

NUS Research Week 2022
January 7, 2022

A New Learning Paradigm - Green Learning

C.-C. Jay Kuo
University of Southern California

Concerns with Deep Learning

- **May not be suitable for academic research**

- Demanding heavy resources

- Computing resource (GPU)

- Data collection/labeling cost

- Engineering fine-tuning

- Blackbox tools – discouraging original thinking

- **Previous examples**

- Computer graphics and SIGGRAPH

- Image/video coding and standards

Green Learning as An Alternative

Green Machine Learning (or Green AI)

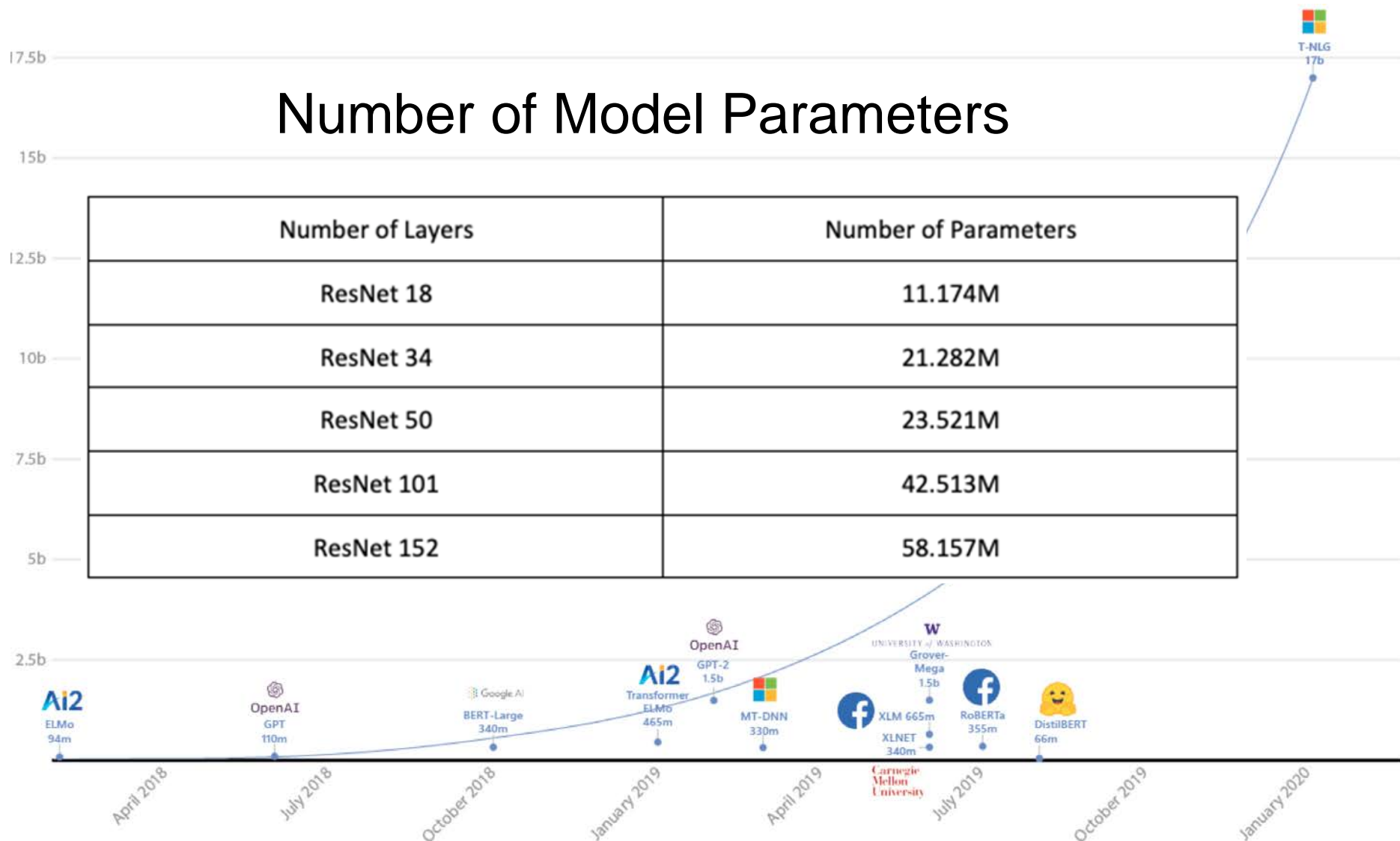
- Decouple “feature extraction” and “decision” again
 - Feature extraction – unsupervised, statistics-based, signal processing (filter banks)
 - Decision – classification, regression, etc.
- Unique characteristics
 - Low power consumption in both training and testing
 - Small model sizes
 - Suitable for edge/mobile devices
 - Also, beneficial to carbon footprint reduction in cloud servers

Outline

- **Why Green Learning?**
- **Fundamentals of Green Learning**
- **Green Learning for Image Classification**
- **Green Learning for Fake Image Detection**
- **Green Learning for Point Cloud Classification and Registration**

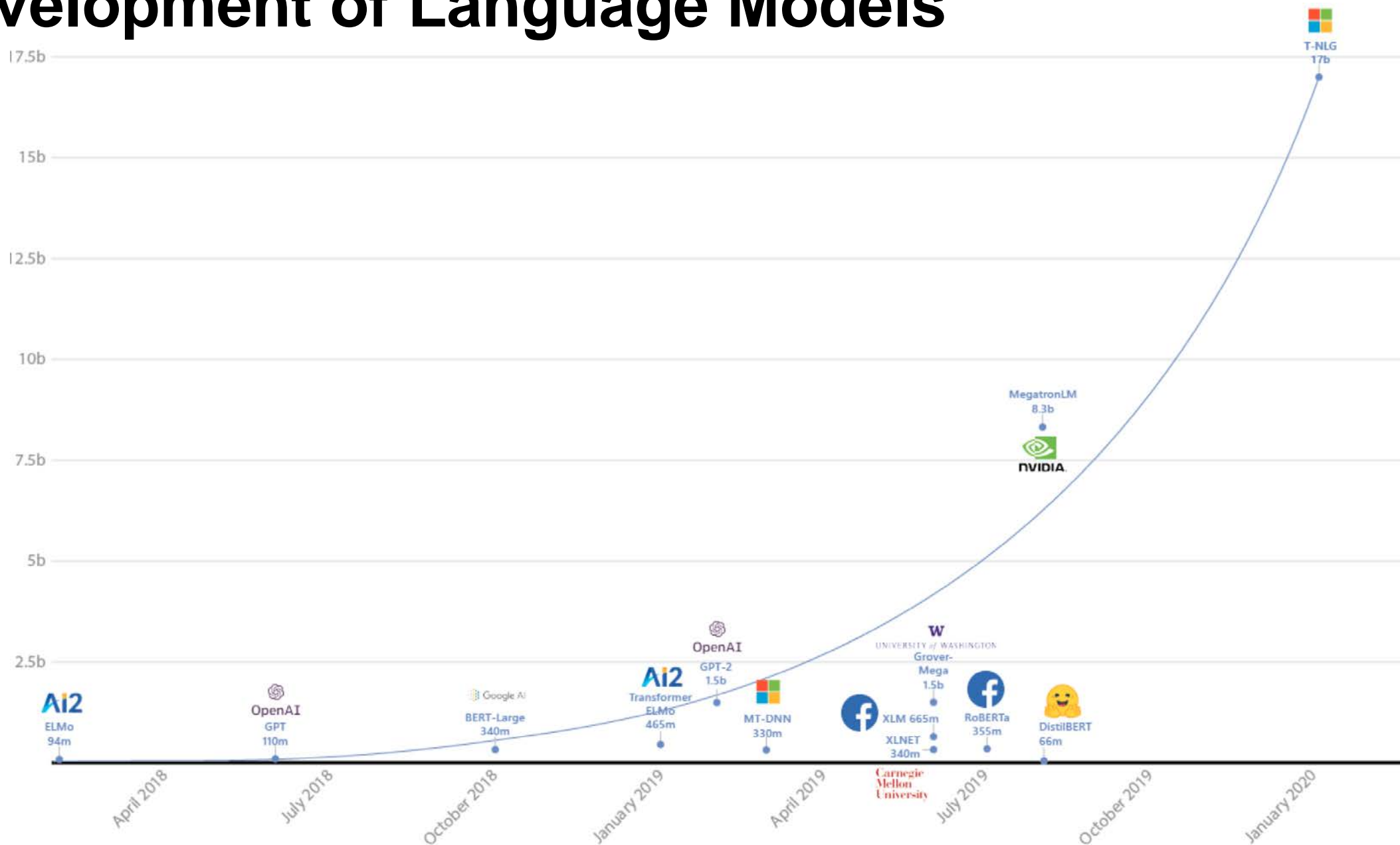
Why Green Learning?

How About Image Models?

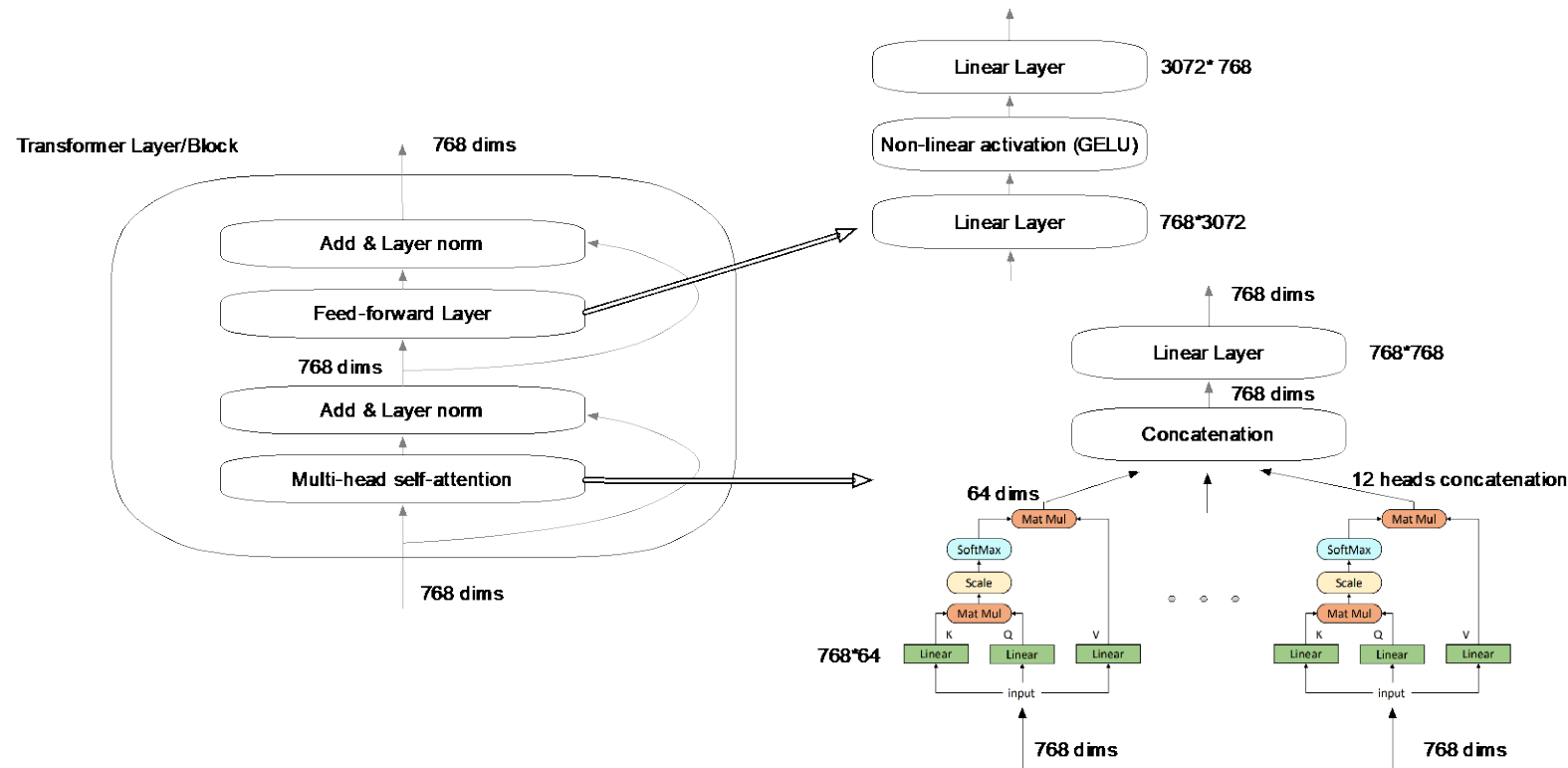


Development of Language Models

No. of model parameters



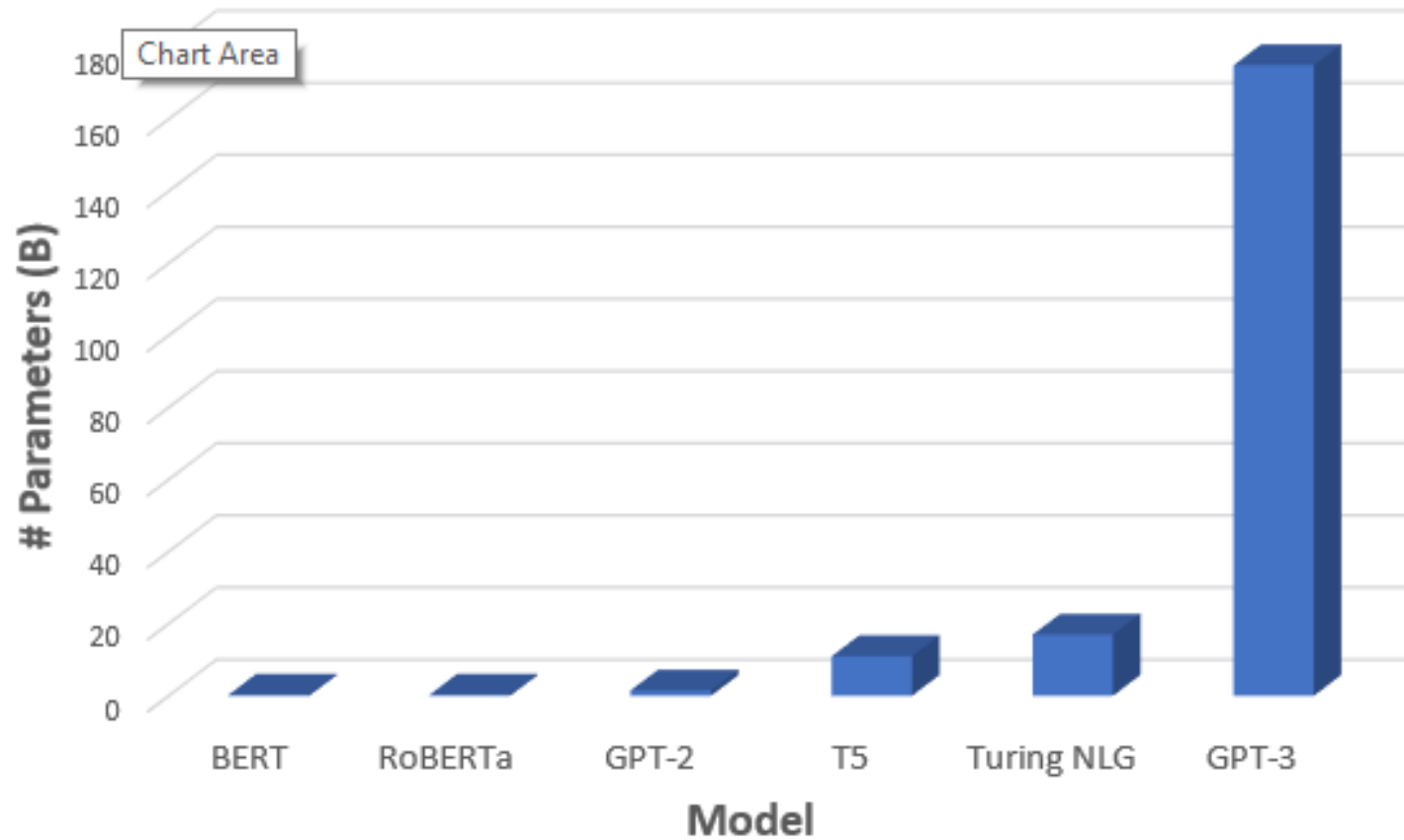
Language Model – Transformer Architecture



- Transformer is much more expensive than CNN
 - Transformer: Multiple feed-forward layers (close to MLP)
 - CNN: shared filters

Development of Language Models

GPT 3 (2020) = 10 * Turing NLG



- **Data hungry**
- **Huge model size**
- **Better performance**

Environmental Problem

Carbon Footprint for DL in NLP

Common carbon footprint benchmarks

in lbs of CO₂ equivalent

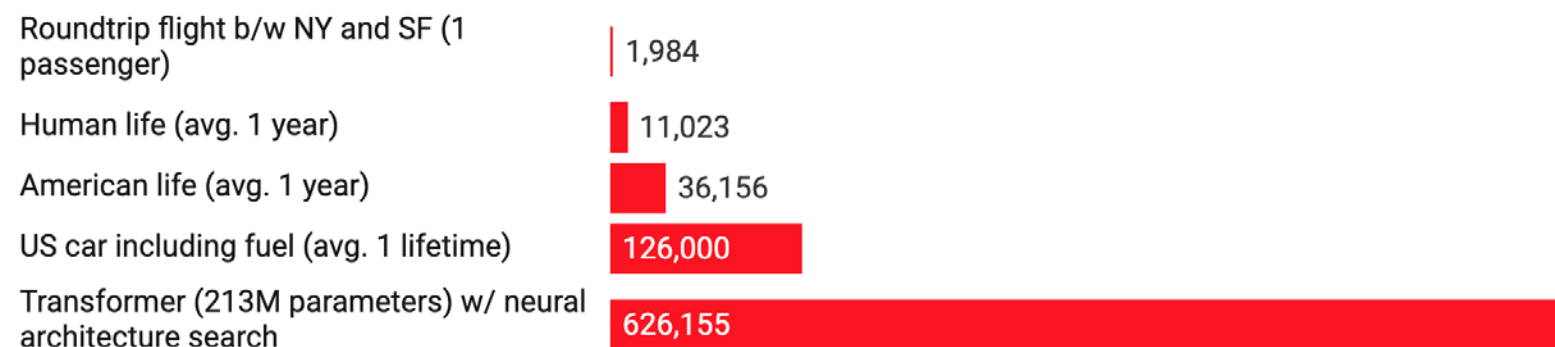


Chart: MIT Technology Review • Source: Strubell et al. • Created with Datawrapper

Consumption	CO ₂ e (lbs)
Air travel, 1 passenger, NY↔SF	1984
Human life, avg, 1 year	11,023
American life, avg, 1 year	36,156
Car, avg incl. fuel, 1 lifetime	126,000

Training one model (GPU)	
NLP pipeline (parsing, SRL)	39
w/ tuning & experimentation	78,468
Transformer (big)	192
w/ neural architecture search	626,155

Table 1: Estimated CO₂ emissions from training common NLP models, compared to familiar consumption.¹



Strubell, Emma, Ananya Ganesh, and Andrew McCallum. "Energy and policy considerations for deep learning in NLP." arXiv preprint arXiv:1906.02243 (2019).

