Computational Visual Pathways for Multi-Task Learning and Simulation

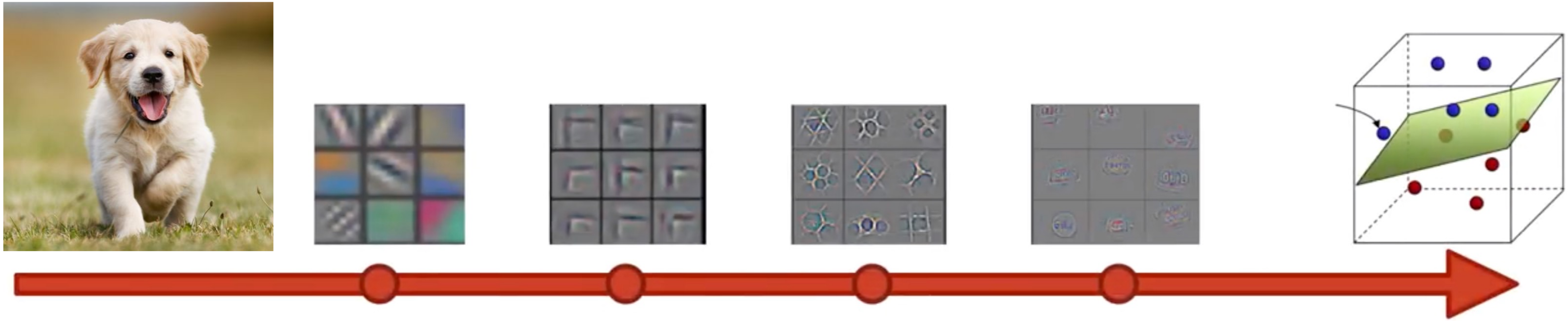
Rogerio Schmidt Feris

Principal Scientist and Manager

MIT-IBM Watson AI Lab

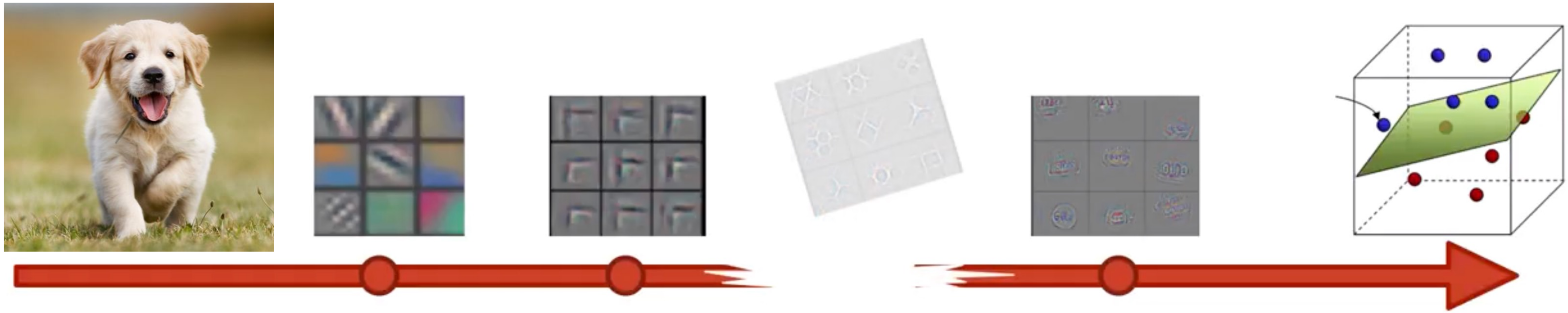
Feed-Forward Convolutional Neural Networks

