
  - Nonlinear activation provides a constraint to block case (b) in above - a rectifier

C.-C. Jay Kuo, "Understanding Convolutional Neural Networks with A Mathematical Model", the Journal of Visual Communication and Image Representation, Vol. 41, pp. 406-413, 2016

Rubin Vase Illusion



Inverse RECOS Transform

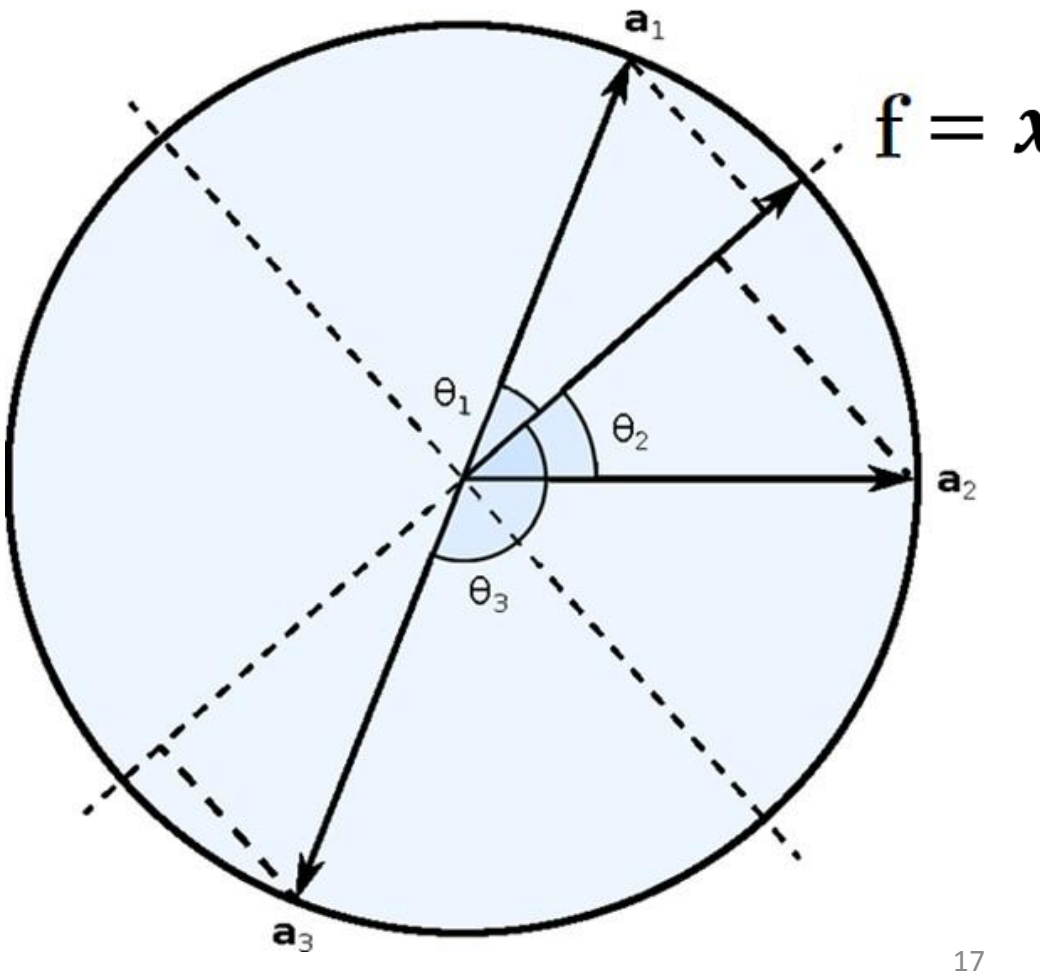
How to reconstruct input $\boldsymbol{f}$ from its projected values?

Unrectified responses:

$$p_k = \mathbf{a}_k^T \mathbf{f}, \quad k = 0, 1, \cdots K.$$

Rectified responses:

$$g_k = \begin{cases} p_k, & \text{if} \quad p_k > 0, \\ 0, & \text{if} \quad p_k \leqslant 0. \end{cases}$$



Subspace Approximation

If the no. of anchor vectors is less than the dimension of input **f** , there is an approximation error

$$\mathbf{f} \approx \hat{\mathbf{f}} = \sum_{k=0}^{K} \alpha_k \mathbf{a}_k.$$

Filter weights as spanning vectors for a linear space

$$p_k \approx \mathbf{a}_k^T \hat{\mathbf{f}} = \mathbf{a}_k^T \left(\sum_{k'=0}^{K} \alpha_{k'} \mathbf{a}_{k'} \right)$$

Approximation Loss

- Controlled by the no. of anchor filters
- Find optimal anchor filters
  - Truncated Karhunen Loeve Transform (or PCA)
  - Orthogonal eigenvectors

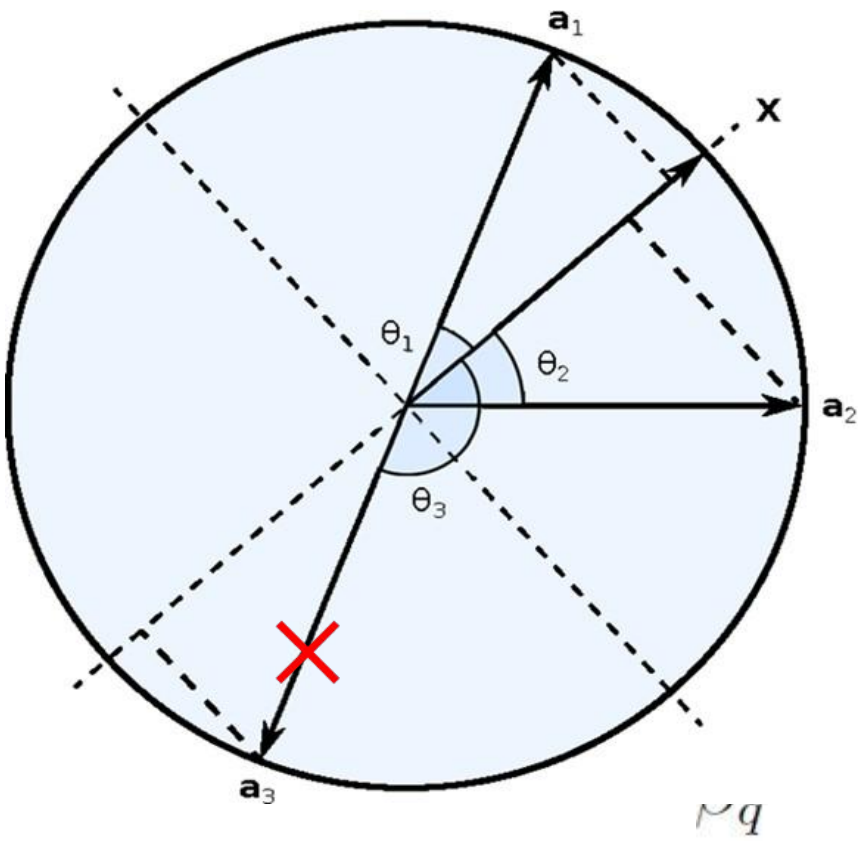
$$\mathbf{a}_i^T \mathbf{a}_j = < \mathbf{a}_i, \mathbf{a}_j > = \delta_{i,j}$$

  - Easy to invert

$$\hat{\mathbf{f}} = \sum_{k=0}^{K} p_k \mathbf{a}_k,$$

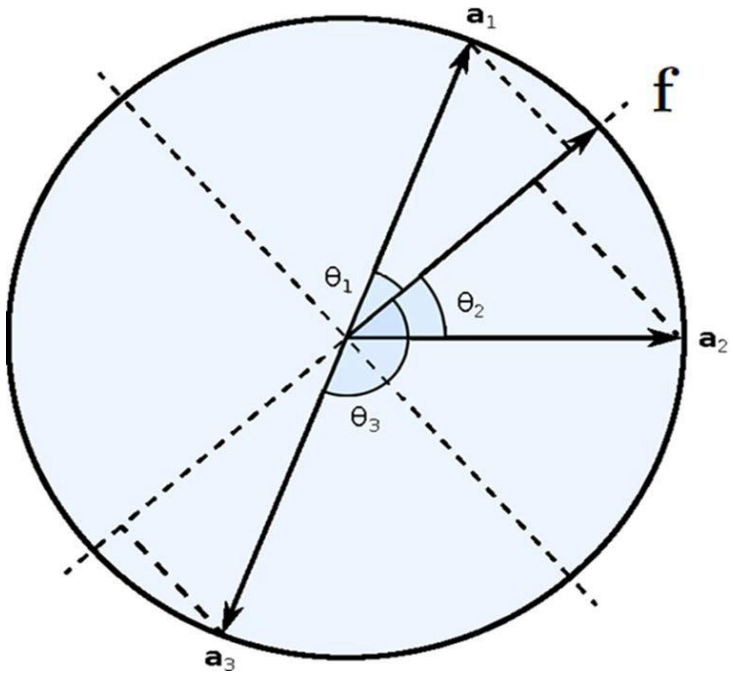
Rectification Loss

- **Due to Nonlinear Activation**
  - Needed to resolve the sign confusion problem



Recovering Rectification Loss – Saak Transform

- Augment anchor vectors by their negatives
- **Subspace approximation with augmented kernels (Saak) transform**



- Both $\boldsymbol{a}_k$ and $-\boldsymbol{a}_k$ are selected as anchor vectors
- Cost: double the output dimension

C.-C. Jay Kuo and Yueru Chen, "On data-driven Saak transform," the Journal of Visual Communications and Image Representation, Vol. 50, pp. 237-246, January 2018