On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? (Timnit Gebru, 2020)

Sharir, Or, Barak Peleg, and Yoav Shoham. "The Cost of Training NLP Models: A Concise Overview." arXiv preprint arXiv:2004.08900 (2020).

What Green Learning Attempts to Achieve

Objectives:

- Low power consumption in training and inference (primary goal)
- Small model size
- High performance
- Weak supervision
- Interpretability

Fundamentals of Green Learning

Green Learning (GL)

Traditional Pattern Recognition Paradigm

- 1st module (from data to features) - feature extraction
- 2nd module (from features to decision) – classifier or regressor

Deep Learning Paradigm

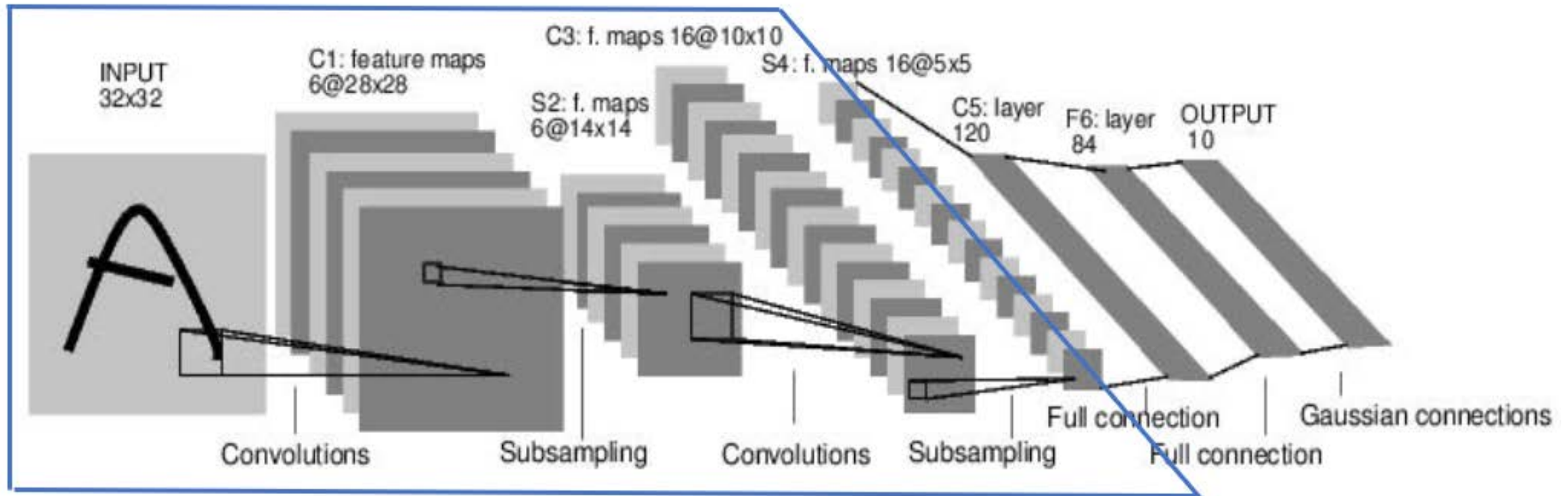
- An integrated module (from data to decision)

Advantages of modular design

- Multi-tasking
- Unsupervised feature learning

Supervised Feature Learning in DL

Exemplary network: LeNet-5



2 convolutional layers + 2 FC layers + 1 output layer

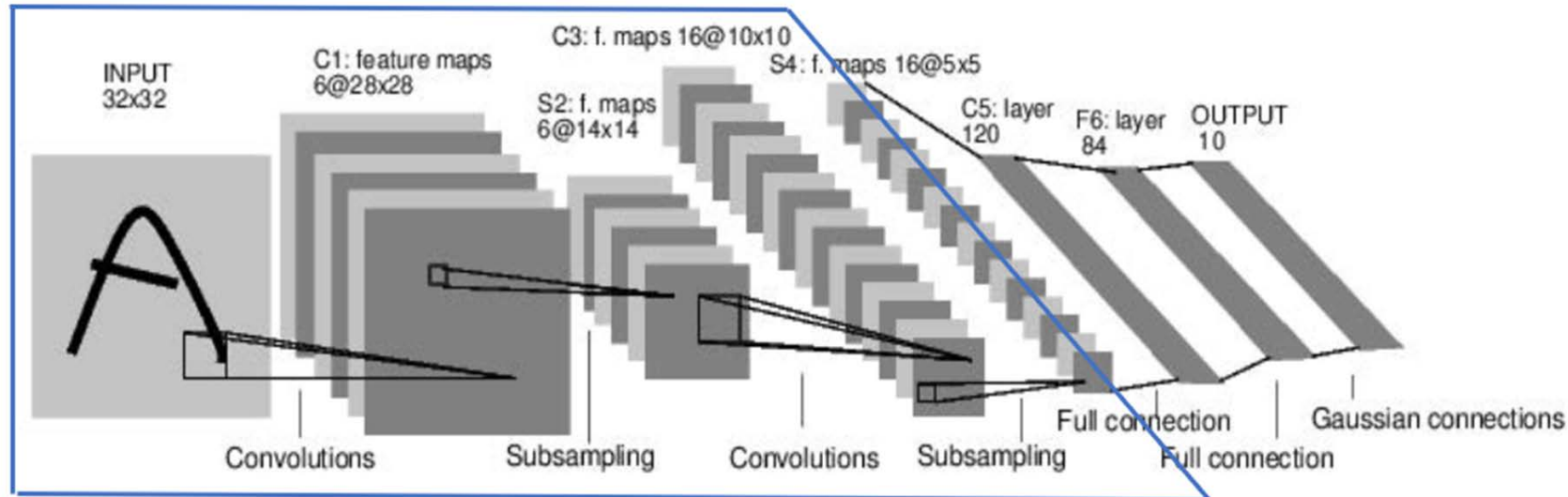
Unsupervised Feature Learning in GL

Filter banks

- Multiple filters operating on local spatial patches
- Joint spatial-spectral representation

Filter kernel design

- Kernels form a base of a linear subspace
- Subspace approximation



One-Stage Transform with Filter Banks

Filter Banks: A set of filters operating in parallel on the input

Example: Laws' 3x3 filters for texture analysis

$$\frac{1}{36} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \quad \frac{1}{12} \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \quad \frac{1}{12} \begin{bmatrix} -1 & 2 & -1 \\ -2 & 4 & -2 \\ -1 & 2 & -1 \end{bmatrix}$$

Laws 1

Laws 2

Laws 3

$$\frac{1}{12} \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad \frac{1}{4} \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix} \quad \frac{1}{4} \begin{bmatrix} -1 & 2 & -1 \\ 0 & 0 & 0 \\ 1 & -2 & 1 \end{bmatrix}$$

Laws 4

Laws 5

Laws 6

$$\frac{1}{12} \begin{bmatrix} -1 & -2 & -1 \\ 2 & 4 & 2 \\ -1 & -2 & -1 \end{bmatrix} \quad \frac{1}{4} \begin{bmatrix} -1 & 0 & 1 \\ 2 & 0 & -2 \\ -1 & 0 & 1 \end{bmatrix} \quad \frac{1}{4} \begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

Laws 7

Laws 8

Laws 9

Laws' Filter Banks

- **Input & Output of Laws' filter bank**
 - Input: an image of $N \times N$ pixels
 - Output (w/o padding): a 3D tensor of dimension $(N-2) \times (N-2) \times 9$
- **Interpretation**
 - The response of each filter indicates the frequency components of a local neighborhood of size 3×3 (9 channels)
- **Limitations**
 - Filter coefficients are fixed (not adaptive to image contents)
 - Only one-stage transform (no information of mid- and long-range neighborhood)

New Transforms for Unsupervised Feature Learning

- **One-stage Transform**
 - Saab transform
 - Saab means “subspace approximation with adjusted bias”
 - Improved Laws’ filter banks
 - A variant of PCA
- **Multi-stage Transform**
 - channel-wise (c/w) Saab transform

Saab Transform

- Subspace decomposition

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

- DC subspace is spanned by constant-element vector d $(1, \dots, 1)$
- AC subspace is its orthogonal complement
- Conduct PCA on the AC subspace

Example of Saab Transform (1)

- **Gray-scale images: 3x3x1 Saab Transform**
 - 1 DC filter: constant-element filter (= mean of a 3x3 patch)
 - 8 AC filters (PCA analysis applied to AC components)
 - Covariance matrix of mean-removed 3x3 patches
 - The first 8 eigenvectors of the 9x9 covariance matrix
 - The last eigenvector has an eigenvalue close to 0
 - Output: $(N-2) \times (N-2) \times 9$ three-D tensor
 - Why not apply PCA to 3x3 patches directly
 - Need to subtract the ensemble mean of these 3x3 patches, which is sensitive to the image input
 - The ensemble mean of residuals approximates to a zero vector

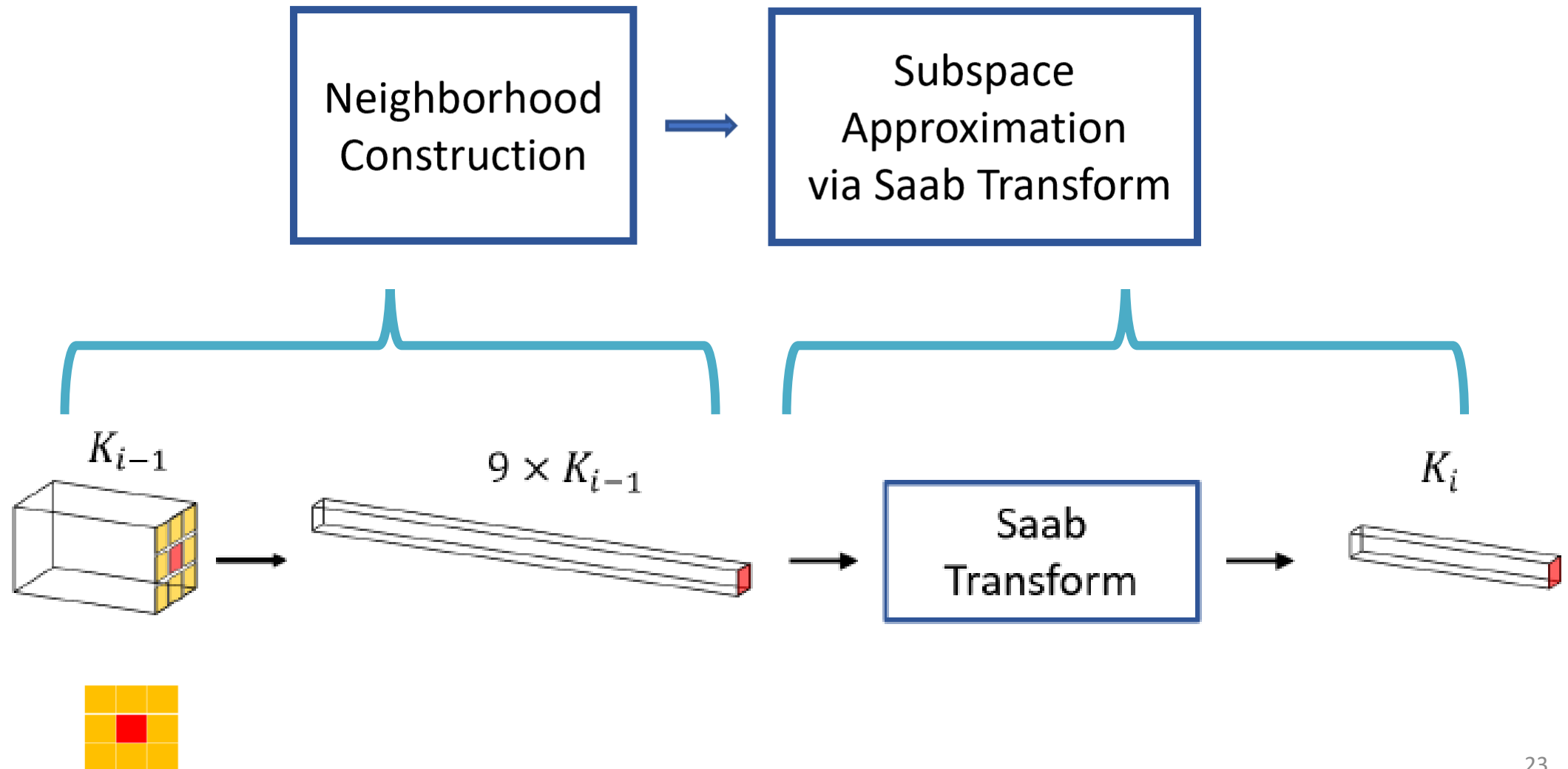
Example of Saab Transform (2)

- **Color images: 3x3x3 Saab Transform**
 - 1 DC filter: constant-element filter (= mean of a 3x3x3 patch)
 - 26 AC filters (PCA analysis applied to AC components)
 - Covariance matrix of mean-removed 3x3x3 patches
 - The first 26 eigenvectors of the 27x27 covariance matrix
 - The last eigenvector has an eigenvalue close to 0
- **Common filter sizes in spatial domain**
 - 2x2, 3x3, 4x4, 5x5, etc.
 - Should avoid the use of large filter sizes
 - Correlation between long-range pixels is weaker
 - The dimension of the output tensor would become too large

Lossless and Lossy Saab Transform

- **Example of Lossless Saab Transform**
 - Input: $N \times N$ (N even)
 - Filter size: 2×2
 - Stride: 2
 - Output: $(N/2) \times (N/2) \times 4$
- **Redundant Saab Transform**
 - The above setting but with stride = 1
 - Output: $(N-2) \times (N-2) \times 4$
 - Redundancy removal: (2×2) to (1×1) pooling
- **Lossy Saab Transform**
 - Discard channels with small responses – dimension reduction

Generalization to Multi-Stage Saab Transform



Correlation Analysis of Saab Coefficients

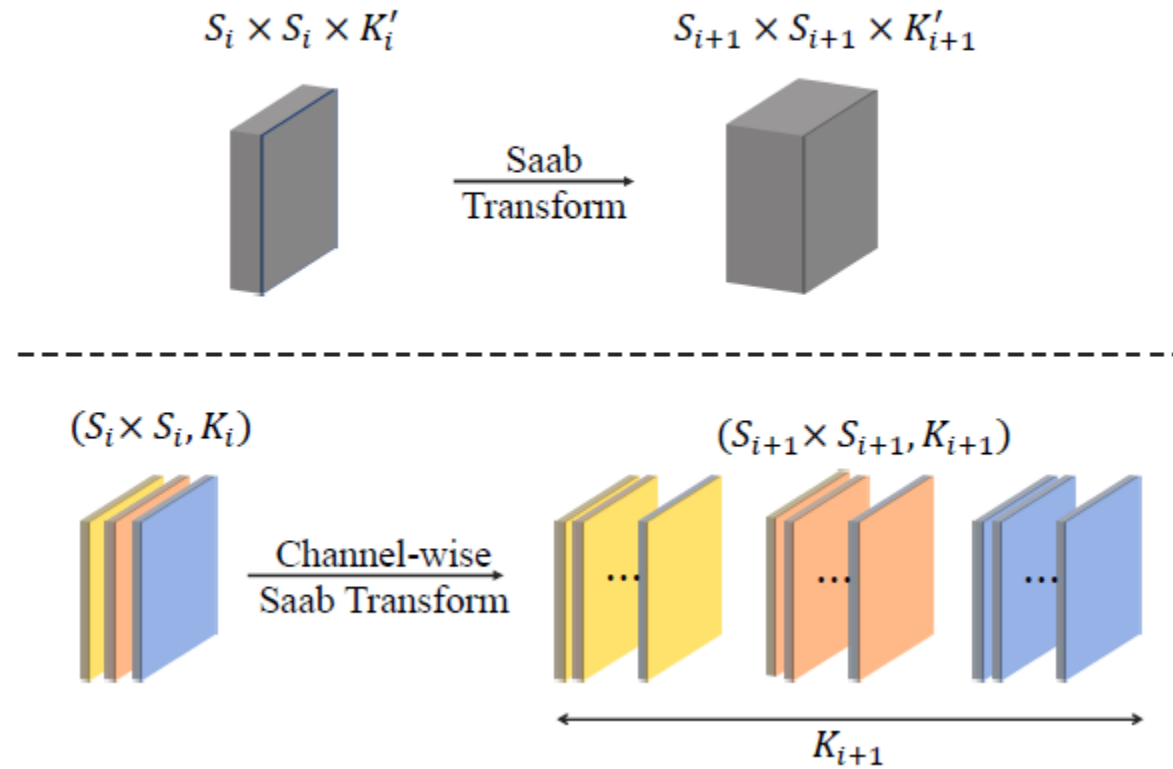


Table 1. Averaged correlations of filtered AC outputs from the first to the third Pixelhop units with respect to the MNIST, Fashion MNIST and CIFAR-10 datasets.