- Single path, where the exact same set of features are extracted for all inputs



Adapted from Veit et al

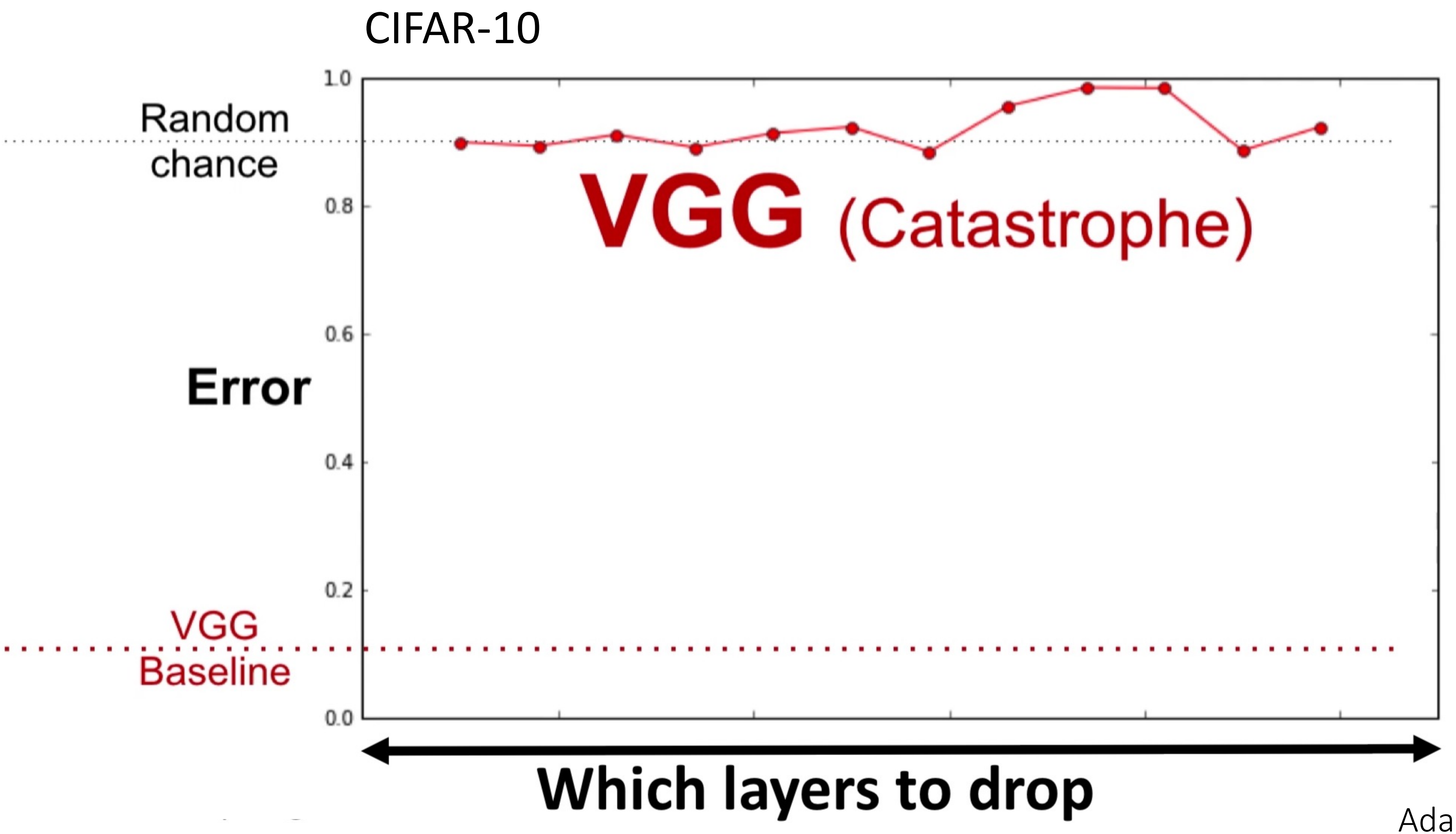
Feed-Forward Convolutional Neural Networks



What happens when we drop a layer at test time?