Recovering Rectification Loss – Saab Transform

- **Subspace approximation with adjusted bias (Saab)**
  - Subspace decomposition

$$\mathcal{S} = \mathcal{S}_{DC} \oplus \mathcal{S}_{AC}$$

  - DC subspace is spanned by constant-element vector $d\,(1, \ldots, 1)$
  - AC subspace is its orthogonal complement

- **Relaxing the assumption ---** b_k =0 **for all** k
  - The bias vector resides in the DC subspace
  - The PCA analysis holds for the AC subspace

Bias Terms Selection (1)

- **Two requirements**

  (B1) Nonlinear activation automatically holds

$$y_k = \sum_{n=0}^{N-1} a_{k,n} x_n + b_k = \mathbf{a}_k^T \mathbf{x} + b_k \geqslant 0$$

$$\Longrightarrow \quad z_k = \phi(y_k) = \max(0, y_k) = y_k$$

  (B2) All bias terms are equal

$$b_0 = b_1 = \cdots = b_{K-1}$$

Bias Terms Selection (2)

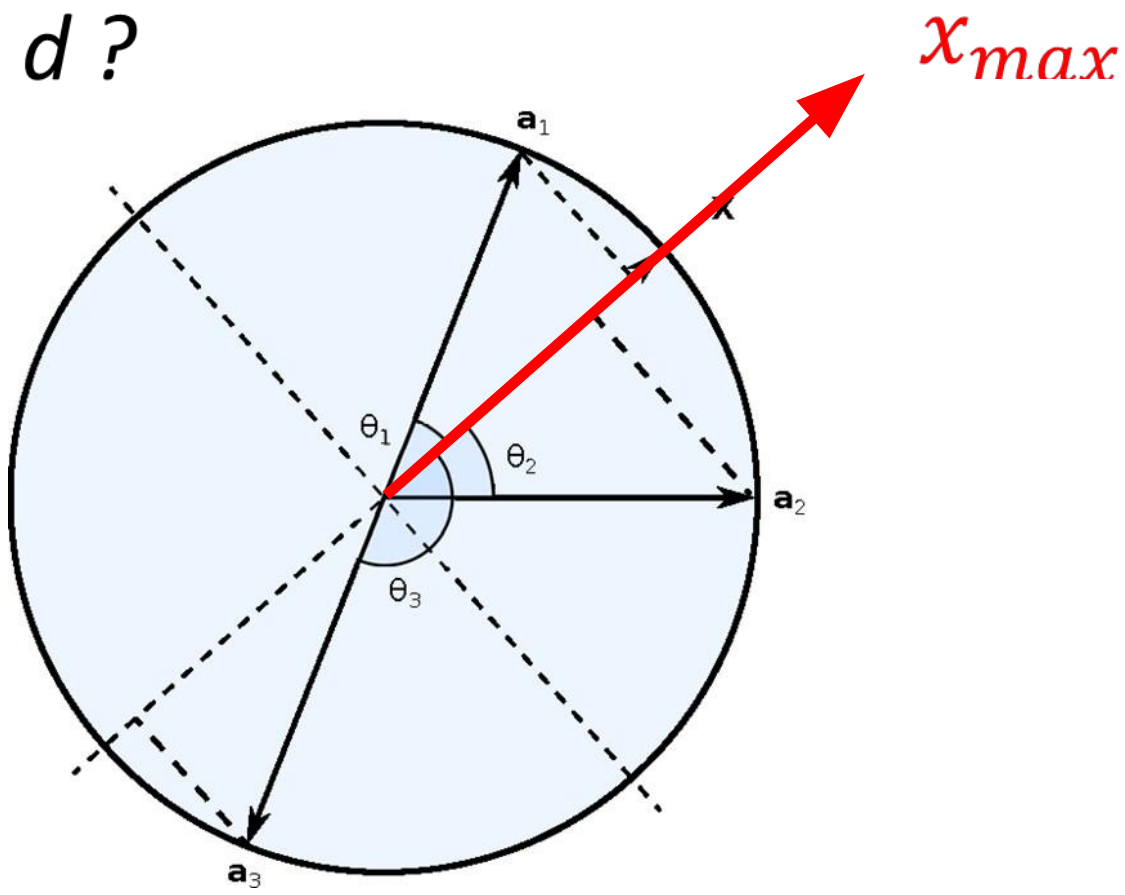
- $\boldsymbol{b} = d\ (1, \ldots, 1)$ ⟹ $\boldsymbol{b}$ is in the DC subspace
- How to choose bias parameter d ?

$$y_{d,k} = \mathbf{a}_k^T \mathbf{x} + b_k \qquad \|\mathbf{a}_k^T\| = 1$$

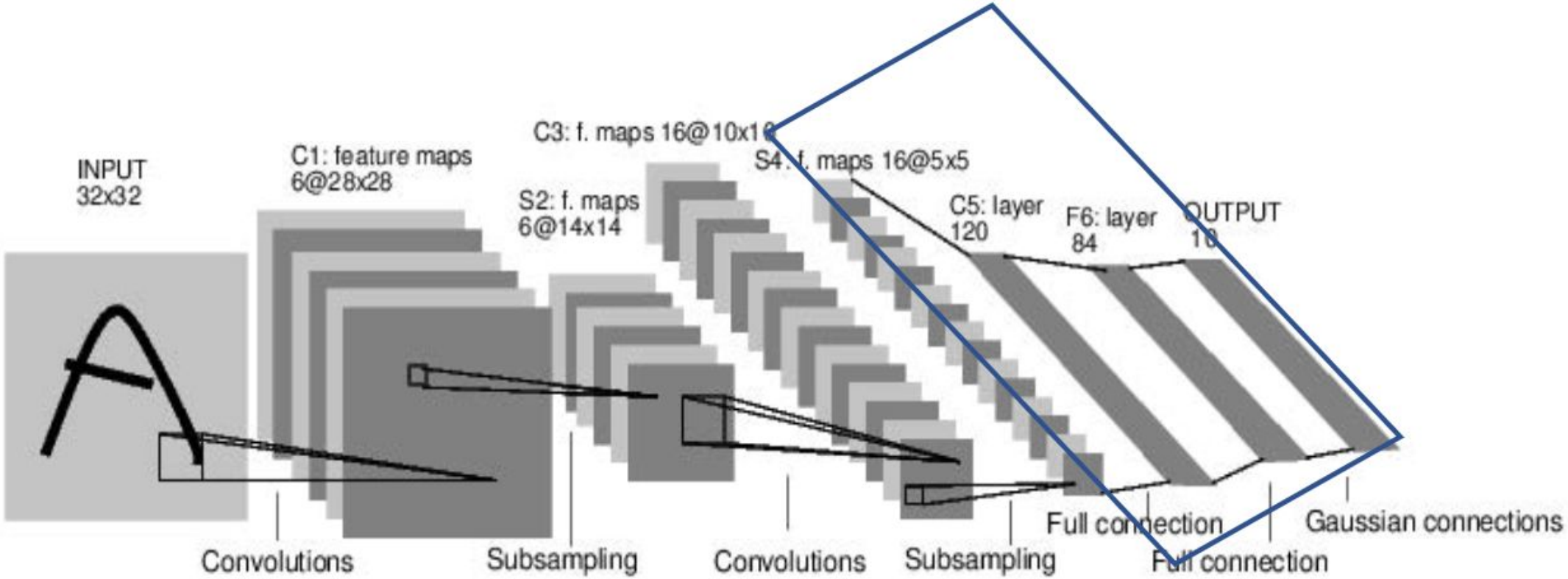
$$\Rightarrow \quad -\|\mathbf{a}_k^T\|\|\mathbf{x}\| \leqslant \|\mathbf{a}_k^T \mathbf{x}\| \leqslant \|\mathbf{a}_k^T\|\|\mathbf{x}\|$$

$$\Rightarrow \quad -\max \|\mathbf{x}\| \leqslant \max_{\mathbf{x}} \|\mathbf{a}_k^T \mathbf{x}\| \leqslant \max \|\mathbf{x}\|$$