Dataset	MNIST	Fashion MNIST	CIFAR-10
Spatial 1	0.48 ± 0.05	0.51 ± 0.03	0.53 ± 0.03
Spatial 2	0.22 ± 0.03	0.29 ± 0.05	0.27 ± 0.06
Spectral 1	0.33 ± 0.07	0.12 ± 0.02	0.0156 ± 0.0005
Spectral 2	0.18 ± 0.02	0.13 ± 0.01	0.0188 ± 0.0004
Spectral 3	0.0099 ± 0.0001	0.0082 ± 0.0001	0.0079 ± 0.0004

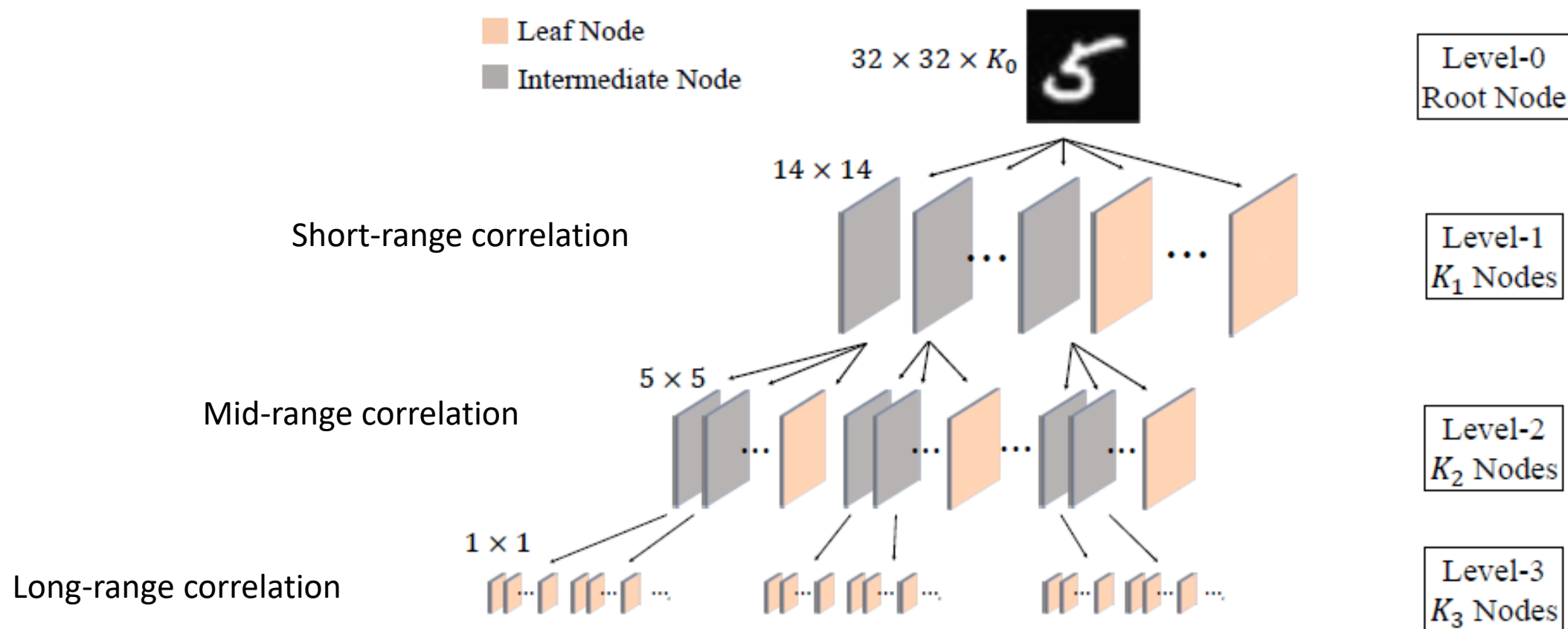
Comparison of Saab and c/w Saab Transforms

Saab Transform



Channel-wise (c/w) Saab Transform

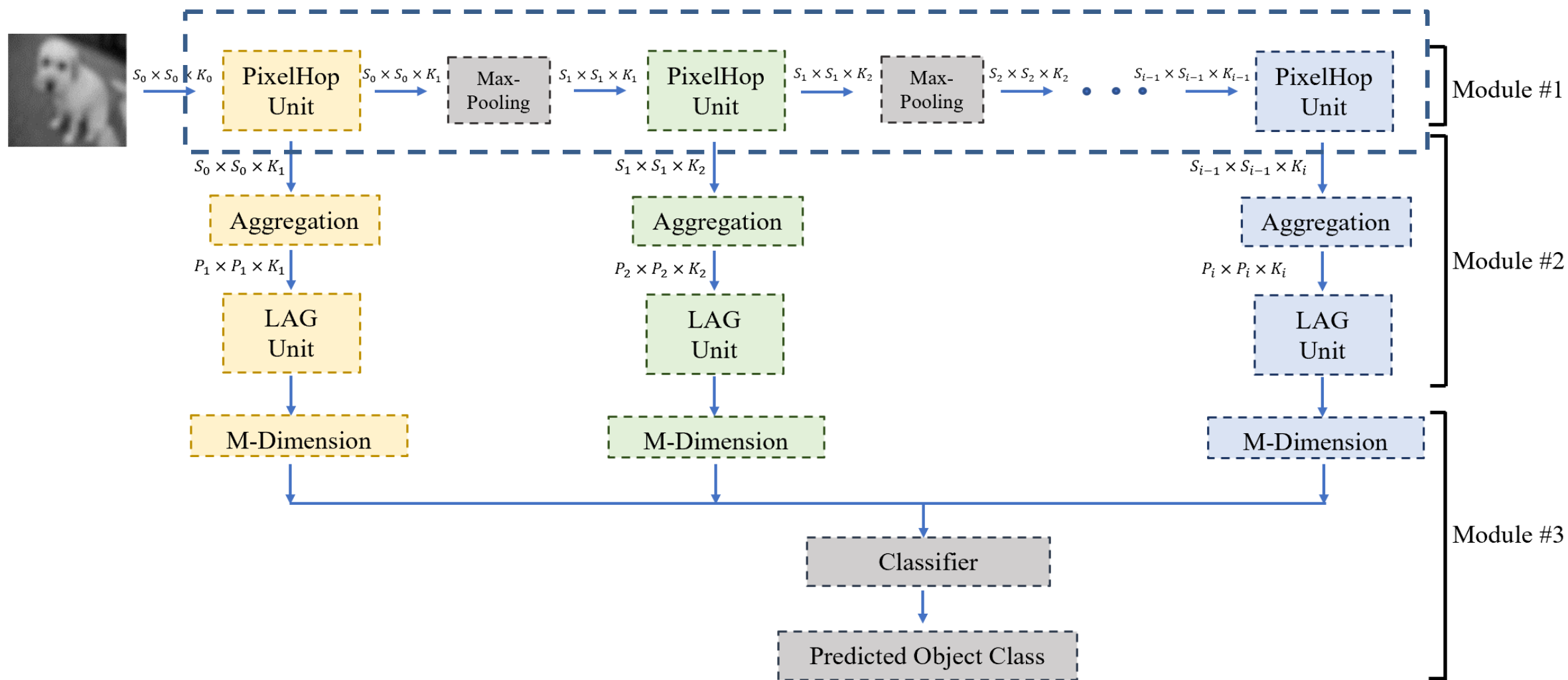
3-Hop c/w Saab Transform



Frequently Asked Questions

- **Is the Saab transform linear?**
 - Yes. More precisely, it is an affine transform.
- **Can a linear transform yield powerful features?**
 - Nonlinear classifiers are important
 - Linear features may not be that bad
 - Easy to understand
 - Clustering can increase the power of Saab features
 - Clustering can be done after (or before) the Saab transform

PixelHop



Green Learning for Image Classification

- Yueru Chen and C.-C. Jay Kuo, “PixelHop: a successive subspace learning (SSL) method for object classification,” the Journal of Visual Communications and Image Representation, Vol. 70, July 2020, 102749.
- Yueru Chen, Mozhdeh Rouhsedaghat, Suyu You, Raghuveer Rao and C.-C. Jay Kuo, “PixelHop++: A Small Successive-Subspace-Learning-Based (SSL-based) Model for Image Classification,” IEEE International Conference on Image Processing (ICIP), Dubai, United Arab Emirates, October 25-28, 2020.

Experiment Set-up

❖ Datasets:

➤ MNIST

- Handwritten digits 0-9
- Gray-scale images with size 32x32
- Training set: 60k, Testing set: 10k

➤ Fashion-MNIST

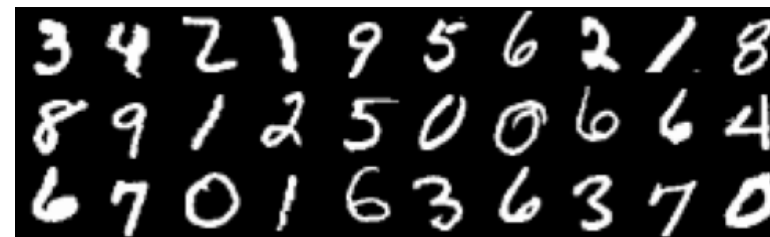
- Gray-scale fashion images with size 32×32
- Training set: 60k, Testing set: 10k

➤ CIFAR-10

- 10 classes of tiny RGB images with size 32×32
- Training set: 50k, Testing set: 10k

❖ Evaluation:

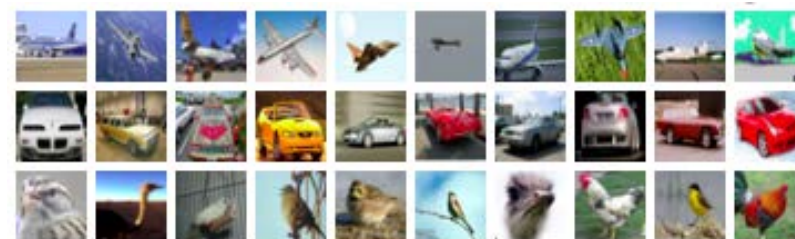
- Top-1 classification accuracy



MNIST



Fashion-MNIST



CIFAR-10

Performance Comparison

Table 8

Comparison of testing accuracy (%) of LeNet-5, feedforward-designed CNN (FF-CNN), PixelHop and PixelHop⁺ for MNIST, Fashion MNIST and CIFAR-10.

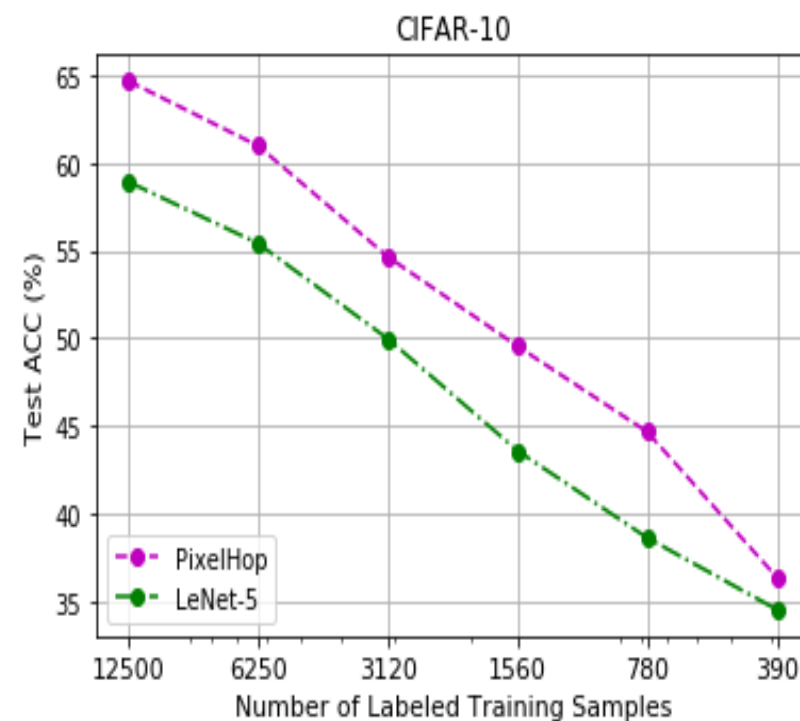
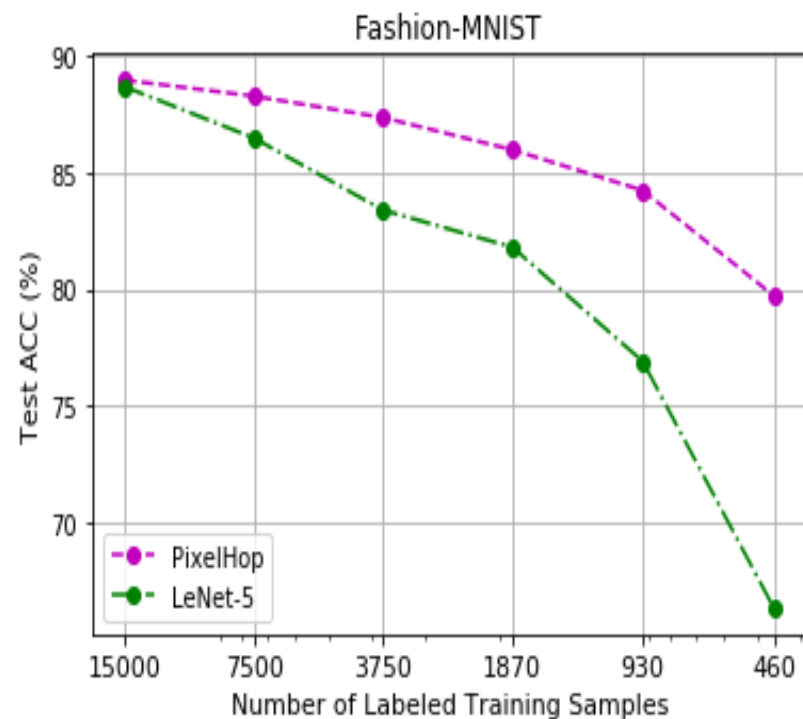
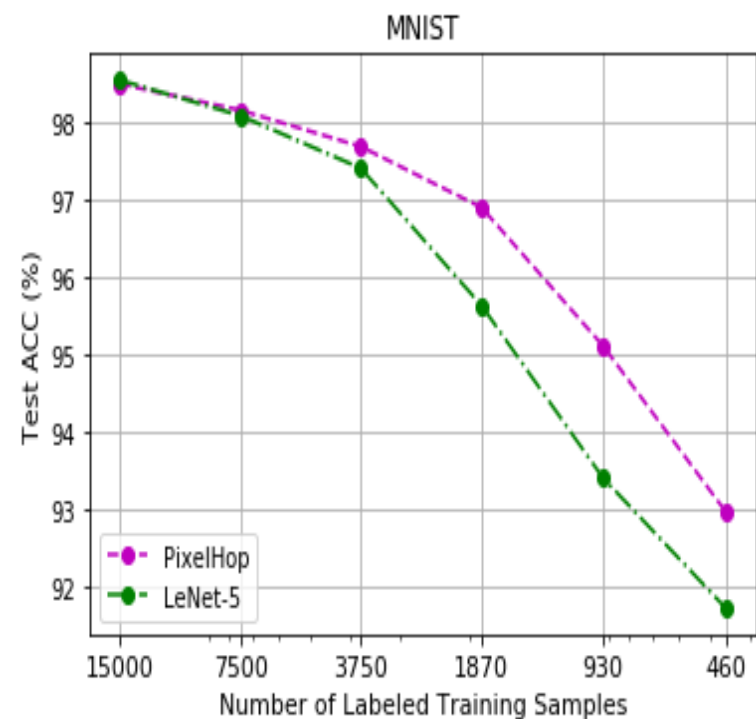
Method	MNIST	Fashion MNIST	CIFAR-10
LeNet-5	99.04	91.08	68.72
FF-CNN	97.52	86.90	62.13
PixelHop	98.90	91.30	71.37
PixelHop ⁺	99.09	91.68	72.66

Table 9

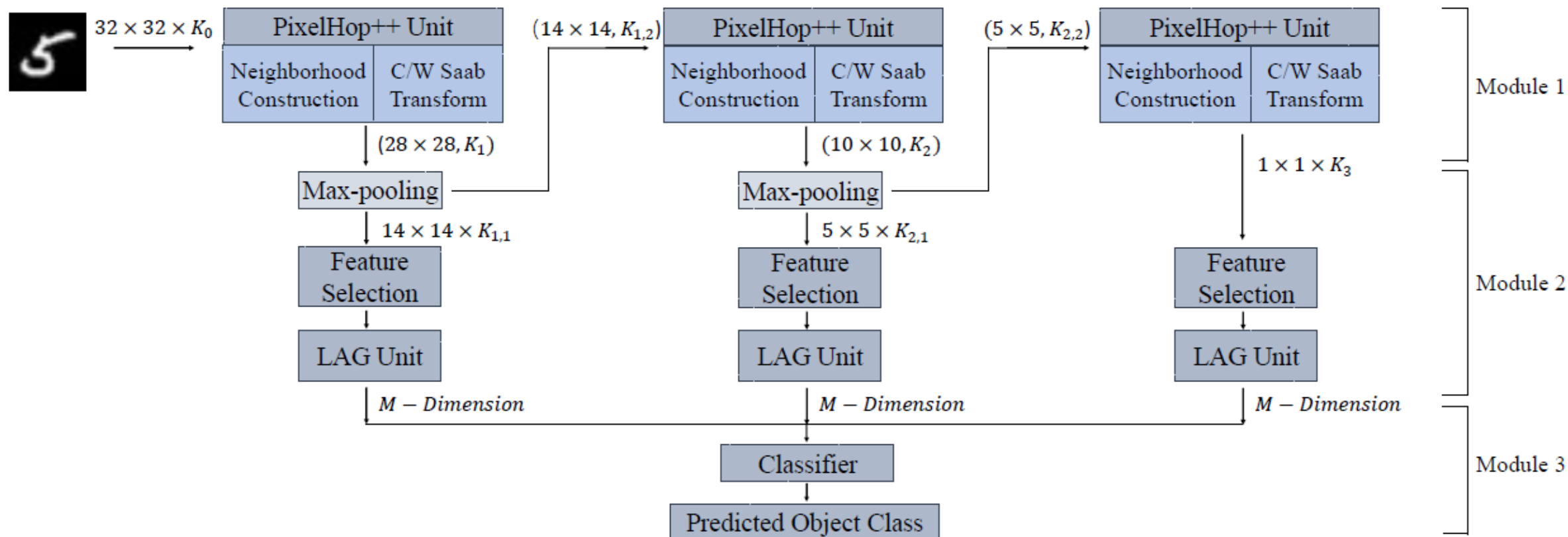
Comparison of training time of the LeNet-5 and the PixelHop method on the MNIST, the Fashion MNIST and the CIFAR-10 datasets.

Method	MNIST	Fashion MNIST	CIFAR-10
LeNet-5	~25 min	~25 min	~45 min
PixelHop	~15 min	~15 min	~30 min

Weak Supervision



PixelHop++



Model Size and Test Accuracy Comparison

Table 3. Comparison of test accuracy (%) of LeNet-5 and PixelHop++ for MNIST, Fashion MNIST and CIFAR-10.

Method	MNIST	Fashion MNIST	CIFAR-10
LeNet-5	99.04	89.74	68.72
PixelHop++ (Large)	98.49	90.17	66.81
PixelHop++ (Small)	97.98	88.84	64.75

Table 4. Comparison of the model size (in terms of the total parameter numbers) of LeNet-5 and PixelHop++ for the MNIST, the Fashion MNIST and the CIFAR-10 datasets.

Method	MNIST	Fashion MNIST	CIFAR-10
LeNet-5	61,706	194,558	395,006
PixelHop++ (Large)	111,981	127,186	115,623
PixelHop++ (Small)	29,514	33,017	62,150

Green Learning for Fake Image Detection

Hong-Shuo Chen, Mozhdeh Rousedaghat, Hamza Ghani, Shouwen Hu, Suyu You and C.-C. Jay Kuo,
“Defakehop: a light-weight high-performance deepfake detector,” IEEE International Conference on
Multimedia and Expo (ICME), Shenzhen, China, July 5-9, 2021.

Introduction

- **Deepfake videos** are synthetic media in which a person in a video is replaced with someone else
- **Deepfake videos** can be potentially harmful to society, from non-consensual explicit content creation to forged media by foreign adversaries used in disinformation campaigns
- As the number of Deepfake video contents grows rapidly, **an automatic and effective Deepfake detection** mechanism is in urgent need

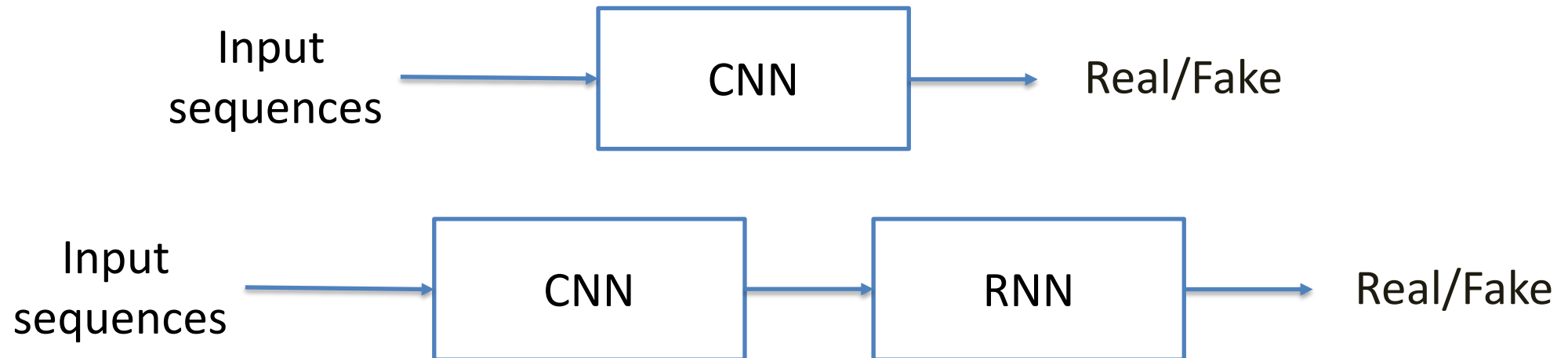


Original video

Fake video

Motivation

- Most state-of-the-art Deepfake detection methods are based upon **deep learning (DL)** technique
- They can be mainly categorized into two types
 - **convolutional neural networks (CNNs)**
 - integrate **CNNs** and **recurrent neural networks (RNNs)**



Motivation

- The **size** of DL-based methods is **large** -- containing hundreds of thousands or even millions of model parameters
- **Training** deep neural networks is **computationally expensive**
- There are also **non-DL-based Deepfake detection** methods, where handcrafted features are extracted and fed into classifiers
- The performance of non-DL-based methods is usually **inferior to that of DL-based ones**
- Our goal is to develop a **light-weight** non-DL-based methods and achieve a high-performance results

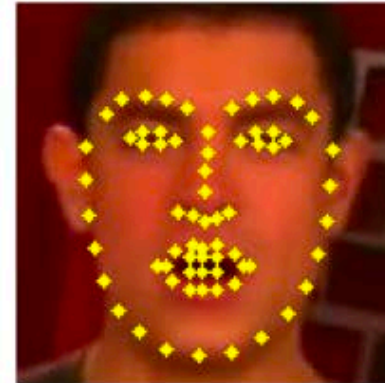
Face Preprocessing



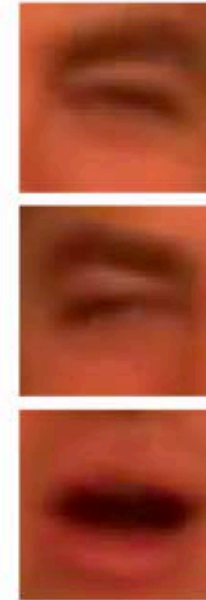
Sampling Frames



Landmarks Extraction

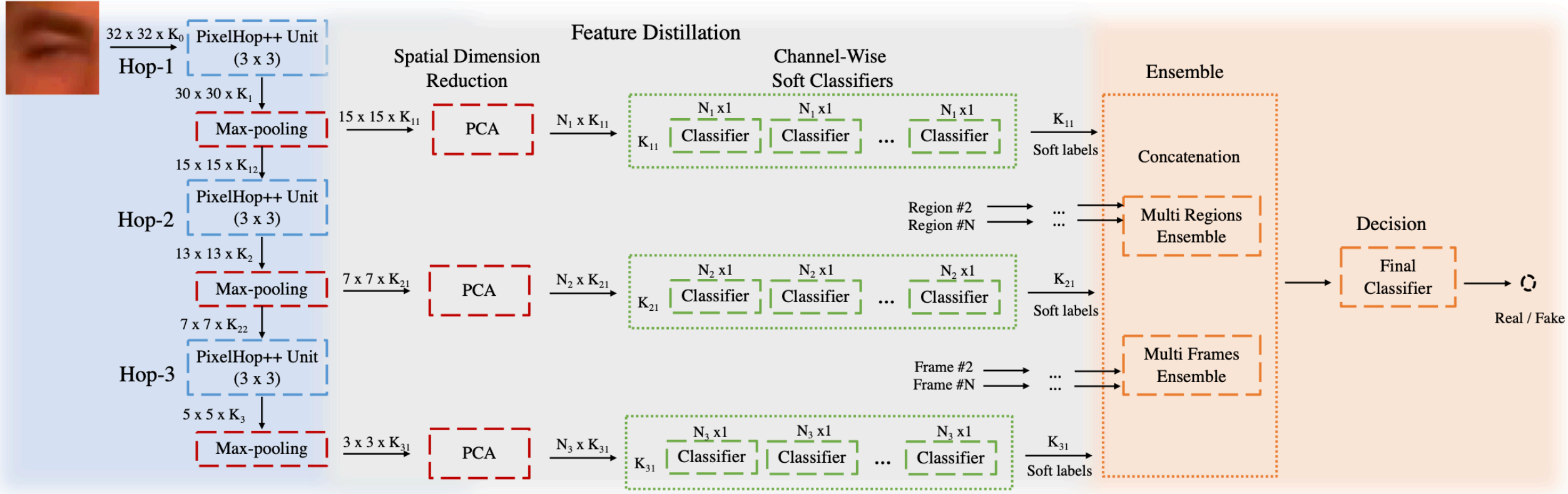


Face Alignment

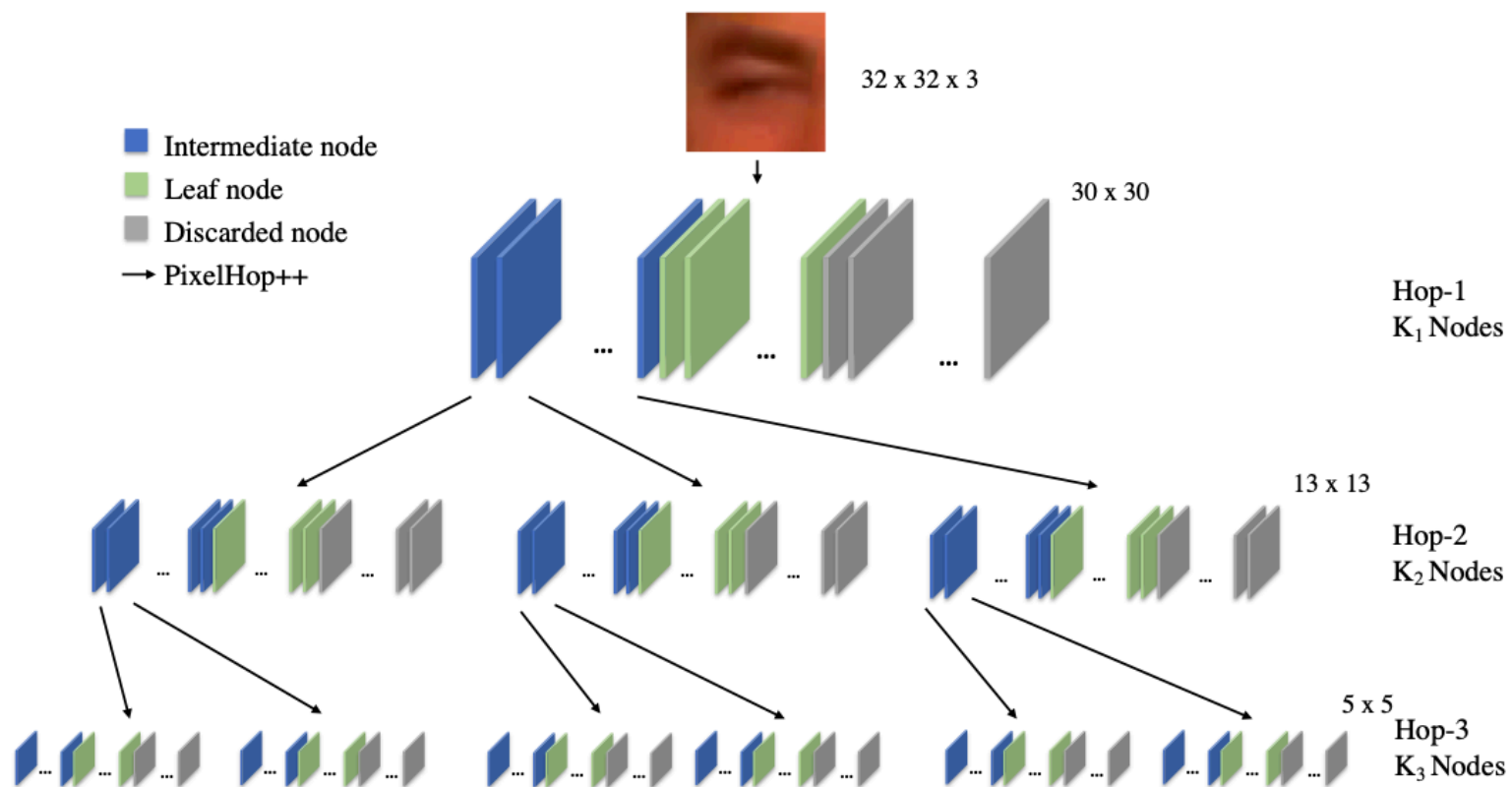


Regions Extraction

Defakehop Framework



Channel Wise Saab Transform



Model Size

Table 4. The number of parameters for various parts.

Subsystem	Number of Parameters
Pixelhop++ Hop-1	270
Pixelhop++ Hop-2	90
Pixelhop++ Hop-3	90
PCA Hop-1	10,125
PCA Hop-2	1,225
PCA Hop-3	45
Channel-Wise XGBoost(s)	12,000
Fianl XGBoost	19,000
Total	42,845

Datasets

- We use two datasets from 1st generation dataset and two datasets from 2nd generation dataset.
- The numbers of real, fake, train and test video for each dataset are shown.

Datasets	Real	Fake	Train	Test
UADFV	49	49	78	20
FaceForensics++	1000	1000	1440	280
Celeb-DF v1	408	795	1103	100
Celeb-DF v2	890	5639	6011	518

Experiments

Table 2. Comparison of the detection performance of benchmarking methods with the AUC value at the frame level as the evaluation metric. The **boldface** and the underbar indicate the best and the second-best results, respectively. The *italics* means it does not specify frame or video level AUC. The AUC results of DefakeHop is reported in both frame-level and video-level. The AUC results of benchmarking methods are taken from [19] and [20]. ^a deep learning method, ^b non deep learning method.