$$\Rightarrow \quad b_k \geqslant \max_{\mathbf{x}} \|\mathbf{x}\|$$



Selecting Parameters in FC Layers



2 FC layers (120D, 84D) + 1 output layer (10D)

Two Ideas in Parameters Selection

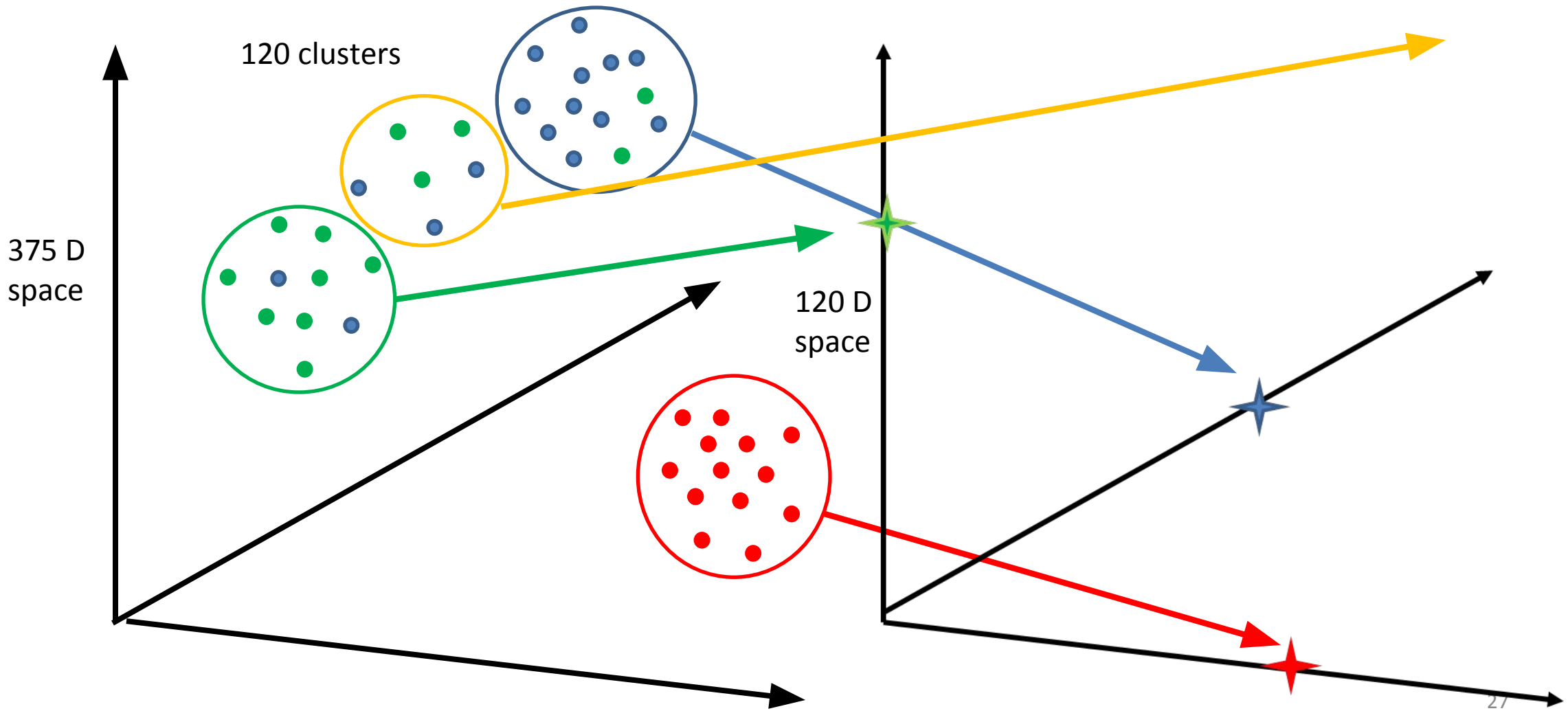
1st viewpoint (BP)

- Parameters to optimize in large nonlinear networks
- Backpropagation – SGD

2nd viewpoint (FF)

- Parameters of linear least-squared regression (LSR) models
- Label-assisted linear LSR
  - True label used in the output layer
  - Pseudo label used in intermediate FC layers

LSR Problem Setup



Hard Pseudo-Labels

- **Training phase** (use 375D-to-120D FC layer as an example)
  - K-mean clustering
    - Cluster samples of each object class into 12 sub-clusters
    - Assign a pseudo label to samples in each sub-cluster

      Ex. 0-i, 0-ii, … , 0-xii, 1-i, 1-i, …, 1-xii, …, 9-i, 9-ii, …, 9-xii

      12 pseudo labels 12 pseudo labels 12 pseudo labels
  - Least squared regression (LSR)
    - Set up an LSR model (one sub-cluster -> one equation)
      - Inputs of 375D
      - Outputs of 120D (one-hot vectors)

Filter Weights Determination via LSR

LSR Model Parameters | **Input Data Vectors/Matrix** | **Output One-hot Vectors/Matrix**

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & w_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & w_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{c1} & a_{c2} & \cdots & a_{cn} & w_c \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \\ 1 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_c \end{bmatrix}$$

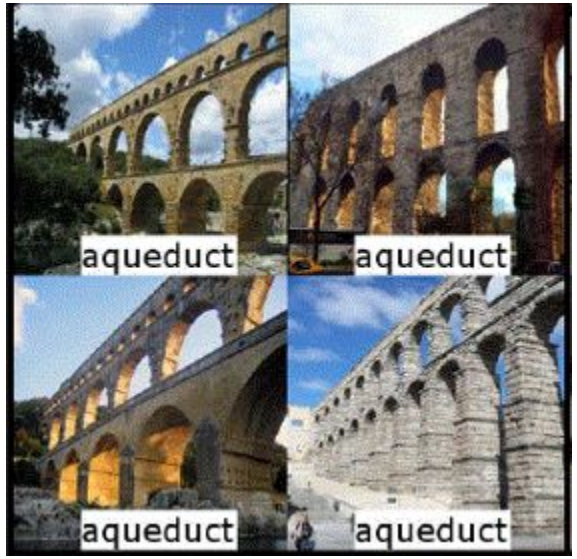
- **Intermedia FC Layer: using pseudo labels with c=120 or 84**
- **Output layer using true labels with c=10**

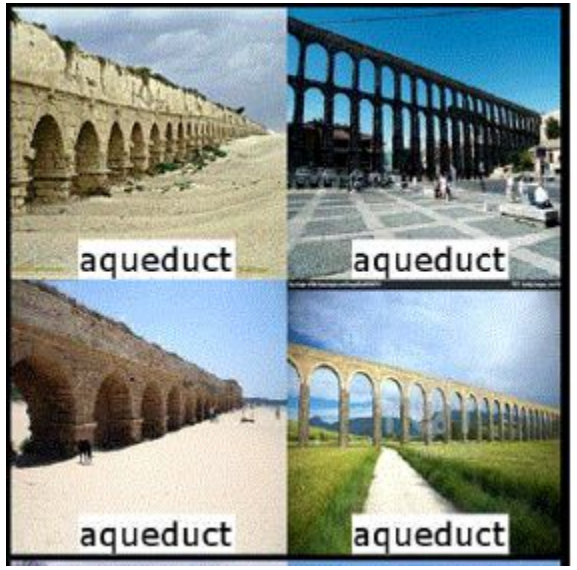
Why Pseudo-Labels?

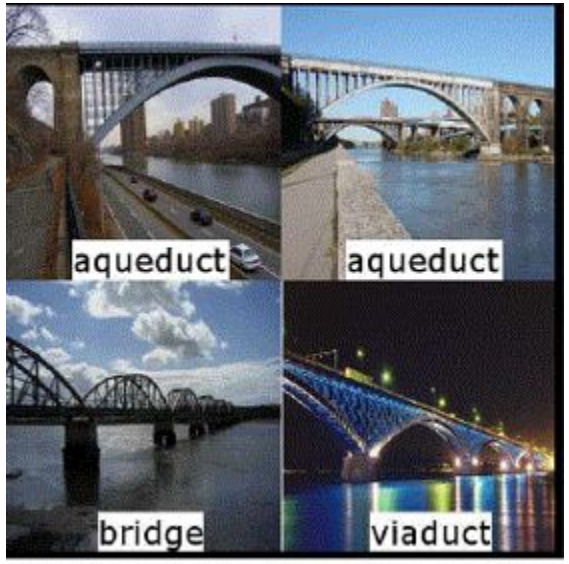
Intra-class variability: example #1

Why Pseudo-Labels?

Intra-class variability: example #2



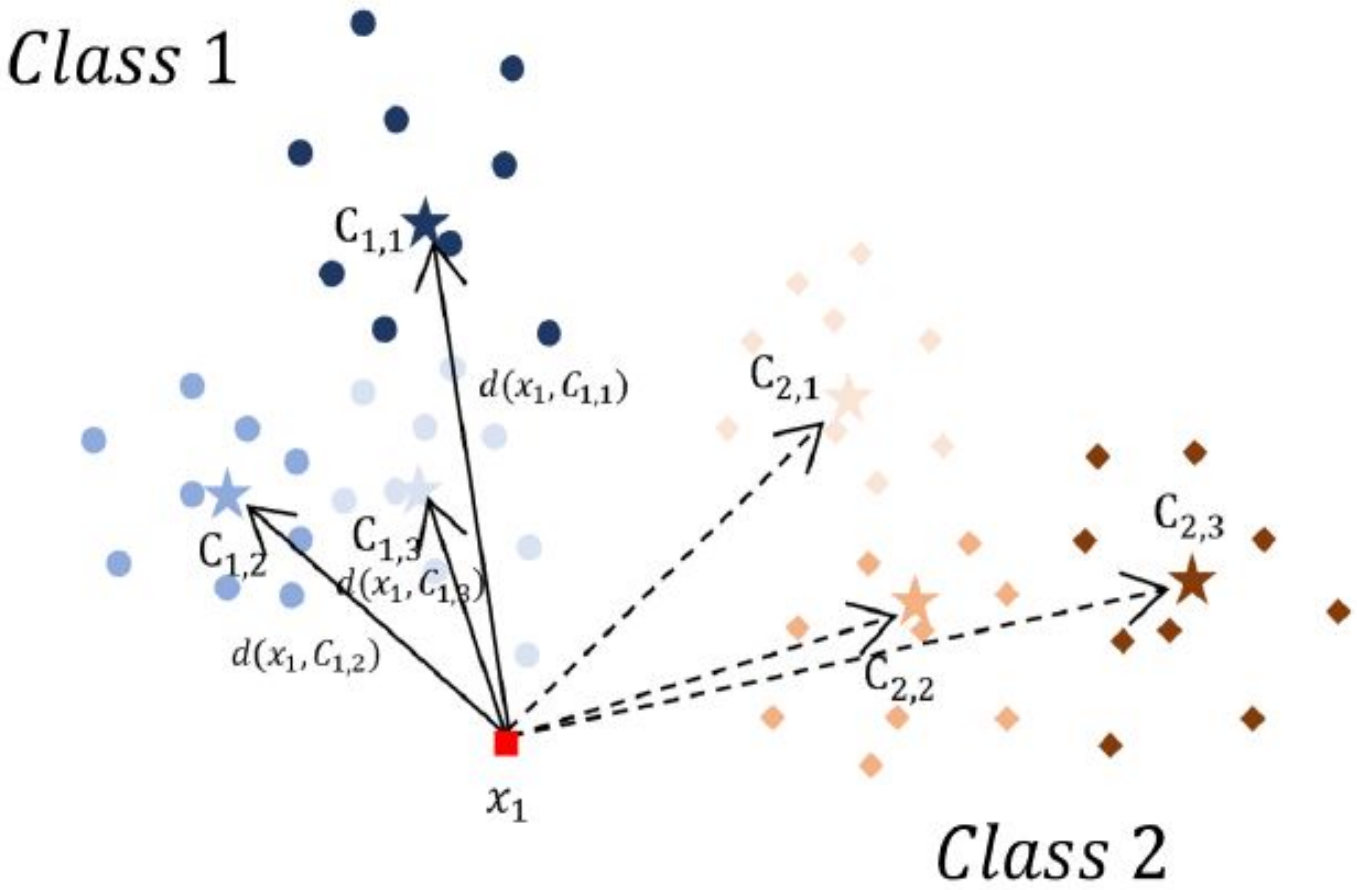




Soft Pseudo-Labels

$$\mathrm{Prob}(\mathbf{x}_j, \mathbf{c}_{j,l}) = \frac{\exp(-\alpha d(\mathbf{x}_j, \mathbf{c}_{j,l}))}{\sum_{l=1}^{L} \exp(-\alpha d(\mathbf{x}_j, \mathbf{c}_{j,l}))},$$

$$\mathrm{Prob}(\mathbf{x}_j, \mathbf{c}_{j',l}) = 0 \quad \text{if } j \neq j',$$



Label-Assisted Regression (LAG)

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & w_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & w_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{M1} & a_{M2} & \cdots & a_{Mn} & w_M \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1(\mathbf{x}) \\ \vdots \\ \mathbf{p}_j(\mathbf{x}) \\ \vdots \\ \mathbf{p}_J(\mathbf{x}) \end{bmatrix},$$

$$\mathbf{p}_j(\mathbf{x}_j) = (\mathrm{Prob}(\mathbf{x}_j, \mathbf{c}_{j,1}), \cdots \mathrm{Prob}(\mathbf{x}_j, \mathbf{c}_{j,l}), \cdots \mathrm{Prob}(\mathbf{x}_j, \mathbf{c}_{j,L}))^T$$

$$\mathbf{p}_j(\mathbf{x}_{j'}) = \mathbf{0}, \quad \text{if } j \neq j',$$

CIFAR-10: Modified LeNet-5 Architecture

Architecture	Original LeNet-5	Modified LeNet-5
1st Conv Layer Kernel Size	5x5x1	5x5x3
1st Conv Layer Filter No.	6	32
2nd Conv Layer Kernel Size	5x5x6	5x5x32
2nd Conv Layer Filter No.	16	64
1st FC Layer Filter No.	120	200
2nd FC Layer Filter No.	84	100
Output Node No.	10	10

MNIST

CIFAR-10

Classification Performance

Testing Accuracy

Dataset	MNIST	CIFAR-10
FF	97.2%	62%
Hybrid	98.4%	64%
BP	99.1%	68%

Decision Quality

Feature Quality

Hybrid: Convolutional layers (FF) + FC layers (BP-optimized MLP)

Adversarial Attacks

Case 1: Attacking BP-CNN using Deepfool

	Clean MNIST	Attacked MNIST	Clean CIFAR-10	Attacked CIFAR-10
BP	99.9%	1.7%	68%	14.6%
FF	97.2%	95.7%	62%	58.8%

Case 2: Attacking FF-CNN using Deepfool

	Clean MNIST	Attacked MNIST	Clean CIFAR-10	Attacked CIFAR-10
BP	99.9%	97%	68%	68%
FF	97.2%	2%	62%	16%

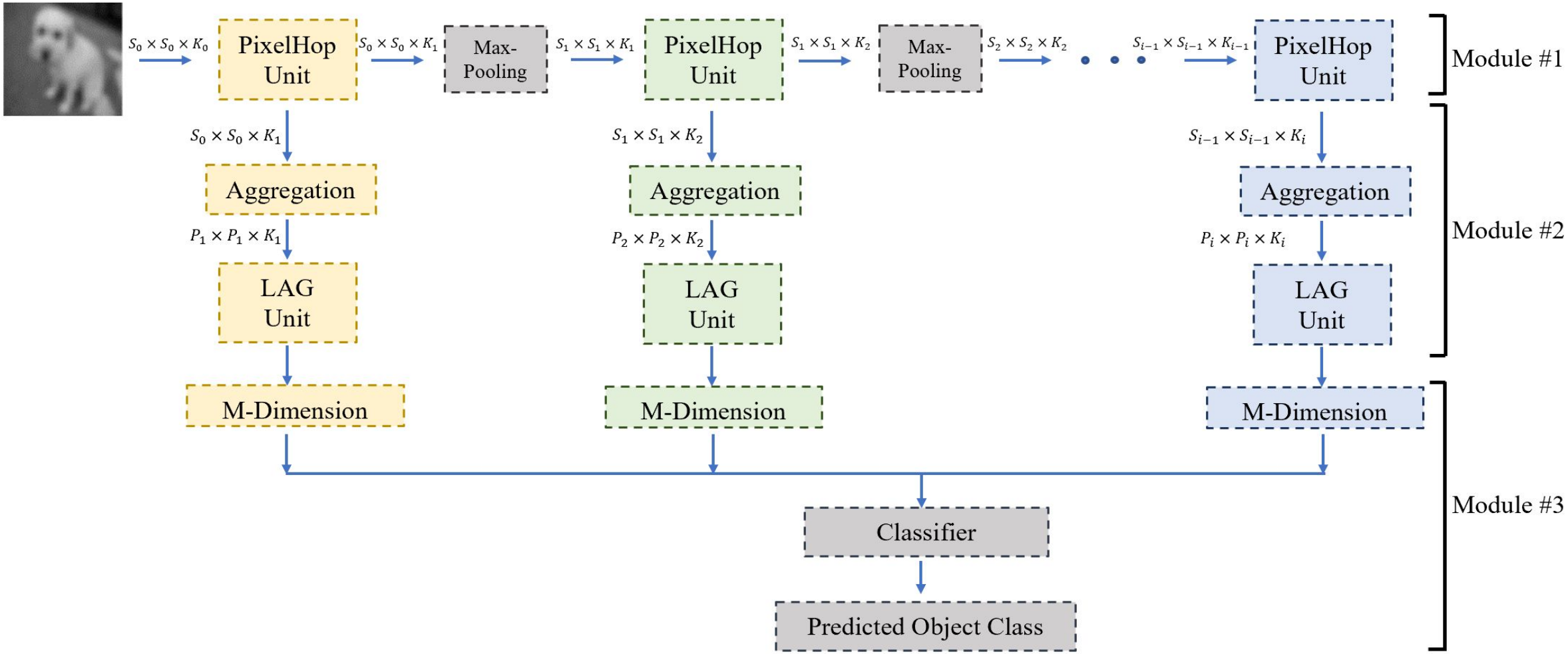
Limitations of FF-CNN

- **Lower classification accuracy**
  - Can we use FF-CNN to initialize BP-CNN? -> no advantage
  - The label information is used after the convolutional layers
  - How to introduce the label information earlier?
- **Vulnerability to adversarial attacks**
  - BP-CNN and FF-CNN are both vulnerable to adversarial attacks since there exists a direct path from the output (or decision) layer to the input (or source image) layer
- **Multi-tasking**
  - One network for one specific task
- **One solution**
  - We need to abandon the network architecture

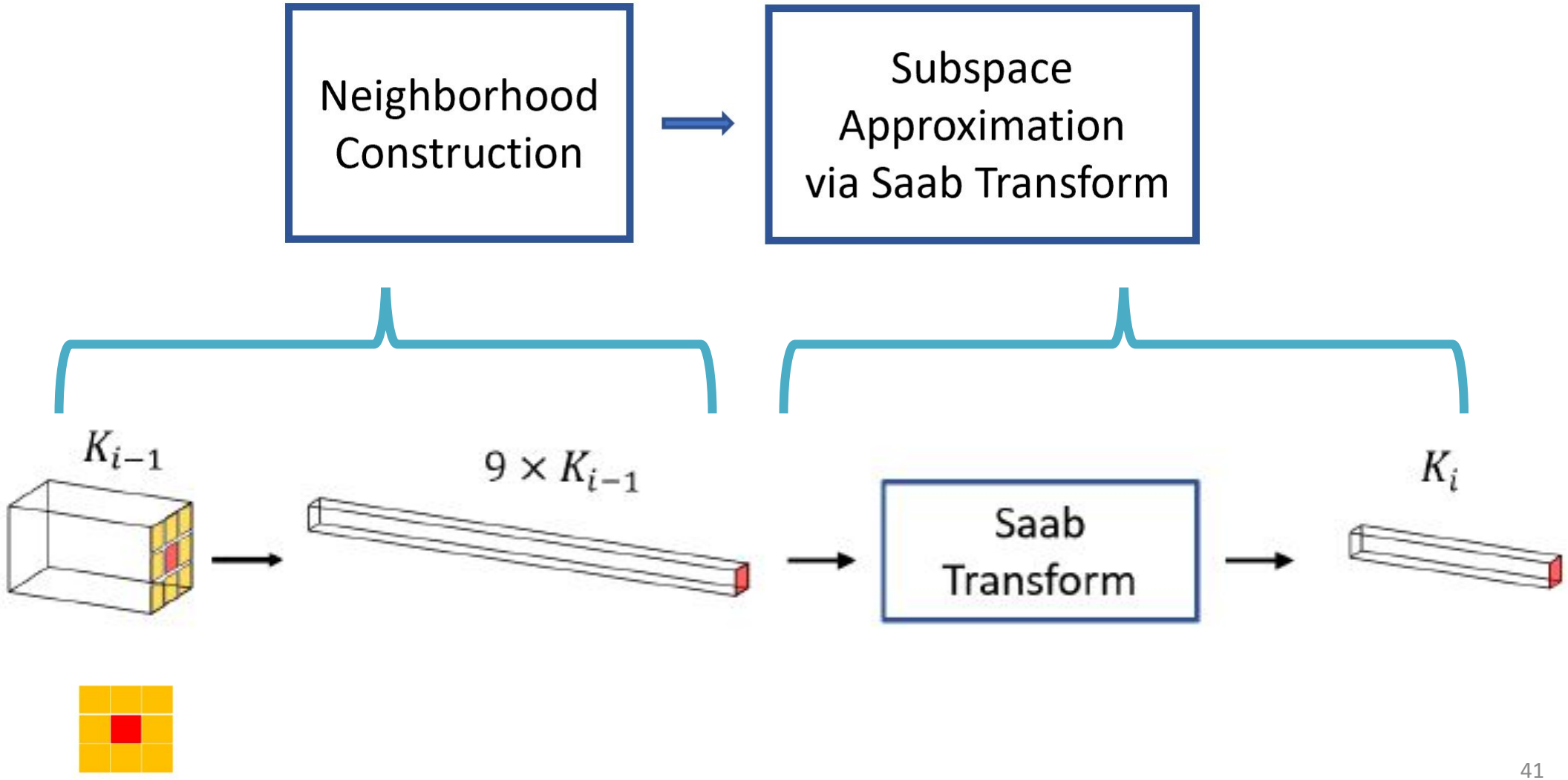
Successive Subspace Learning (SSL)