		1st Generation datasets		2nd Generation datasets		
	Method	UADFV	FF++ / DF	Celeb-DF v1	Celeb-DF v2	Number of parameters
Zhou <i>et al.</i> .(2017) [3]	InceptionV3 ^a	85.1%	70.1%	55.7%	53.8%	24M
Afchar <i>et al.</i> .(2018) [4]	Meso4 ^a	84.3%	84.7%	53.6%	54.8%	27.9K
Li <i>et al.</i> .(2018) [17]	FWA ^a (ResNet-50)	97.4%	80.1	53.8%	56.9%	23.8M
Yang <i>et al.</i> .(2019) [9]	HeadPose ^b (SVM)	89%	47.3%	54.8%	54.6%	-
Matern <i>et al.</i> .(2019) [11]	VA-MLP ^b	70.2%	66.4%	48.8%	55%	-
Rossler <i>et al.</i> .(2019) [2]	Xception-raw ^a	80.4%	99.7%	38.7%	48.2%	22.8M
Nguyen <i>et al.</i> .(2019) [5]	Multi-task ^a	65.8%	76.3%	36.5%	54.3%	-
Nguyen <i>et al.</i> .(2019) [6]	CapsuleNet ^a	61.3%	96.6%	-	57.5%	3.9M
Sabir <i>et al.</i> .(2019) [8]	<i>DenseNet+RNN</i> ^a	-	<u>99.6%</u>	-	-	25.6M
Li <i>et al.</i> .(2020) [17]	DSP-FWA ^a (SPPNet)	<u>97.7%</u>	93%	-	64.6%	-
Tolosana <i>et al.</i> .(2020) [1]	<i>Xception</i> ^a	100%	99.4%	83.6%	-	22.8M
Ours	DefakeHop (Frame)	100%	95.95%	<u>93.12%</u>	<u>87.65%</u>	42.8K
	DefakeHop (Video)	100%	97.45%	94.95%	90.56%	42.8K

Experiments

- The ensemble of multiple facial regions can boost the AUC values by up to 5%. Each facial region has different strengths on various faces, and their ensemble gives the best result
- The performance of DefakeHop degrades by 5% as video quality becomes worse

Table 1. The AUC value for each facial region and the final ensemble result.

	Left eye	Right eye	Mouth	Ensemble
UADFV	100%	100%	100%	100%
FF++ / DF	94.37%	93.73%	94.25%	97.45%
Celeb-DF v1	89.69%	88.20%	92.66%	94.95%
Celeb-DF v2	85.17%	86.41%	89.66%	90.56%

Table 3. Comparison of Deepfake algorithms and qualities.

	FF++ with Deepfakes		FF++ with FaceSwap	
	HQ (c23)	LQ (c40)	HQ (c23)	LQ (c40)
Frame	95.95%	93.01%	97.87%	89.14%
Video	97.45%	95.80%	98.78%	93.22%

Experiments

- DefakeHop can achieve about 85% AUC with less than 5% (250 videos) of the whole training data.

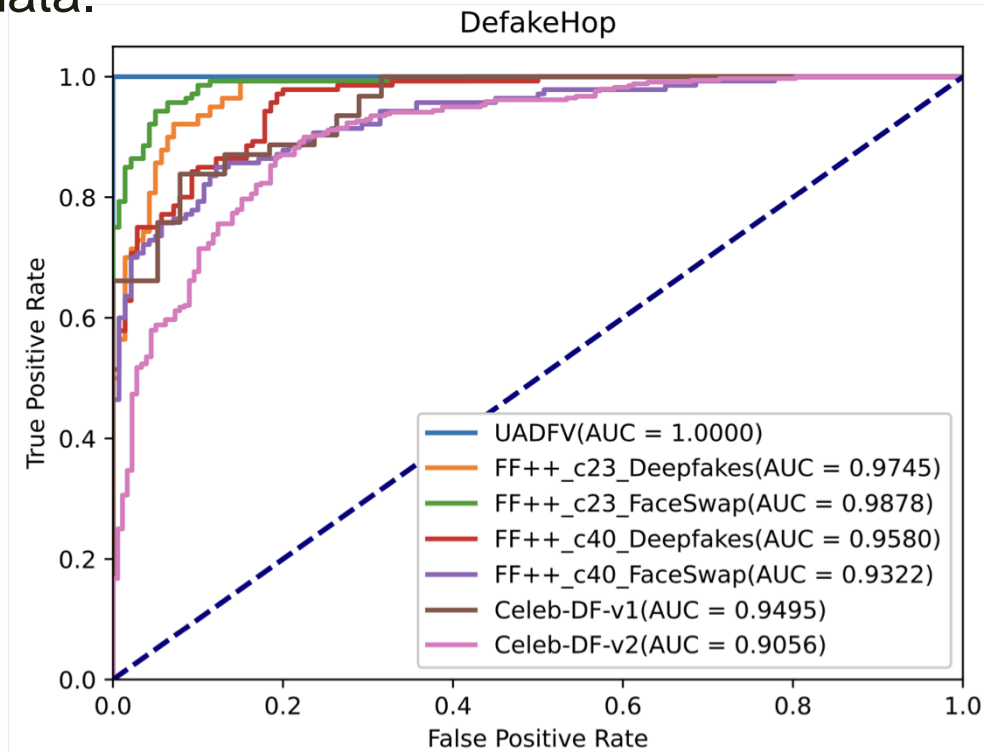


Fig. 4. The ROC curve of DefakeHop for different datasets.

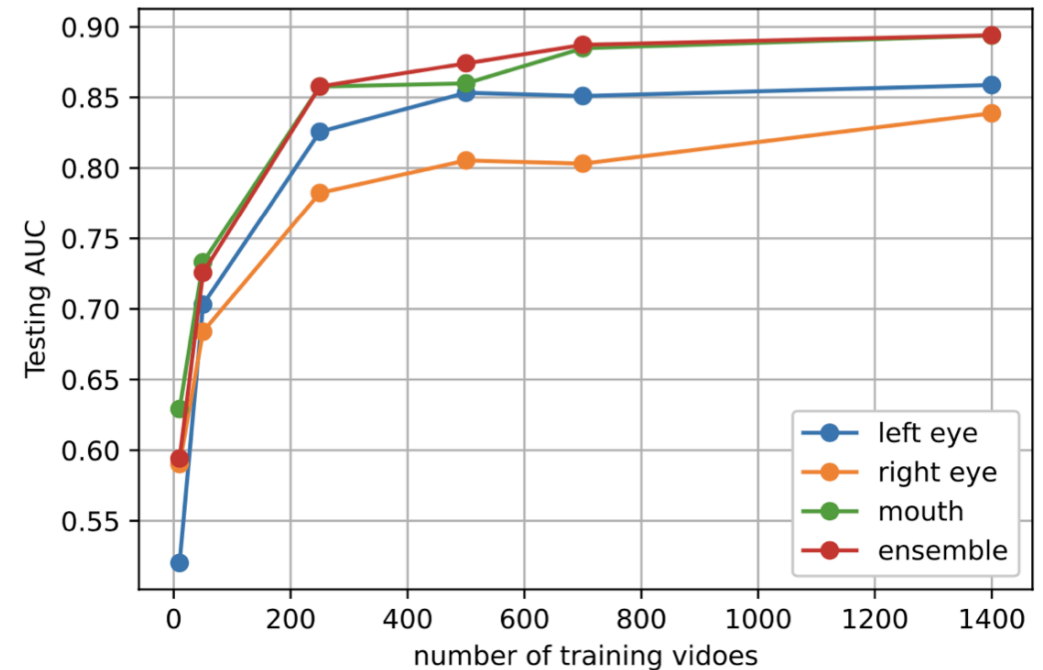



Fig. 5. The plot of AUC values as a function of the training video number.



2nd generation dataset: Celeb-DF

Our codes are released in GitHub!

 [hongshuochen / DefakeHop](#)

Unwatch1

Unstar14

Fork6

<> Code

Issues

Pull requests

Actions

Projects

Wiki


Security

Insights

Settings

master1 branch0 tags

Go to fileAdd fileCode

 **hongshuochen** Update model.py ... 74338b7 on Mar 29 18 commits

data/UADFV	Delete .DS_Store	2 months ago
img	add images and data	2 months ago
preprocessing	add preprocessing	2 months ago
README.md	Update README.md	2 months ago
defakeHop.py	first version	2 months ago
model.py	Update model.py	2 months ago
multi_cwSaab.py	add images and data	2 months ago
saab.py	first version	2 months ago
utils.py	first version	2 months ago

About

Official code for DefakeHop: A Light-Weight High-Performance Deepfake Detector

[arxiv.org/abs/2103.06929](#)

[deepfake-detection](#)

[successive-subspace-learning](#)

[green-learning](#)

Readme

Releases

No releases published

[Create a new release](#)

Summary

- Defakehop has several advantages
 - a smaller model size
 - fast training procedure
 - high detection AUC
 - needs fewer training samples
- Extensive experiments were conducted to demonstrate its high detection performance

Green Learning for Point Cloud Classification and Registration

- Min Zhang, Haoxuan You, Pranav Kadam, Shan Liu and C.-C. Jay Kuo, "PointHop: an explainable machine learning method for point cloud classification," IEEE Trans. on Multimedia, Vol. 22, No. 7, pp. 1744-1755, July 2020.
- Min Zhang, Yifan Wang, Pranav Kadam, Shan Liu and C.-C. Jay Kuo, "PointHop++: A Lightweight Learning Model on Point Sets for 3D Classification." IEEE International Conference on Image Processing (ICIP), Dubai, United Arab Emirates, October 25-28, 2020.
- Min Zhang, Pranav Kadam, Shan Liu and C.-C. Jay Kuo, "Unsupervised feedforward feature (UFF) learning for point cloud classification and segmentation," IEEE International Conferences on Visual Communications and Image Processing (VCIP), Macau, Dec. 1-4, 2020.
- Pranav Kadam, Min Zhang, Shan Liu, and C-C. Jay Kuo. "R-PointHop: A Green, Accurate and Unsupervised Point Cloud Registration Method." arXiv preprint arXiv:2103.08129 (2021).

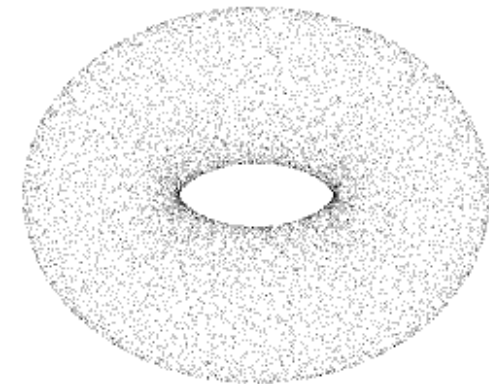
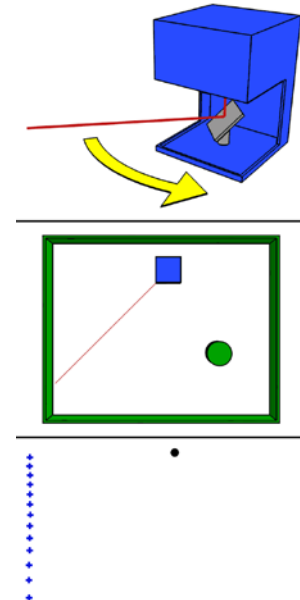
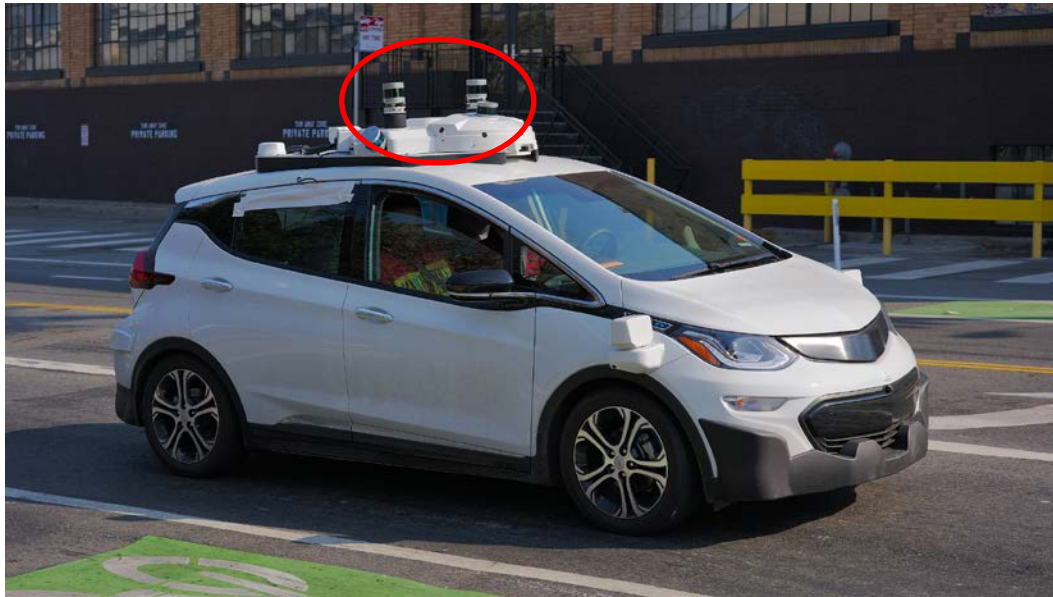


Background

What: A point cloud is a set of points in the 3D space

How: 3D scanning devices such as Lidar, measured by time of flight (ToF)

Why? With reduced cost of sensors, point cloud processing has become popular



Source: Wikipedia



Datasets and Performance Metrics

ModelNet-40

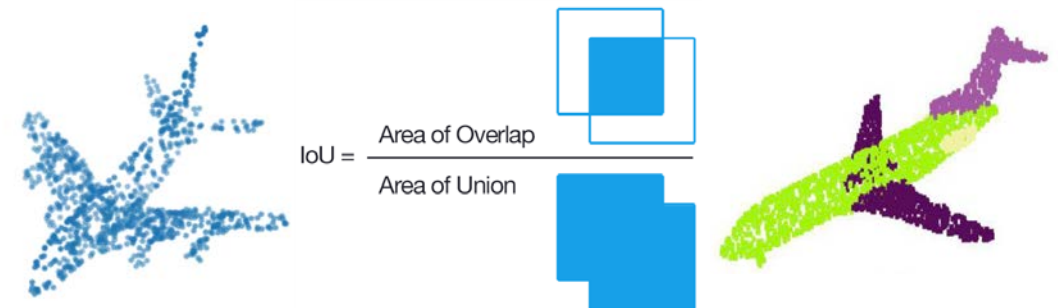
- 40 categories of objects (e.g., airplane, table, desk, sofa)
- Each object has 2048 points

ShapeNet Part

- 16 object categories
- 50 parts: each object is annotated with two to six parts
- Each shape has 2048 points

Evaluation metric:

- Classification - accuracy
- Segmentation – Intersection over Union (IoU)
- Registration - Mean Square Error (MSE)