PixelHop: An SSL Method for Image Classification

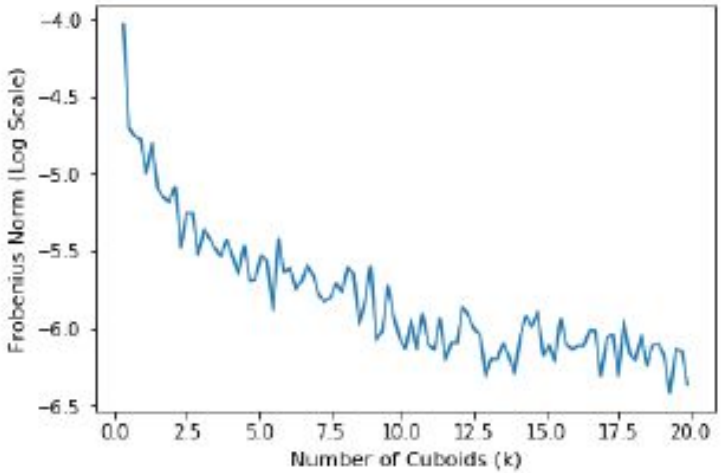
PixelHop System (No More A Network)



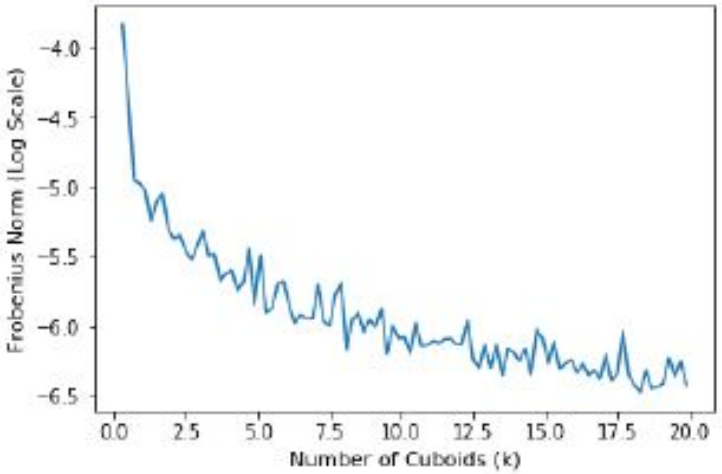
PixelHop Unit



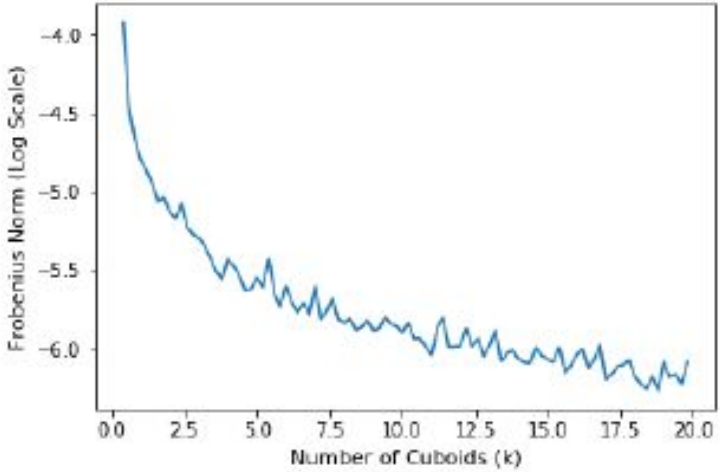
Convergence of Saab Filters (1)



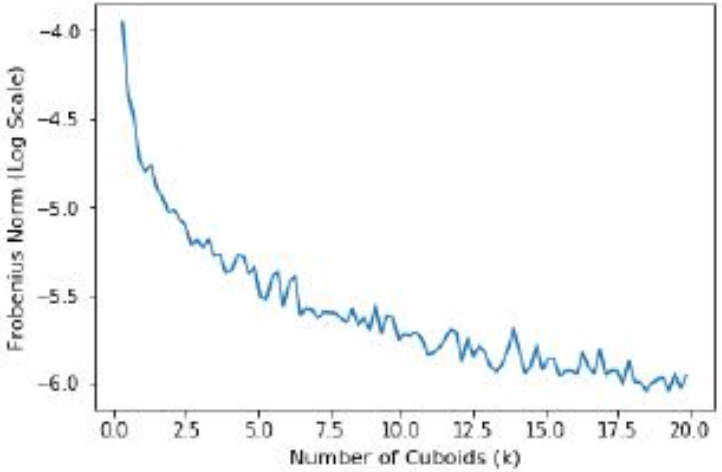
(a) PixelHop Unit1



(b) PixelHop Unit2

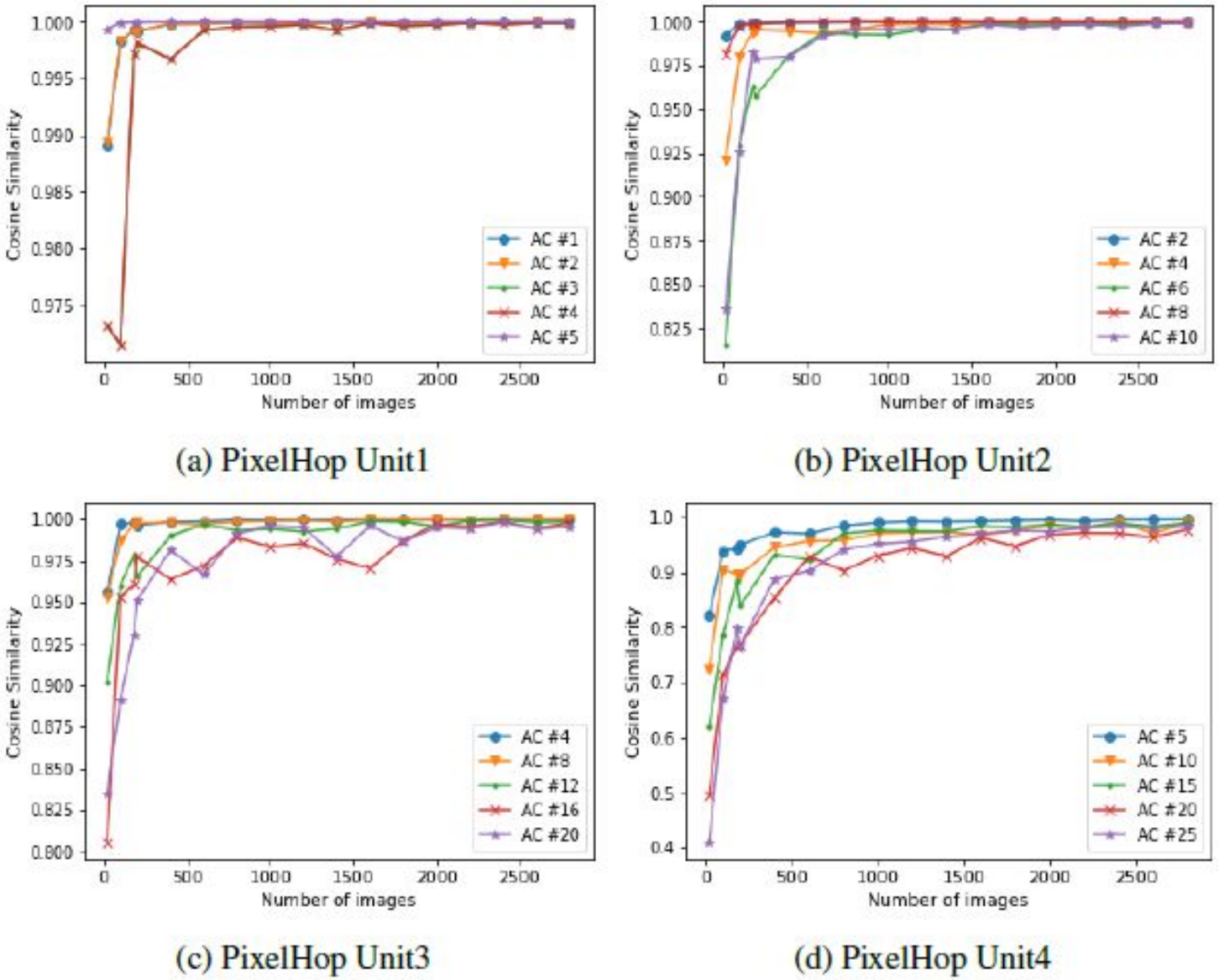


(c) PixelHop Unit3



(d) PixelHop Unit4

Convergence of Saab Filters (2)