Two Topics

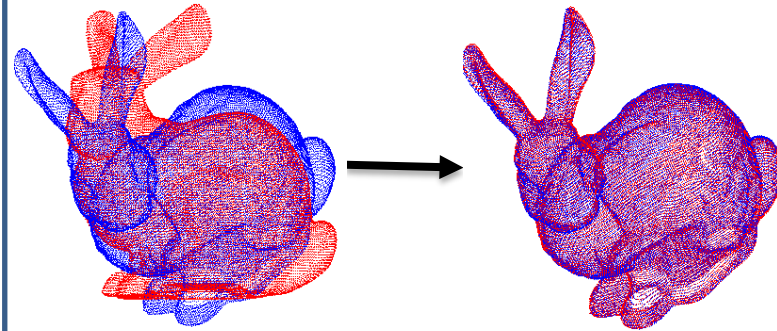


Classification:
label each object



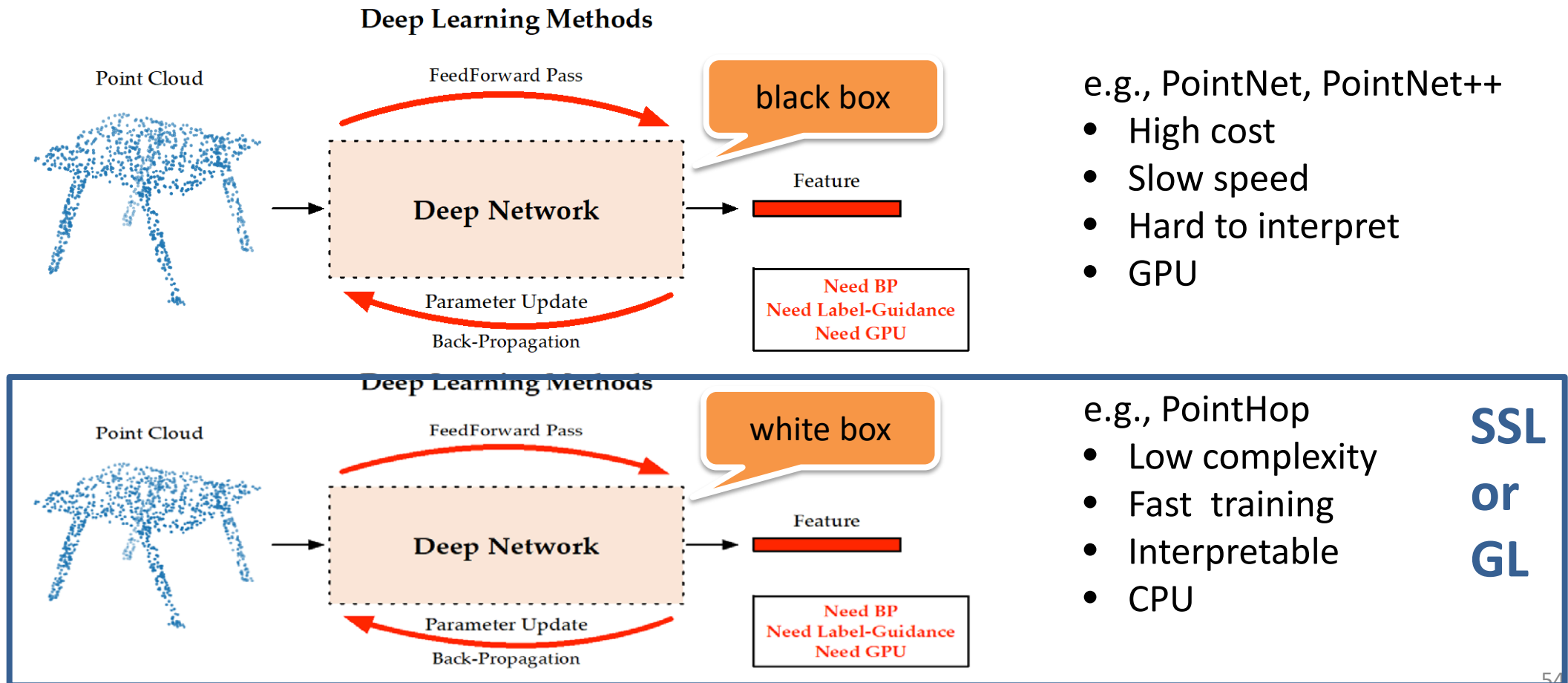
→ Chair

Registration:
align two point clouds





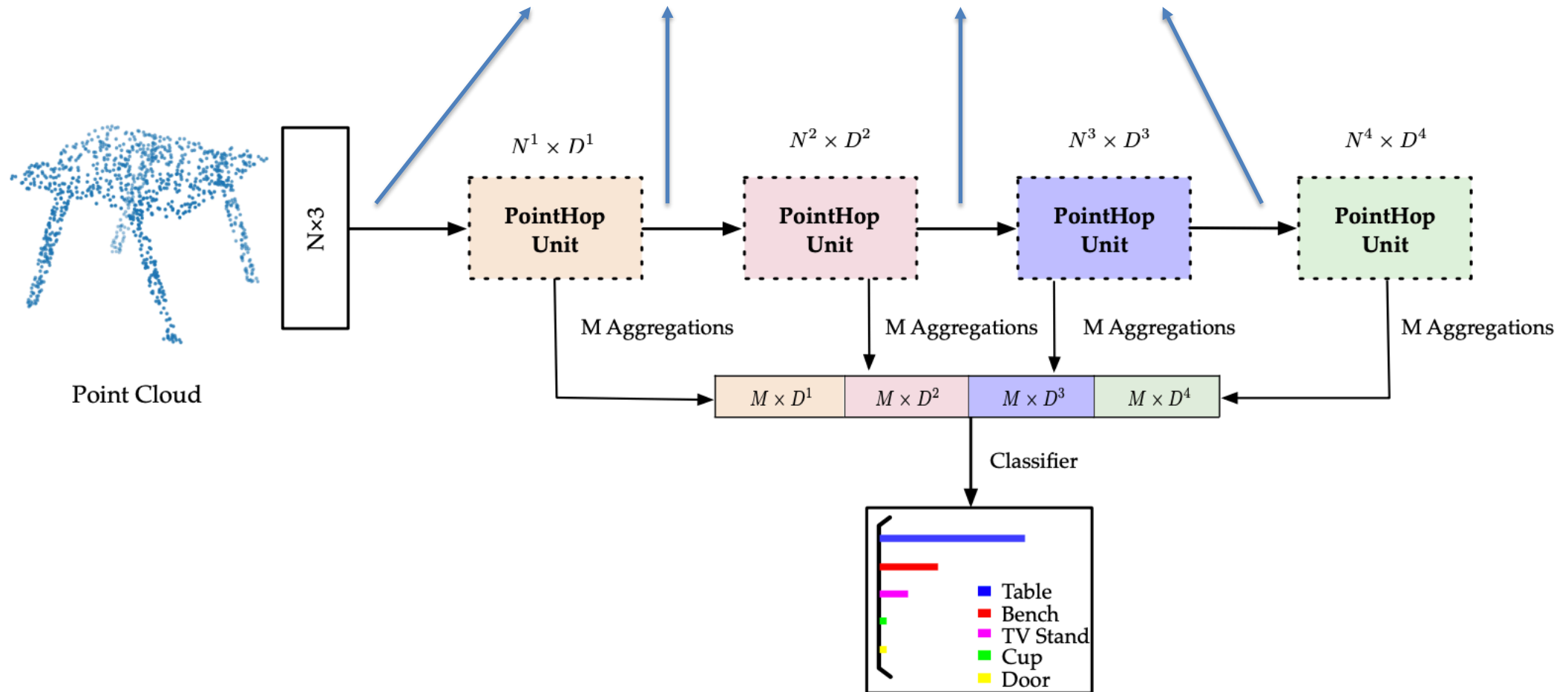
PointHop – A Successive-Subspace-Learning-based (SSL-based) or Green Learning (GL) method





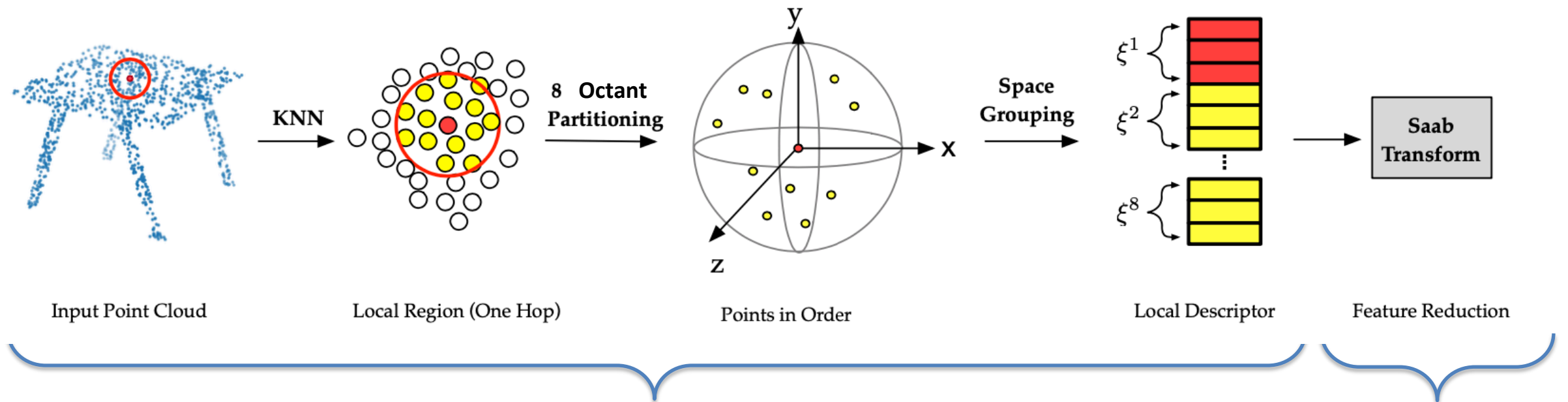
PointHop

- cascade of PointHop units + classifier
- spatial sampling scheme: farthest point sampling (FPS)
- reduce computational complexity
 - speed up the coverage rate





PointHop Unit



Constructing a local descriptor with attributes of one-hop neighbors

- Solve both unordered and disturbance problem
- The attributes of a point grow from a low dimension one to a high dimension one

Using the Saab transform to reduce the dimension of the local descriptor

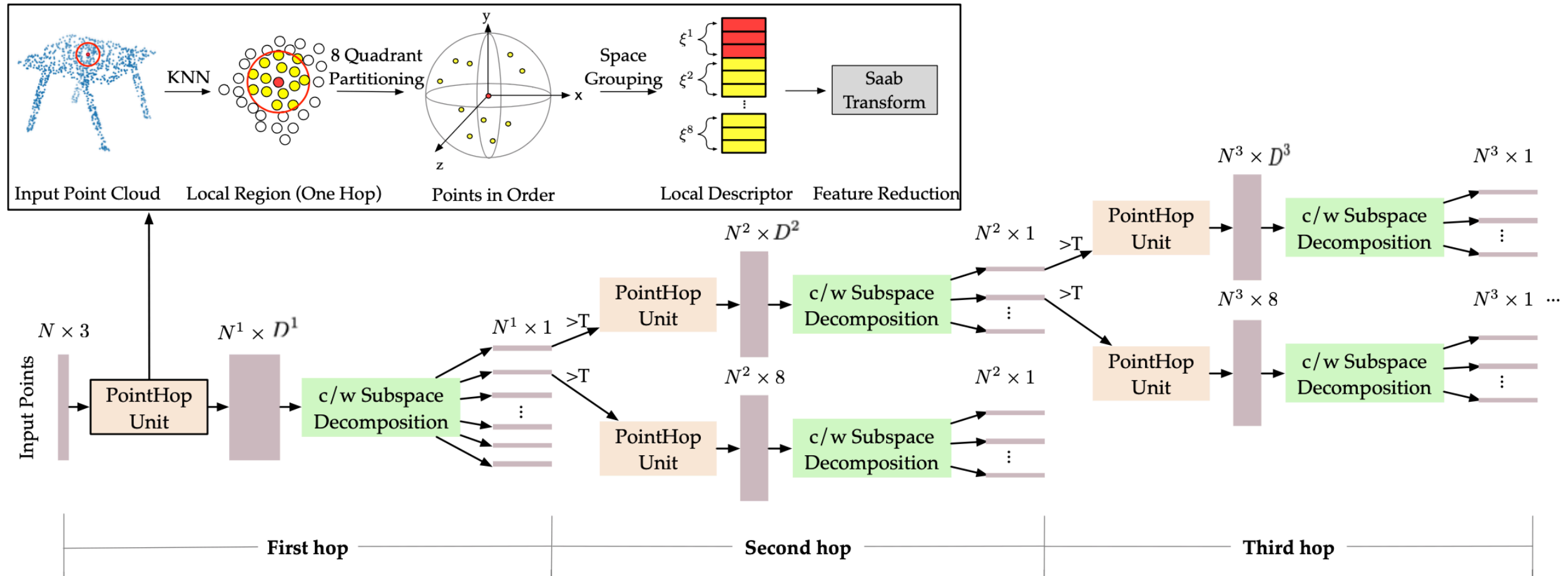
- the dimension grows at a slower rate



PointHop++

New features:

- Reduce model complexity – c/w Saab transform
- Order discriminant features automatically based on the cross-entropy criterion





Performance Evaluation for ModelNet-40: Accuracy and Sparser Models

	Method	Accuracy (%)	
		class-avg	overall
Supervised	PointNet [10]	86.2	89.2
	PointNet++ [11]	-	90.7
	PointCNN [12]	88.1	92.2
	DGCNN [13]	90.2	92.2
Unsupervised	LFD-GAN [28]	-	85.7
	FoldingNet [29]	-	88.4
	PointHop [15]	84.4	89.1
	PointHop++ (baseline)	85.6	90.3
	PointHop++ (FS)	86.5	90.8
	PointHop++ (FS+ES)	87	91.1



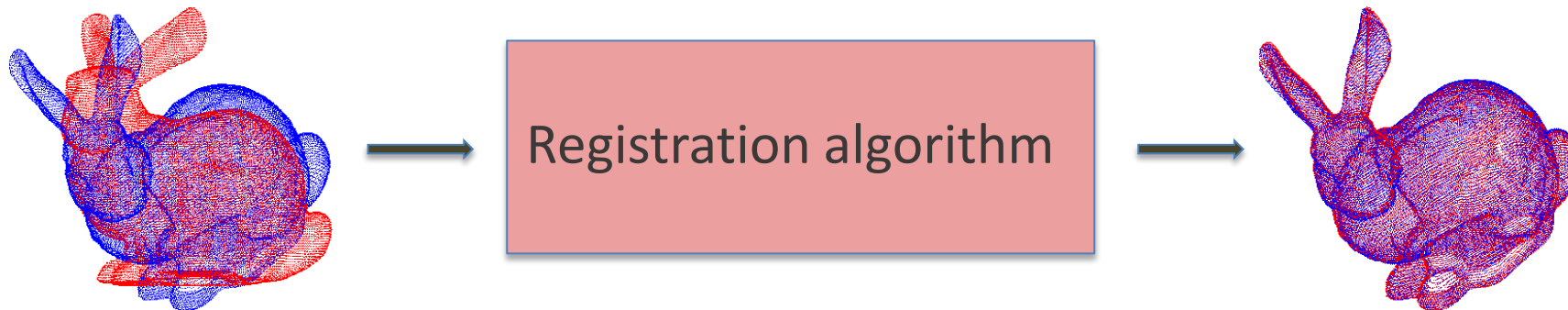
Performance Evaluation for ModelNet-40: Complexity and Model Size

	Method	Time		Parameter No. (MB)		
		Training	Inference	Filter	Classifier	Total
GPU	PointNet [10]	7	10	-	-	3.48
	PointNet++ [11]	7	14	-	-	1.48
	DGCNN [13]	21	154	-	-	1.84
CPU	PointHop [15]	0.33	108	0.037	-	-
	PointHop++	0.42	97	0.009	0.15	0.159

(Hours) (ms)

POINT CLOUD REGISTRATION

- Registration is the process of finding a **spatial transformation** that optimally **aligns** two point clouds
- Register point clouds to merge multiple point cloud scans to get a globally consistent view
- Registration acts as a pre-processing step before other tasks



PROBLEM STATEMENT

- We are interested in finding a rigid transformation (**rotation and translation**) that optimally registers the two point clouds

- Input** – 1. target point cloud ($N_t \times 3$)
2. source point cloud ($N_s \times 3$)

$$\text{source} = R * \text{target} + t$$

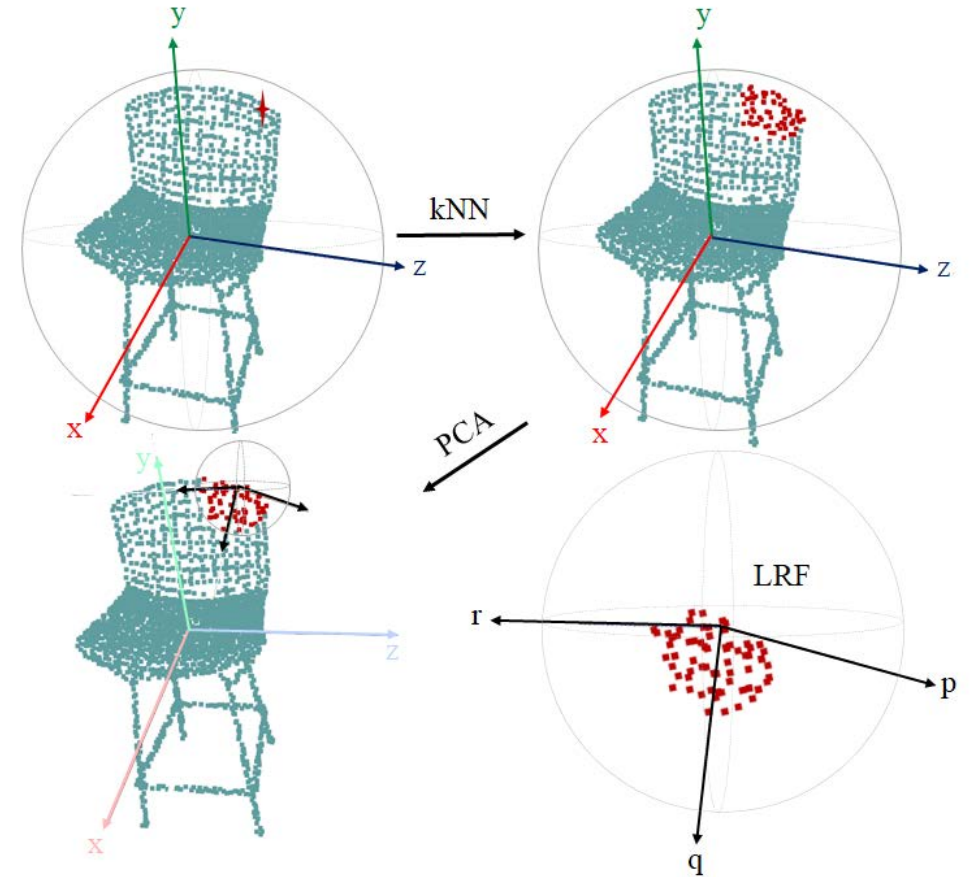
- Goal** – Align source to target ($N_s \times 3 \rightarrow N_s \times 3$)
- Output** – 3D Rotation matrix (R) and translation vector (t) that minimizes point-wise MSE