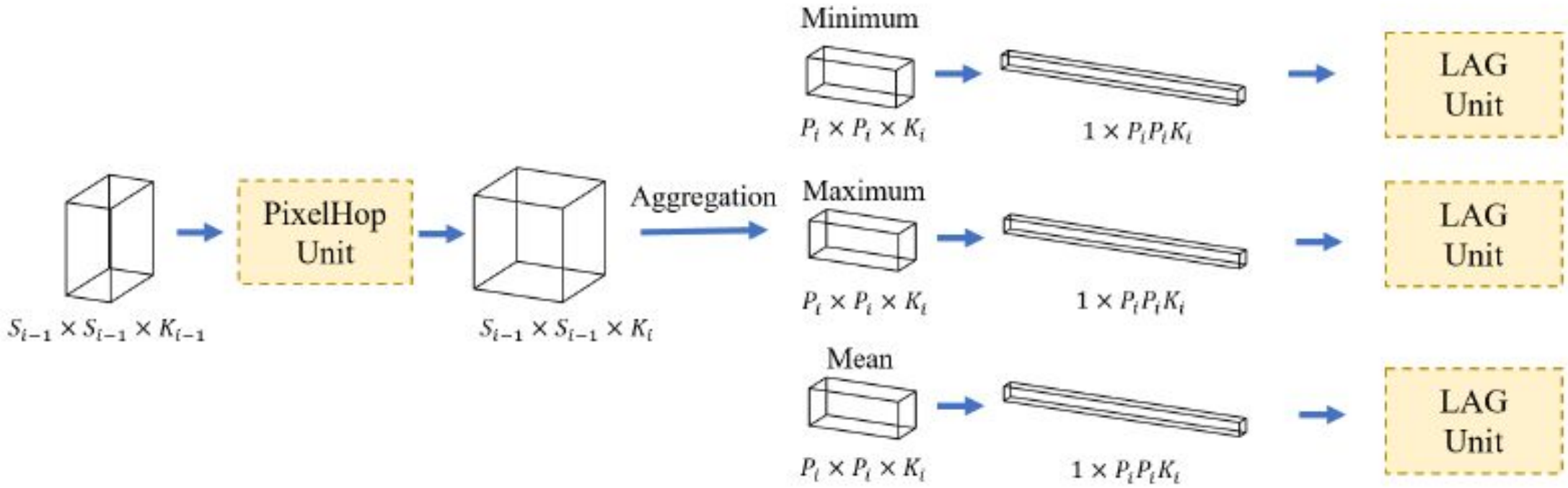
(a) PixelHop Unit1

(b) PixelHop Unit2

(c) PixelHop Unit3

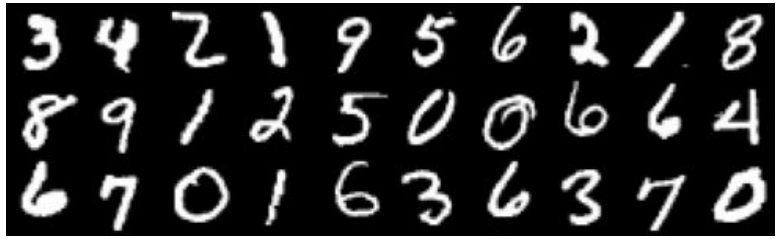
(d) PixelHop Unit4

Aggregation



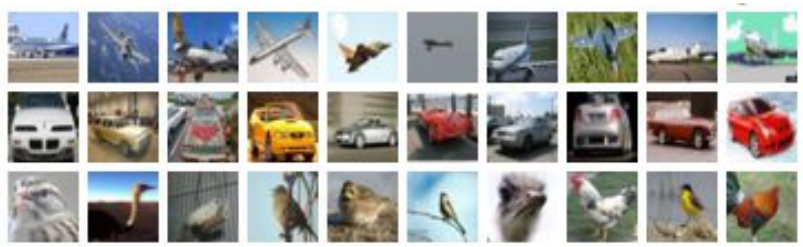
Experiment Set-up

- Datasets:
  - MNIST
    - Handwritten digits 0-9
    - Gray-scale images with size 32x32
    - Training set: 60k, Testing set: 10k
  - Fashion-MNIST
    - Gray-scale fashion images with size 32 × 32
    - Training set: 60k, Testing set: 10k
  - CIFAR-10
    - 10 classes of tiny RGB images with size 32 × 32
    - Training set: 50k, Testing set: 10k
- Evaluation:
  - Top-1 classification accuracy



MNIST

Fashion-MNIST

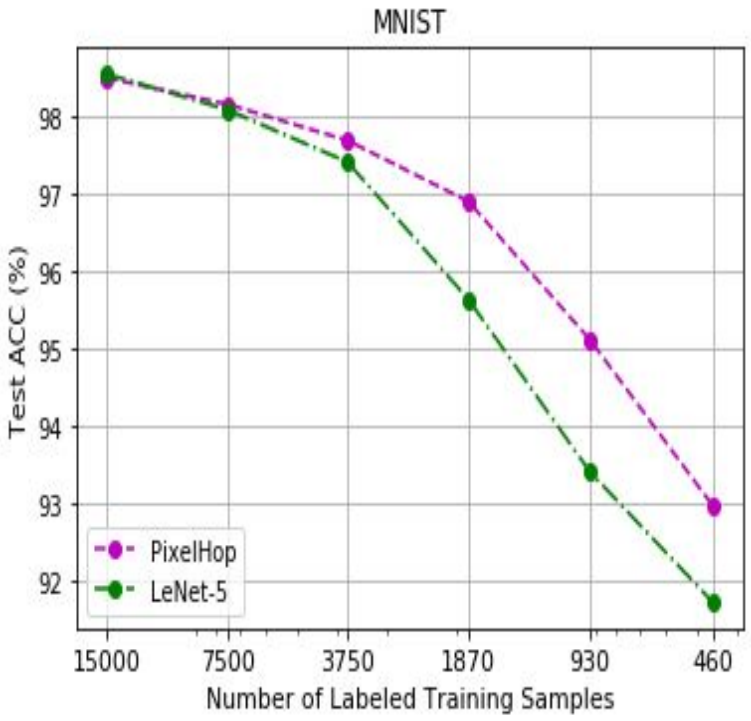


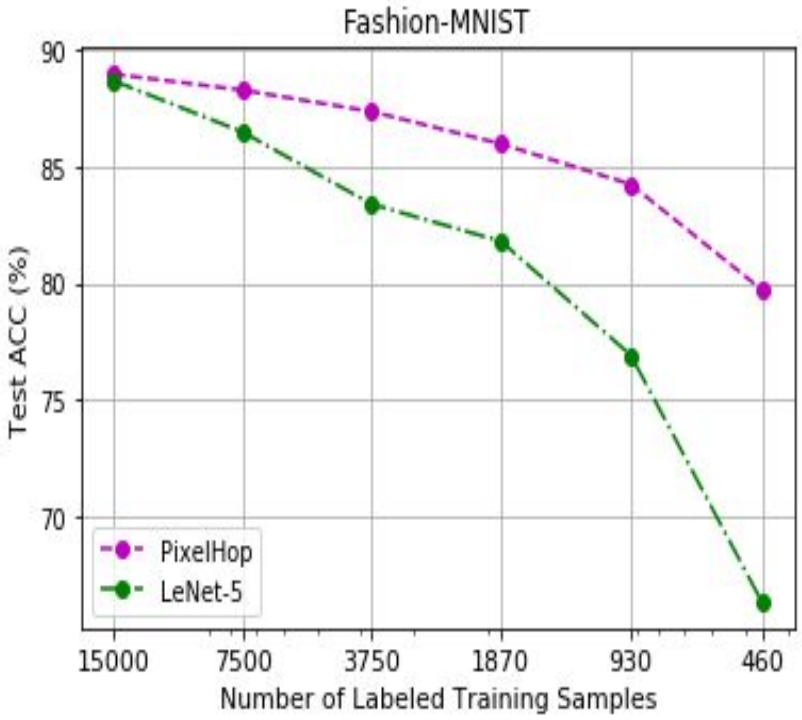
CIFAR-10

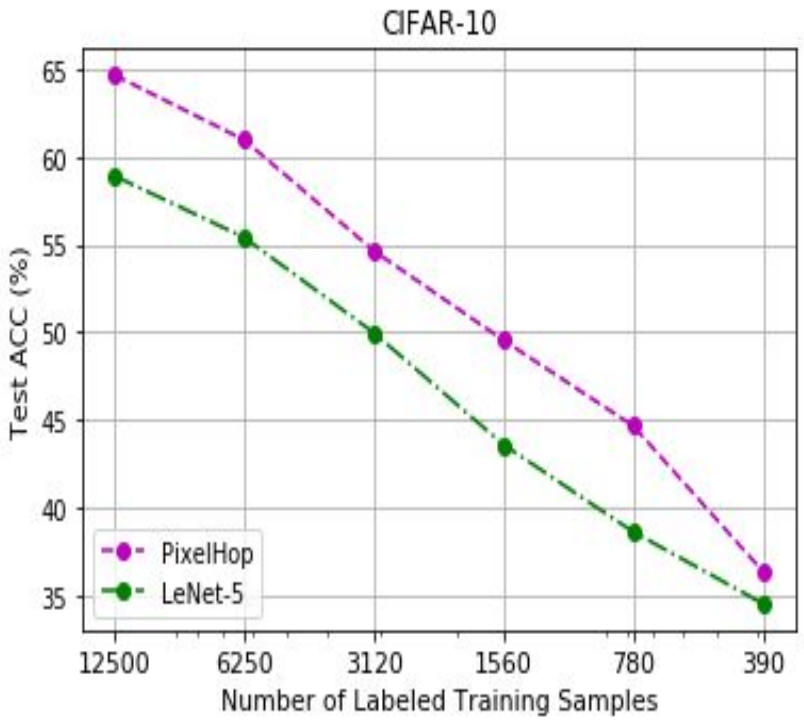
Performance Comparison

Method	MNIST	Fashion MNIST	CIFAR-10
LeNet-5	99.04	91.08	68.72
FF-CNN	97.2	–	62
PixelHop	98.90	91.30	71.37
PixelHop$^+$	**99.09**	**91.68**	**72.66**

Weakly-Supervised Learning







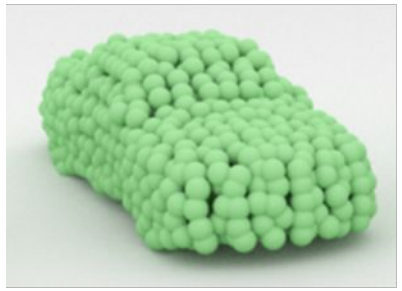
PointHop: An SSL Method for Point Cloud Classification

Point Cloud Set

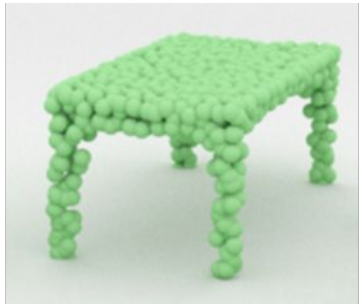
- Set of points in 3D, acquired by 3D scanning devices such as Lidar
- Applications: AR/VR, self-driving cars, robots, 3D CAD modelling etc.

Point Cloud Processing

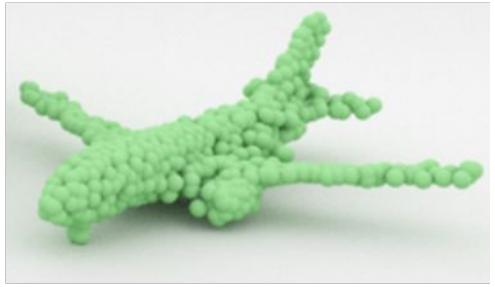
- Classification



CAR



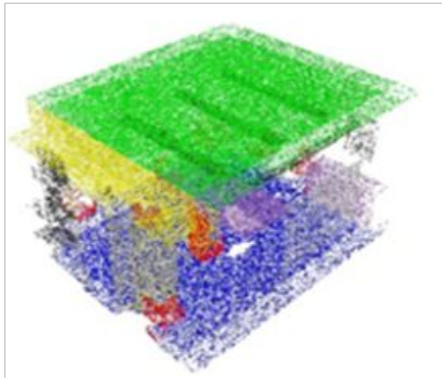
TABLE



AIRPLANE

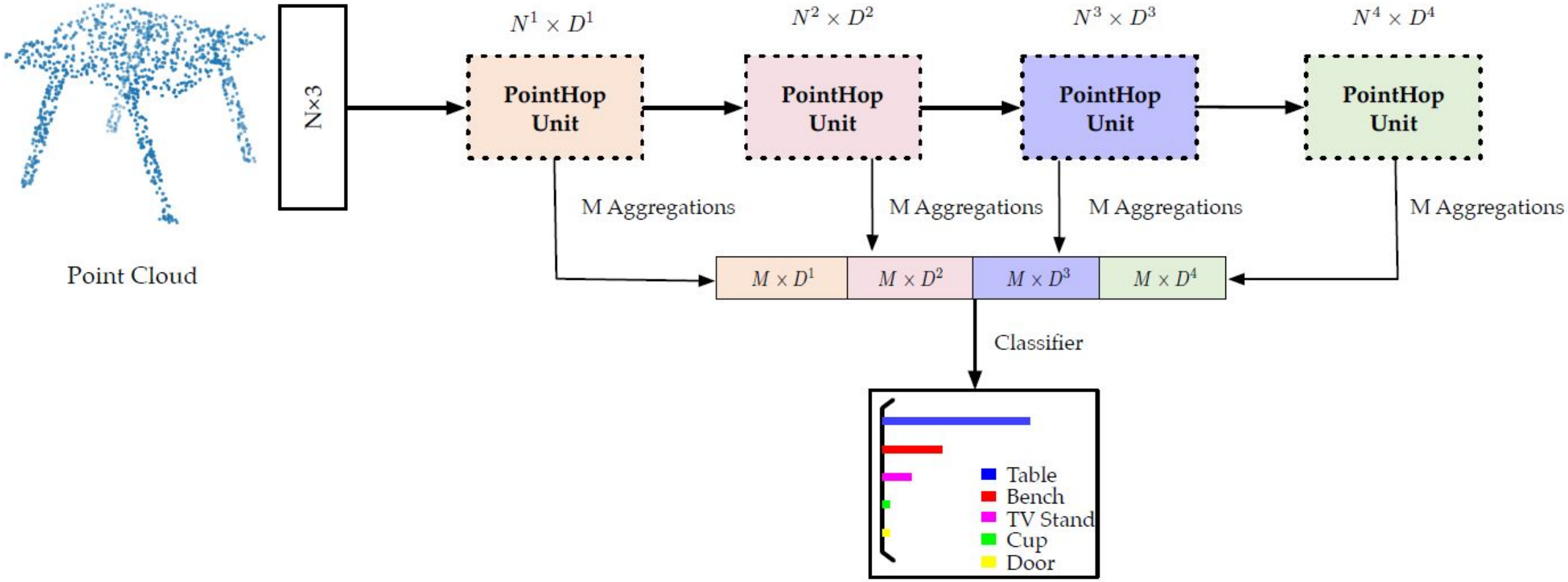
- Segmentation

AIRPLANE

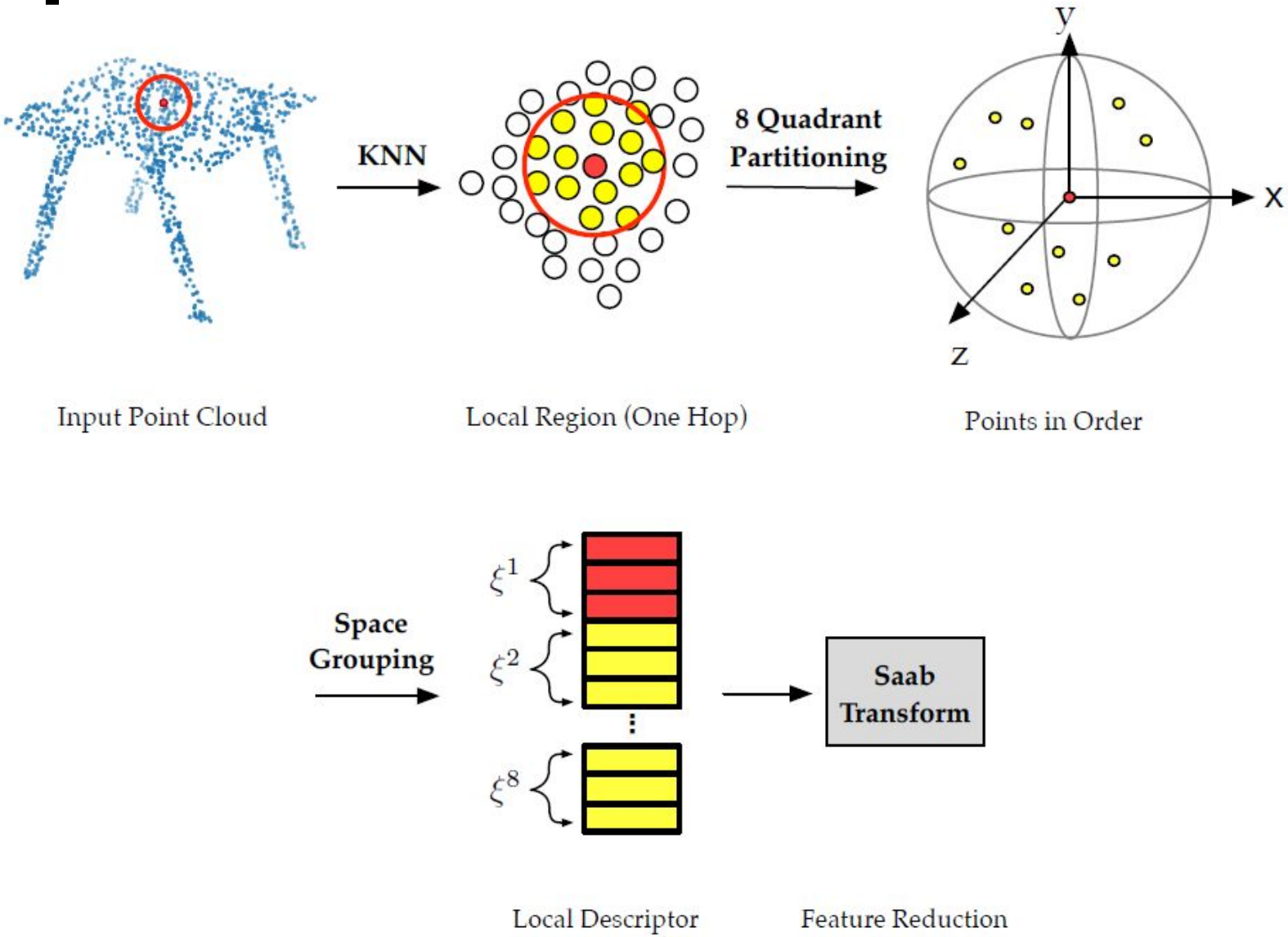


BUILDING

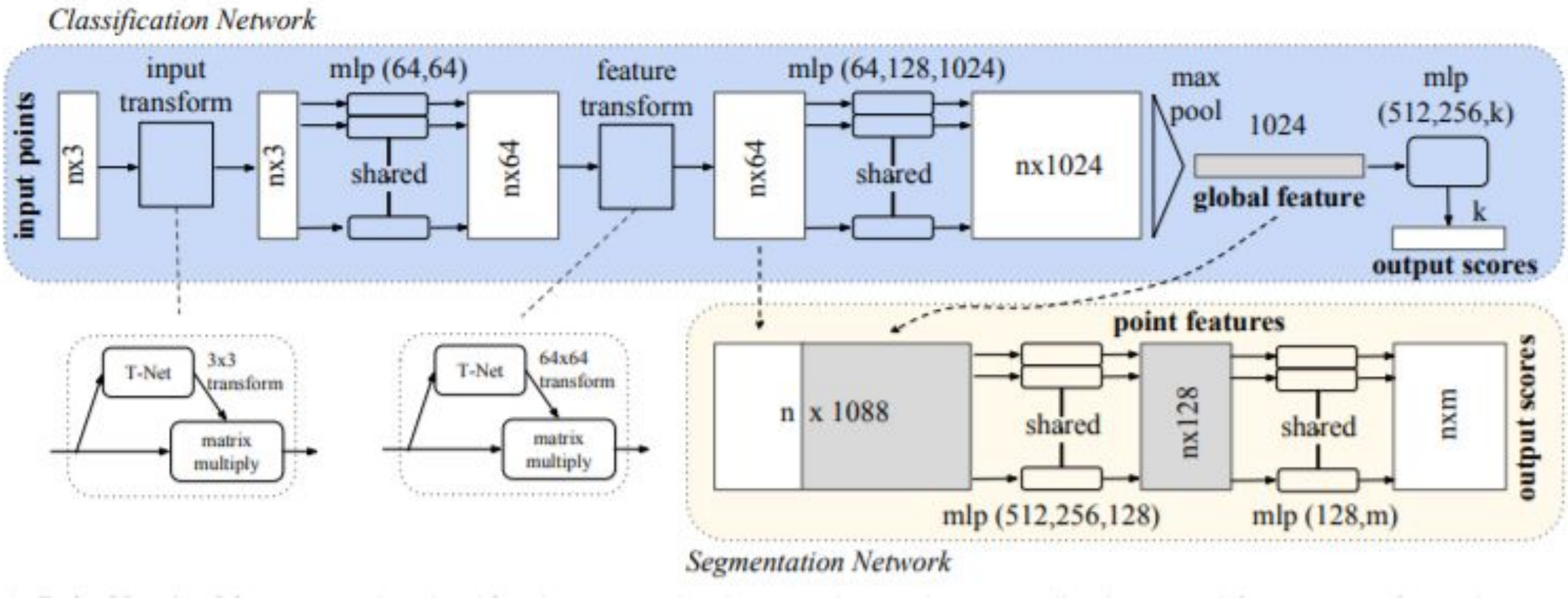
PointHop System



PointHop Unit

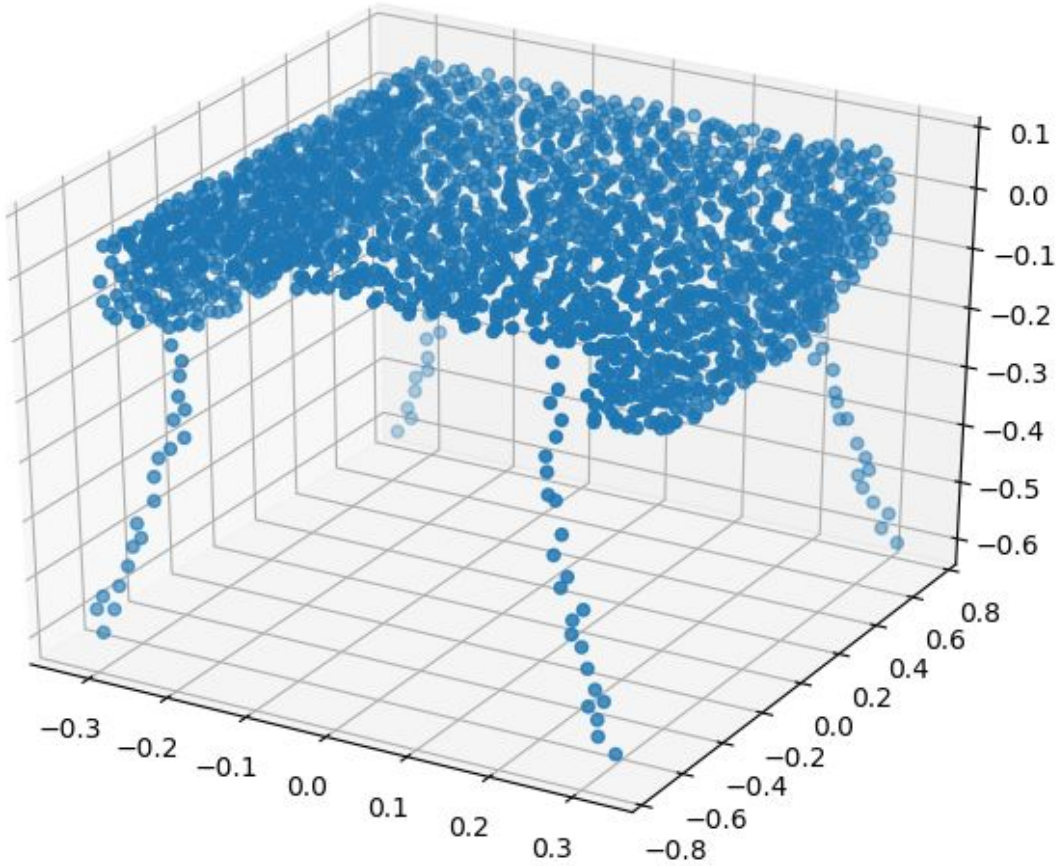


PointNet Architecture



Dataset – ModelNet40

- 40 categories of objects such as airplane, table, desk, sofa
- 9840 training samples, 2468 testing samples
- Every sample has 2468 points
- Every point has 3 coordinates



Performance Comparison

Dataset: ModelNet40

Method	Feature extraction	Average accuracy (%)	Overall accuracy (%)
PointNet [16]	Supervised	86.2	89.2
PointNet++ [17]		-	90.7
PointCNN [34]		88.1	92.2
DGCNN [18]		90.2	92.2
PointNet baseline (Handcraft, MLP)	Unsupervised	72.6	77.4
PointHop (baseline)		83.3	88.65
PointHop		84.4	**89.1**

Training Time Comparison

GPU platform: NVIDIA GeForce GTX 1080
CPU platform: Intel Xeon CPU E5-2620 v3 at 2.40GHz

Method	Total training time	Device
PointNet (1,024 points)	$\sim$ 5 hours	GPU
PointHop (256 points)	$\sim$ 5 minutes	CPU
PointHop (1,024 points)	$\sim$ 20 minutes	CPU

Conclusion: Similarities of SSL and DL

	SSL	DL
Information collection	Successively growing neighborhoods	Gradually enlarged receptive fields
Information processing	Trade spatial dimension for spectral dimension	Trade spatial dimension for spectral dimension
Spatial information reduction	Spatial pooling	Spatial pooling

Conclusion: Differences of SSL and DL

	SSL	DL
Model expandability	Non-parametric model	Parametric model