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha_x & -\sin \alpha_x \\ 0 & \sin \alpha_x & \cos \alpha_x \end{bmatrix} \begin{bmatrix} \cos \alpha_y & 0 & -\sin \alpha_y \\ 0 & 1 & 0 \\ \sin \alpha_y & 0 & \cos \alpha_y \end{bmatrix} \begin{bmatrix} \cos \alpha_z & -\sin \alpha_z & 0 \\ \sin \alpha_z & \cos \alpha_z & 0 \\ 0 & 0 & 1 \end{bmatrix}, t = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$



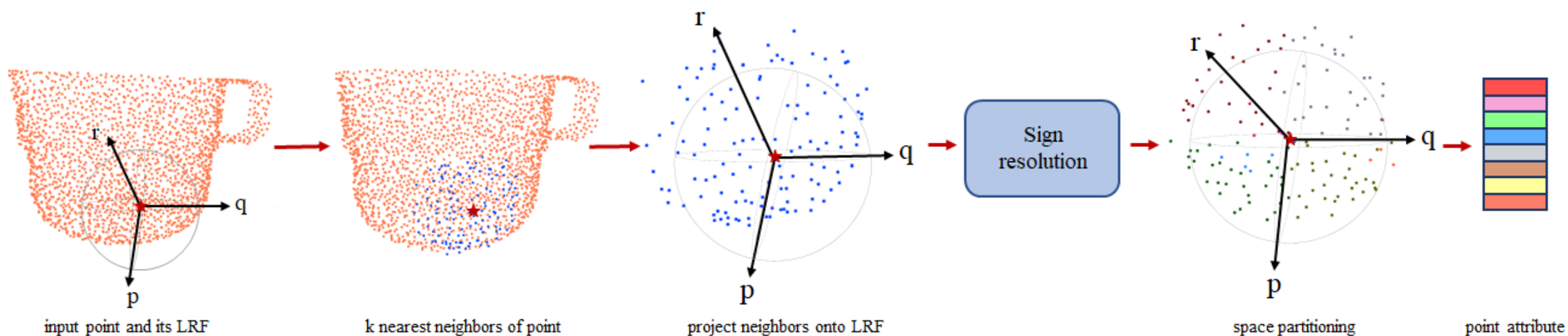
LOCAL REFERENCE FRAME (LRF)

- Consider a local patch around every point – find K nearest points
- Take a PCA of the $K \times 3$ data matrix
- The three principal components gives orthogonal axes ranked in order of decreasing variance
- These axes are invariant under any rigid transformation (rotation and translation)



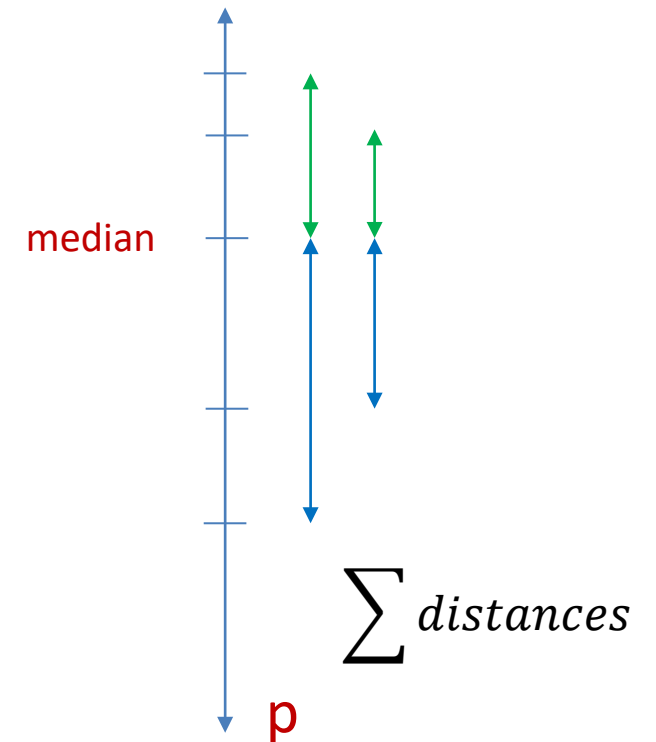
POINT ATTRIBUTE CONSTRUCTION

- Find K nearest neighbors of a point
- Project neighbors onto local reference frame (LRF) – XYZ coordinates to local coordinates
- Divide 3D space into 8 octants based on local coordinates – space partitioning
- Compute mean of points in each octant and concatenate 8 means, total 24D attribute



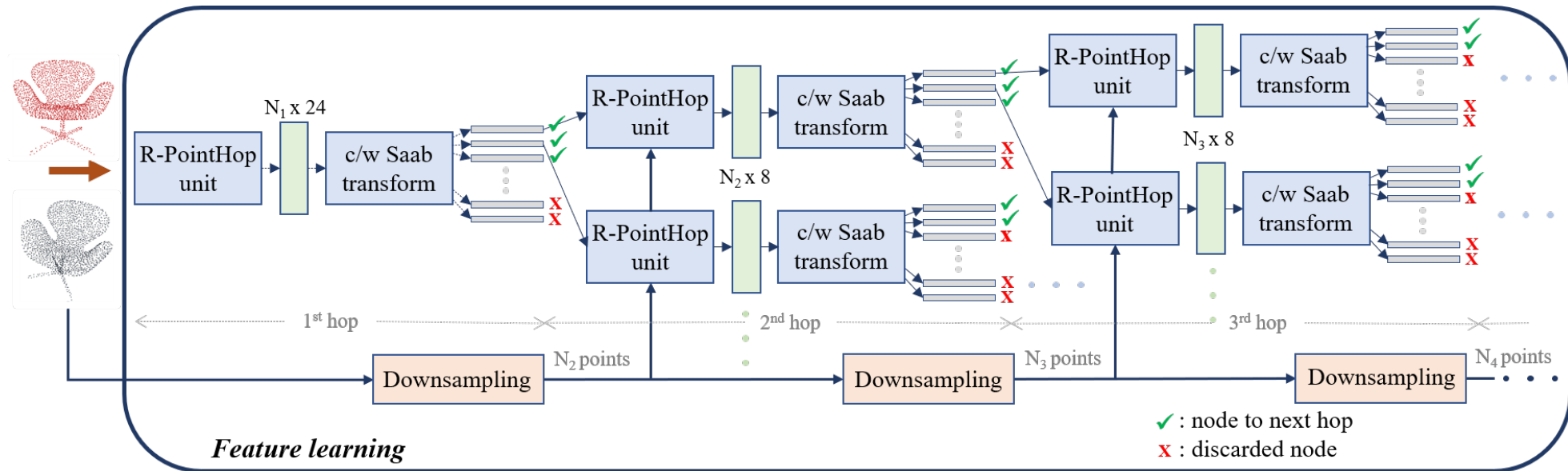
SIGN RESOLUTION

- Every eigen vector comes with a sign ambiguity of +/-
- Choice of sign affects space partitioning, hence a consistency is desired
- **Solution** – project the neighboring points onto a principal axis to get 1D local coordinates
- Order points in ascending order and calculate moment about median point
- If left moment > right moment, flip the sign
- This can be formulated as multiplying local coordinates with reflection matrix whose diagonal entries are 1 / -1



MULTI-HOP FEATURE LEARNING

1. Channel-wise Saab transform – discard nodes with energy less than threshold
2. Downsample the point cloud
3. Repeat attribute building step (R-PointHop unit)
4. Collect all leaf nodes at end of fourth hop (point feature)

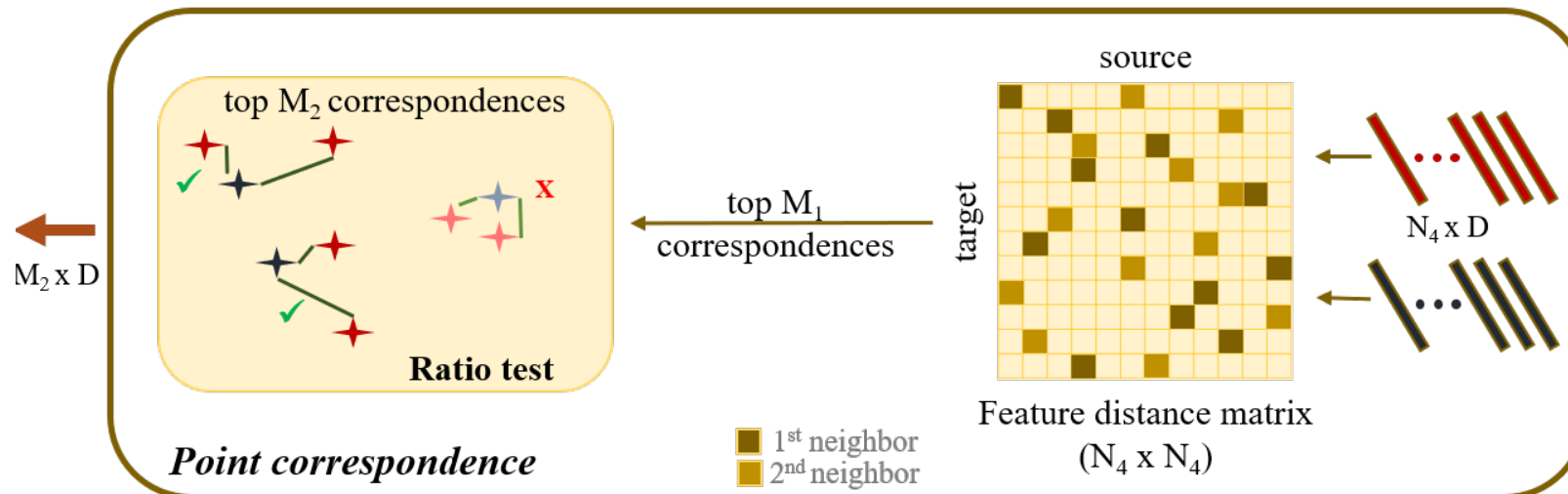


FEATURE LEARNING

1. Use of LRF makes features invariant to rotation and translation – robust correspondence
2. Hierarchical multi-hop approach helps learn short-, mid- and long-range point relations
3. No class label, no pairs of point clouds with ground truth transformations to learn Saab kernels
4. One pass feedforward
5. Independent of correspondence and transformation estimation modules

POINT CORRESPONDENCES

- For every point in the source, find its matching point in target using nearest neighbor in feature space
- Select a subset of good correspondences (**NEW!!**)
 - Smaller l2 distance in feature space
 - Smaller ratio of distance to first neighbor by distance to second neighbor



POINT CORRESPONDENCES

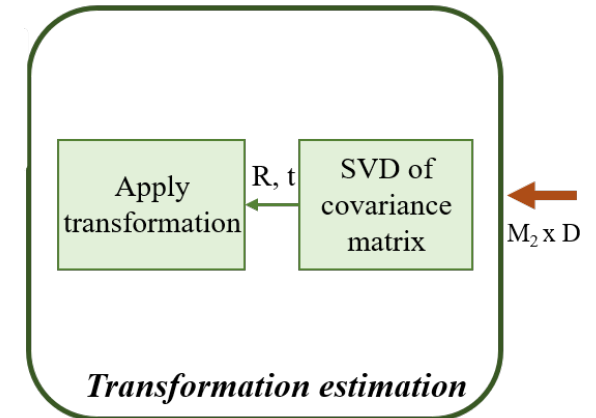
- Rich feature information to filter out correspondences against use of spatial information in SPA
- Favors partial point cloud registration –
 1. overlapping points have smaller l_2 distance in feature space
 2. Cannot comment from their spatial coordinates



TRANSFORMATION ESTIMATION – SVD

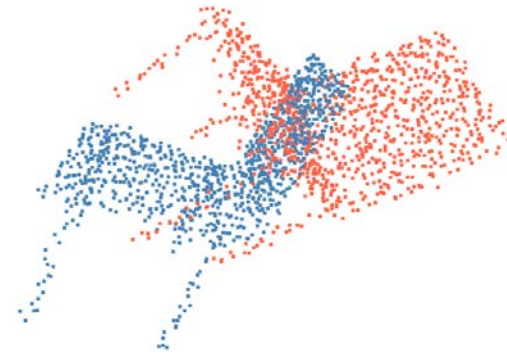
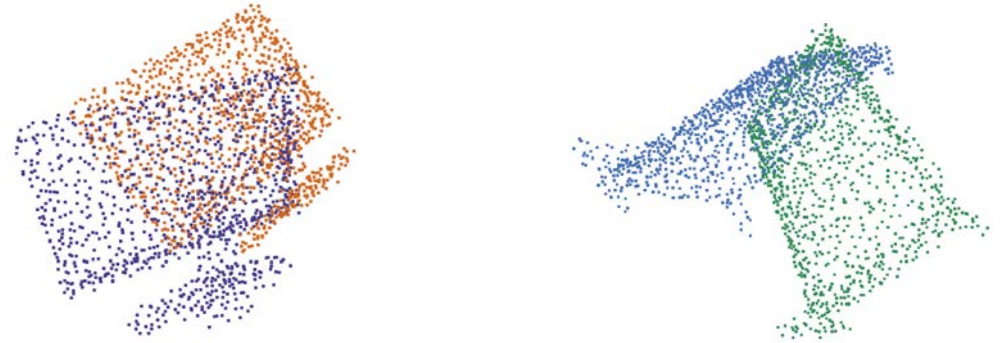
- Given corresponding points (f_i, g_i) solve the **orthogonal procrustes problem**

- Find means $\bar{f} = \frac{1}{N} \sum_{i=1}^N f_i$ and $\bar{g} = \frac{1}{N} \sum_{i=1}^N g_i$
- Compute covariance matrix $Cov(F, G) = \sum_{i=1}^N (f_i - \bar{f})(g_i - \bar{g})^T$
- Find SVD of covariance matrix $Cov(F, G) = USV^T$
- Optimal R and t are given by $R^* = VU^T$ and $t^* = -R^*\bar{f} + \bar{g}$



MODELNET40 DATASET

- 40 classes of CAD models
- 2048 points in each point cloud
- 9840 training samples



EXPERIMENTAL SETUP

- **For training** – use the target point clouds (without rotations)
- **For testing** – apply a random rotation and translation along three coordinate axes to get source
- **Rotation** – uniformly sample rotation angle in $[0, 45^\circ]$
- **Translation** – uniformly sample in $[-0.5, 0.5]$
- **Comparisons with** – ICP, Go-ICP, FGR (model free)
PointNetLK, Deep Closest Point (DCP), PR-Net (deep learning)
Salient Points Analysis (SPA)
- **Evaluation metric** – MSE, RMSE, MAE between ground truth and predicted angles

TEST ON UNSEEN OBJECTS AND UNSEEN CLASSES

- Best performance among benchmark methods
- Moreover, R-PointHop can better generalize on unseen classes than DCP and PointNetLK