Incremental learning	Easy	Difficult
Model architecture	Flexible	Networks
Model interpretability	Easy	Difficult
Model parameter search	Feedforward design	Backpropagation
Training/testing complexity	Low	High
Spectral dim. reduction	Subspace approximation	Number of filters
Task-independent features	Yes	No
Multi-tasking	Easy	Difficult
Incorporation of priors and constraints	Easy	Difficult
Weak supervision	Easy	Difficult
Adversarial Attacks	Difficult	Easy

Conclusion

- **Era of data science and engineering has arrived**
  - Seamless integration of knowledge-based and data-driven approaches will be more powerful
- **Data-driven (rather than feature-driven) machine learning is on the rise**
  - Deep learning is a brute-force black-box approach
  - SSL provides an alternative solution
    - Easier for integration with the knowledge-based priors since no hidden layers/variables in SSL
    - Need a feedforward “attention” mechanism

Main References

1. C.-C. Jay Kuo, “Understanding convolutional neural networks with a mathematical model,” the Journal of Visual Communications and Image Representation, Vol. 41, pp. 406-413, November 2016.
2. C.-C. Jay Kuo and Yueru Chen, “On data-driven Saak transform,” the Journal of Visual Communications and Image Representation, Vol. 50, pp. 237-246, January 2018.
3. C.-C. Jay Kuo, Min Zhang, Siyang Li, Jiali Duan and Yueru Chen, “Interpretable Convolutional Neural Networks via Feedforward Design,” the Journal of Visual Communications and Image Representation, Vol. 60, pp. 346-359, 2019 (or arXiv 1810.02786).
4. M. Zhang, H. You, P. Kadam, S. Liu, C.-C. J. Kuo, “PointHop: an explainable machine learning method for point cloud classification,” arXiv preprint arXiv:1907.12766 (2019), to appear in IEEE Trans. on Multimedia.
5. Yueru Chen and C.-C. Jay Kuo, “PixelHop: a successive subspace learning (SSL) method for object classification,” arXiv preprint arXiv:1909.08190 (2019), to appear in the Journal of Visual Communications and Image Representation.