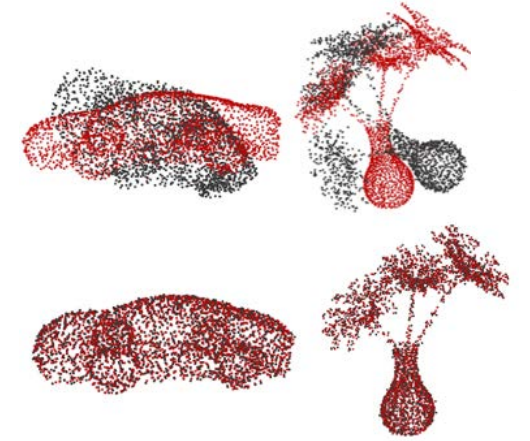


TABLE I
REGISTRATION ON UNSEEN POINT CLOUDS

Method	MSE (R)	RMSE (R)	MAE (R)	MSE (t)	RMSE (t)	MAE (t)
ICP [8]	451.11	21.24	17.69	0.049701	0.222937	0.184111
Go-ICP [31]	140.47	11.85	2.59	0.00659	0.025665	0.007092
FGR [33]	87.66	9.36	1.99	0.000194	0.013939	0.002839
PointNetLK [25]	227.87	15.09	4.23	0.000487	0.022065	0.005405
DCP [23]	1.31	1.14	0.77	0.000003	0.001786	0.001195
SPA [26]	318.41	17.84	5.43	0.000022	0.004690	0.003261
R-PointHop	0.12	0.34	0.24	0.000000	0.000374	0.000295

unseen objects

TABLE II
REGISTRATION ON UNSEEN CLASSES

Method	MSE (R)	RMSE (R)	MAE (R)	MSE (t)	RMSE (t)	MAE (t)
ICP [8]	467.37	21.62	17.87	0.049722	0.222831	0.186243
Go-ICP [31]	192.25	13.86	2.91	0.000491	0.022154	0.006219
FGR [33]	97.00	9.84	1.44	0.000182	0.013503	0.002231
PointNetLK [25]	306.32	17.50	5.28	0.000784	0.028007	0.007203
DCP [23]	9.92	3.15	2.01	0.000025	0.005039	0.003703
SPA [26]	354.57	18.83	6.97	0.000026	0.005120	0.004211
R-PointHop	0.12	0.34	0.25	0.000000	0.000387	0.000298

unseen classes

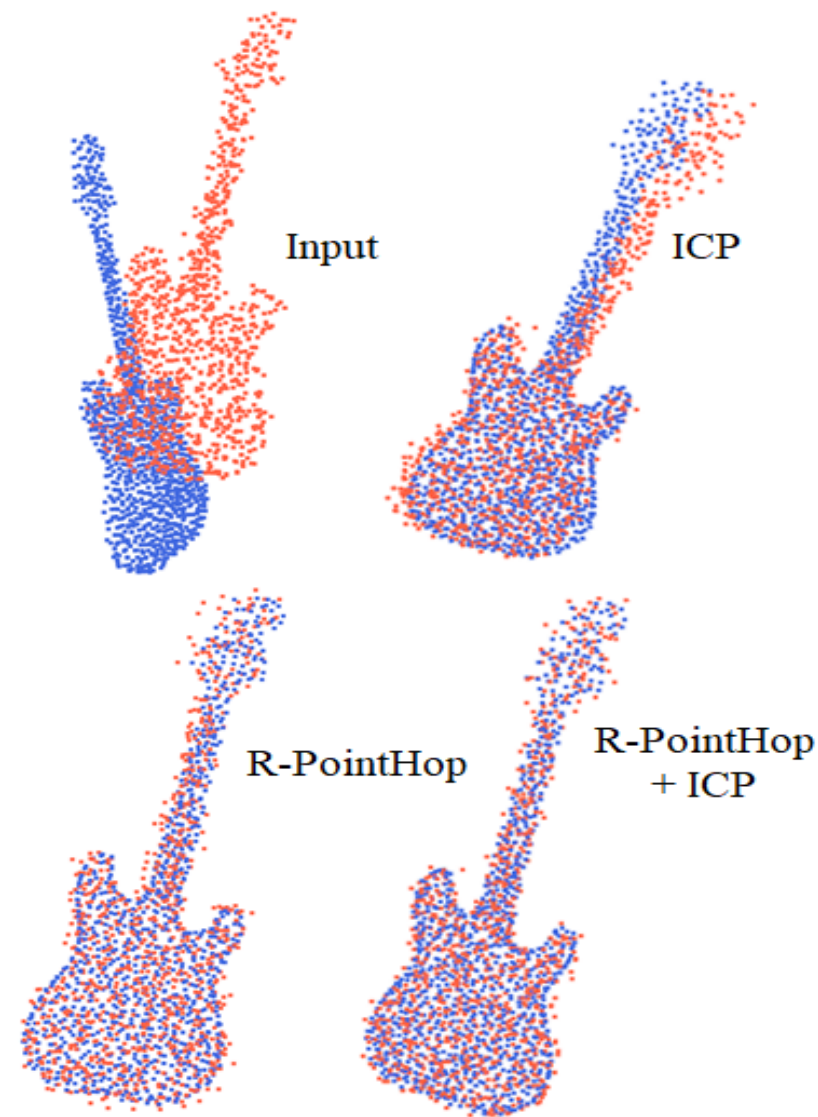
TEST ON NOISY DATA

- Robust to noise
- Refinement using ICP further reduces error

TABLE III
REGISTRATION ON NOISY POINT CLOUDS

Method	MSE (R)	RMSE (R)	MAE (R)	MSE (t)	RMSE (t)	MAE (t)
ICP [8]	558.38	23.63	19.12	0.058166	0.241178	0.206283
Go-ICP [31]	131.18	11.45	2.53	0.000531	0.023051	0.004192
FGR [33]	607.69	24.65	10.05	0.011876	0.108977	0.027393
PointNetLK [25]	256.15	16.00	4.59	0.000465	0.021558	0.005652
DCP [23]	1.17	1.08	0.74	0.000002	0.001500	0.001053
SPA [26]	331.73	18.21	6.28	0.000462	0.021511	0.004100
R-PointHop	7.73	2.78	0.98	0.000001	0.000874	0.003748
R-PointHop + ICP	1.16	1.08	0.21	0.000001	0.000744	0.001002

Add Gaussian noise of zero mean and 0.01 variance to source



TEST ON PARTIAL DATA

- Only a part of the source and target point cloud visible



TABLE IV

REGISTRATION ON PARTIAL POINT CLOUDS (R-POINTHOP* INDICATES CHOOSING CORRESPONDENCES WITHOUT THE RATIO TEST).

Method	Registration errors on unseen objects						Registration errors on unseen classes					
	MSE (R)	RMSE (R)	MAE (R)	MSE (t)	RMSE (t)	MAE (t)	MSE (R)	RMSE (R)	MAE (R)	MSE (t)	RMSE (t)	MAE (t)
ICP [8]	1134.55	33.68	25.05	0.0856	0.2930	0.2500	1217.62	34.89	25.46	0.0860	0.293	0.251
Go-ICP [31]	195.99	13.99	3.17	0.0011	0.0330	0.0120	157.07	12.53	2.94	0.0009	0.031	0.010
FGR [33]	126.29	11.24	2.83	0.0009	0.0300	0.0080	98.64	9.93	1.95	0.0014	0.038	0.007
PointNetLK [25]	280.04	16.74	7.55	0.0020	0.0450	0.0250	526.40	22.94	9.66	0.0037	0.061	0.033
DCP [23]	45.01	6.71	4.45	0.0007	0.0270	0.0200	95.43	9.77	6.95	0.0010	0.034	0.025
PR-Net [38]	10.24	3.12	1.45	0.0003	0.0160	0.0100	15.62	3.95	1.71	0.0003	0.017	0.011
R-PointHop*	3.58	1.89	0.11	0.0002	0.0150	0.0008	3.75	1.94	0.12	0.0002	0.0151	0.0008
R-PointHop	2.75	1.66	0.09	0.0002	0.0149	0.0008	2.53	1.59	0.08	0.0002	0.0148	0.0008

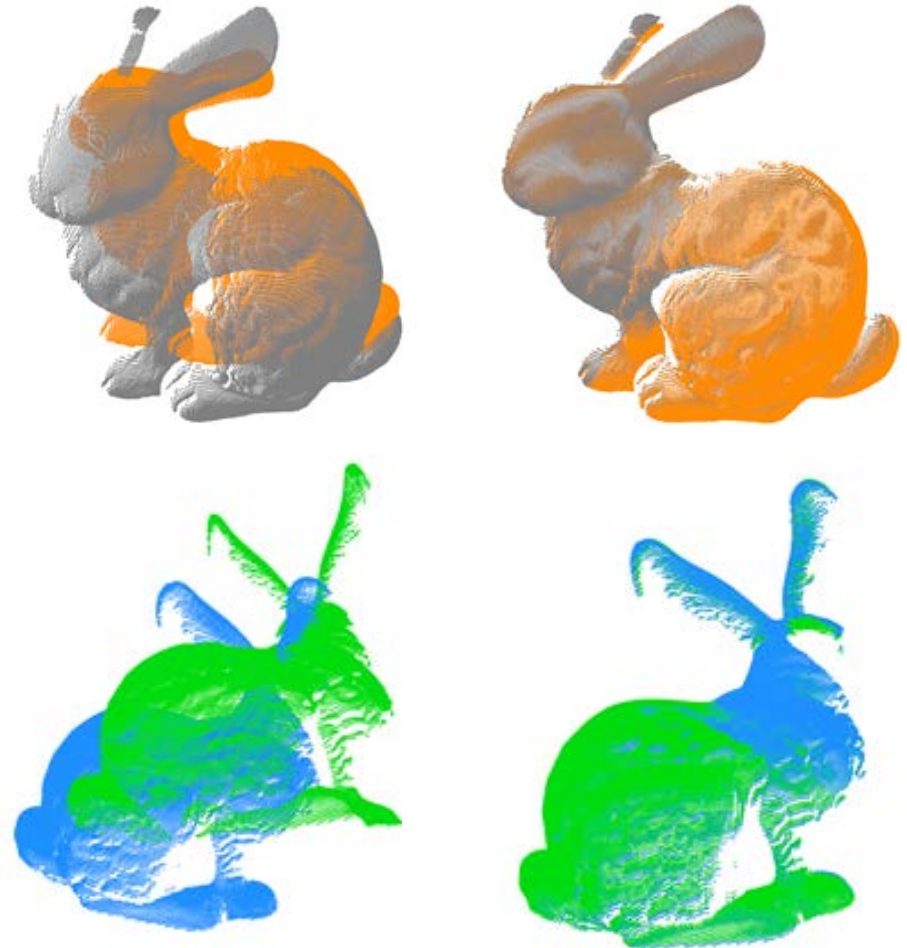
TEST ON REAL WORLD DATA

STANFORD BUNNY

- Real scans of bunny – 10 scans, ~50k points in each point cloud
- The model trained on ModelNet40 is reused
- Surprisingly, DCP does not do so well for this dataset

TABLE V
REGISTRATION ON THE STANFORD BUNNY DATASET

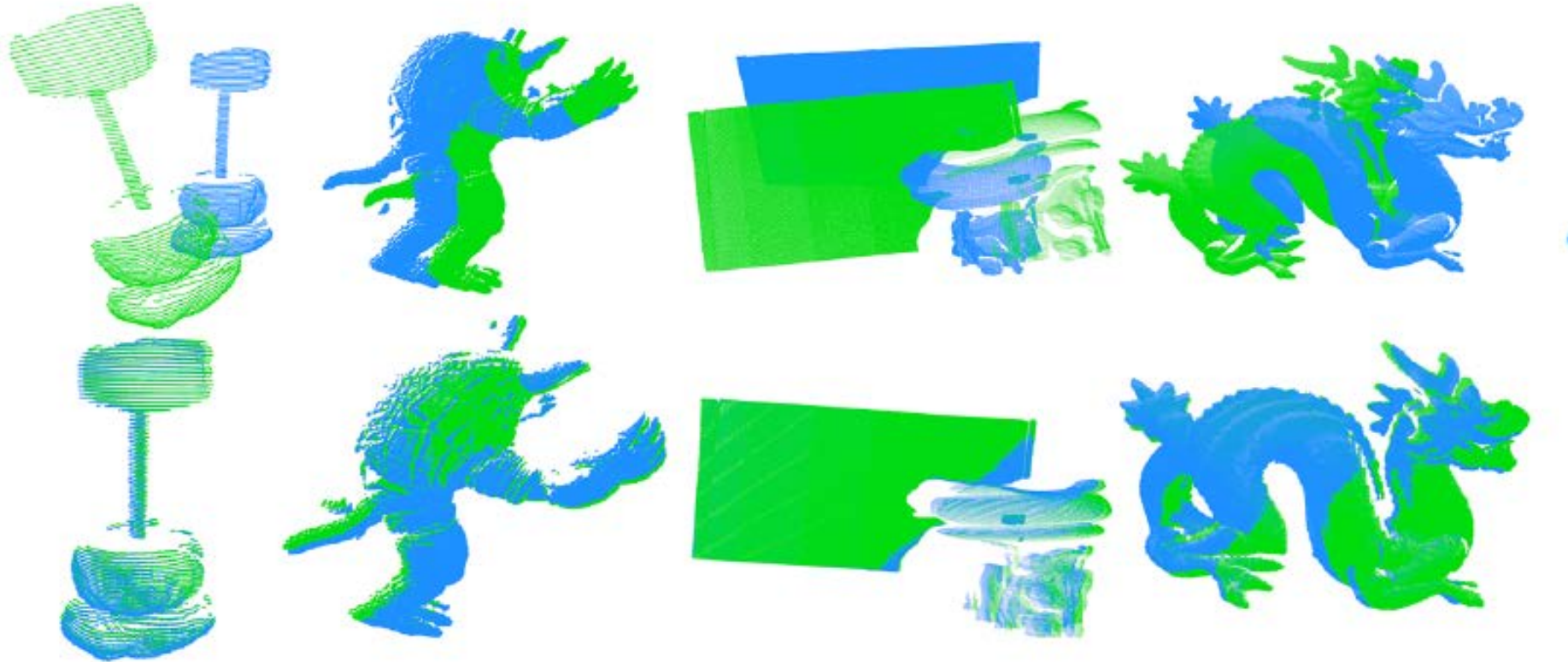
Method	MSE (R)	RMSE (R)	MAE (R)	MSE (t)	RMSE (t)	MAE (t)
ICP [8]	177.35	13.32	10.72	0.0024	0.0492	0.0242
Go-ICP [31]	166.85	12.92	4.52	0.0018	0.0429	0.0282
FGR [33]	3.98	1.99	1.49	0.0397	0.1993	0.1658
DCP [23]	41.45	6.44	4.78	0.0016	0.0406	0.0374
R-PointHop	2.21	1.49	1.09	0.0013	0.0361	0.0269



TEST ON REAL WORLD DATA

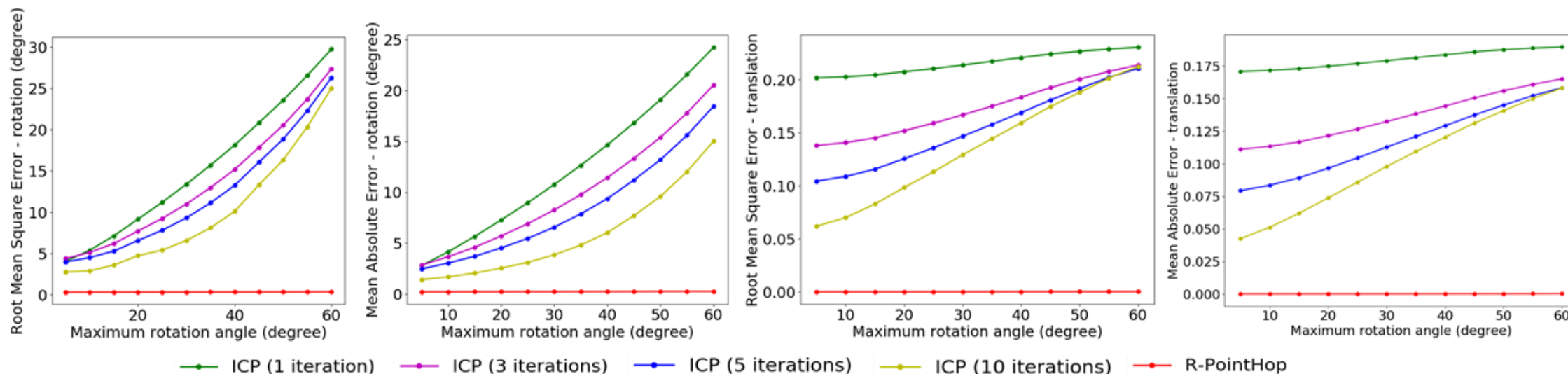
STANFORD 3D SCANNING REPOSITORY

- Registration on some more scans – drill, armadillo, Buddha, dragon



LOCAL VS GLOBAL REGISTRATION

- ICP and its variants work well only in presence of a good initial alignment (local in nature)
- Rotation invariant features make R-PointHop useful for global registration
- Can be used as an initialization for ICP



TOWARDS GREEN LEARNING

- Deep learning has reformulated registration as a **supervised learning problem** – large number of pairs of point clouds used along with ground truth transformations
- Large model size, longer training times, expensive GPUs – **large carbon footprint**
- Classical model-free methods are favorable in this respect, but poor performance

Method	Training time	Model size	Resources
PointNetLK	40 minutes / epoch (200 epochs)	630kB	1 GPU
Deep Closest Point (DCP)	27 hours	21MB	8 GPUs
R-PointHop	40 minutes	200kB	CPU + multithreading

Green learning – low cost and high performance!

Conclusion

Similarities of GL and DL

	GL	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

Differences between GL and DL

	GL	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