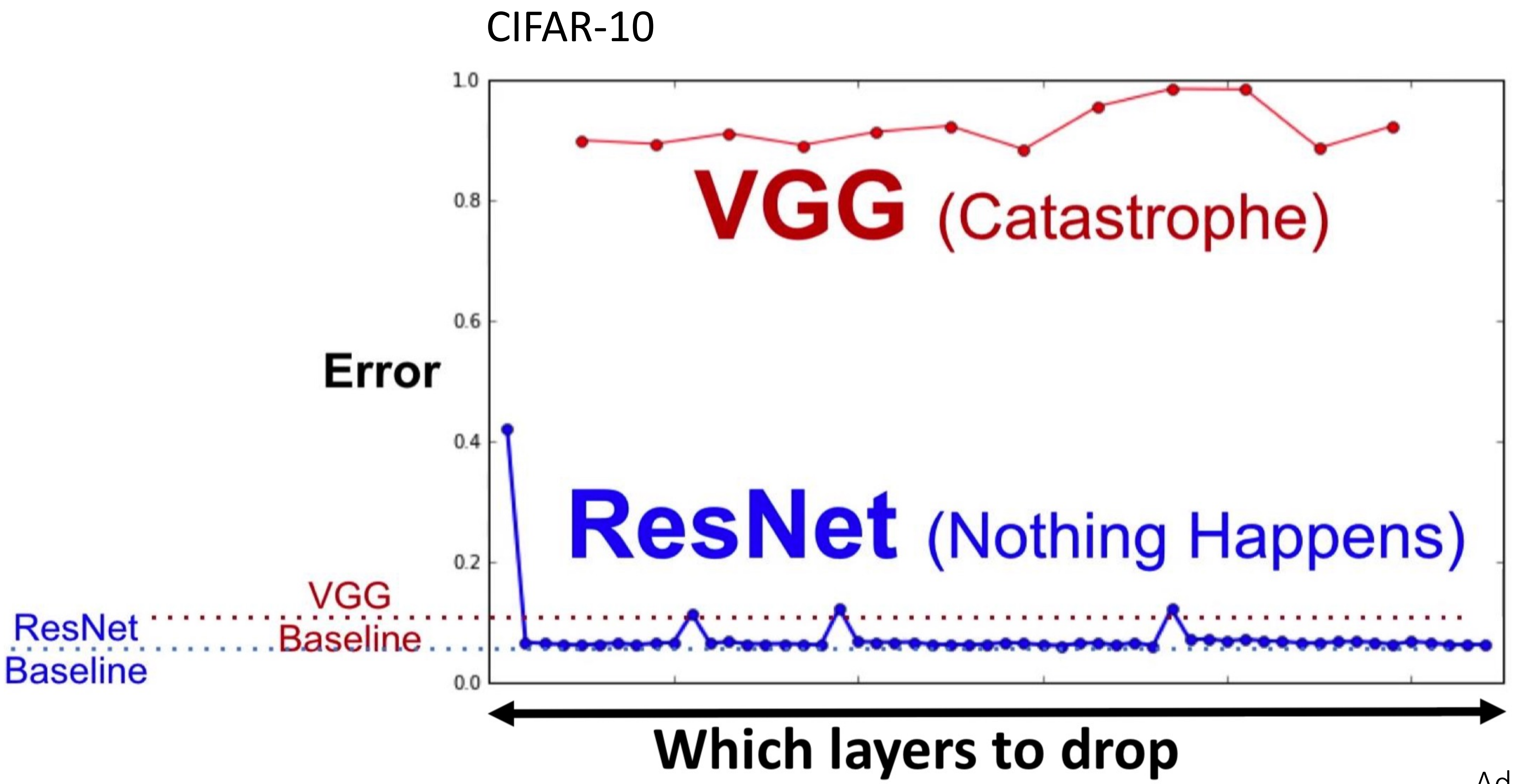
Adapted from Veit et al

What happens when we drop a layer at test time?



Adapted from Veit et al

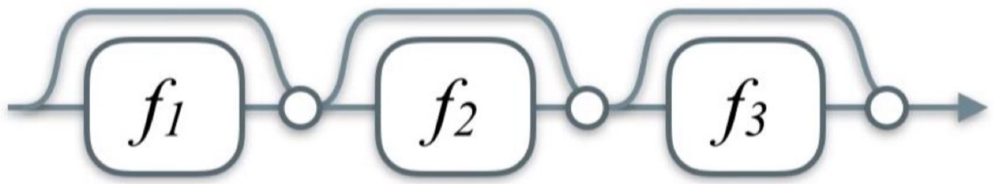
What happens if we delete a layer at test time?



Adapted from Veit et al

Why does this happen?

VGG

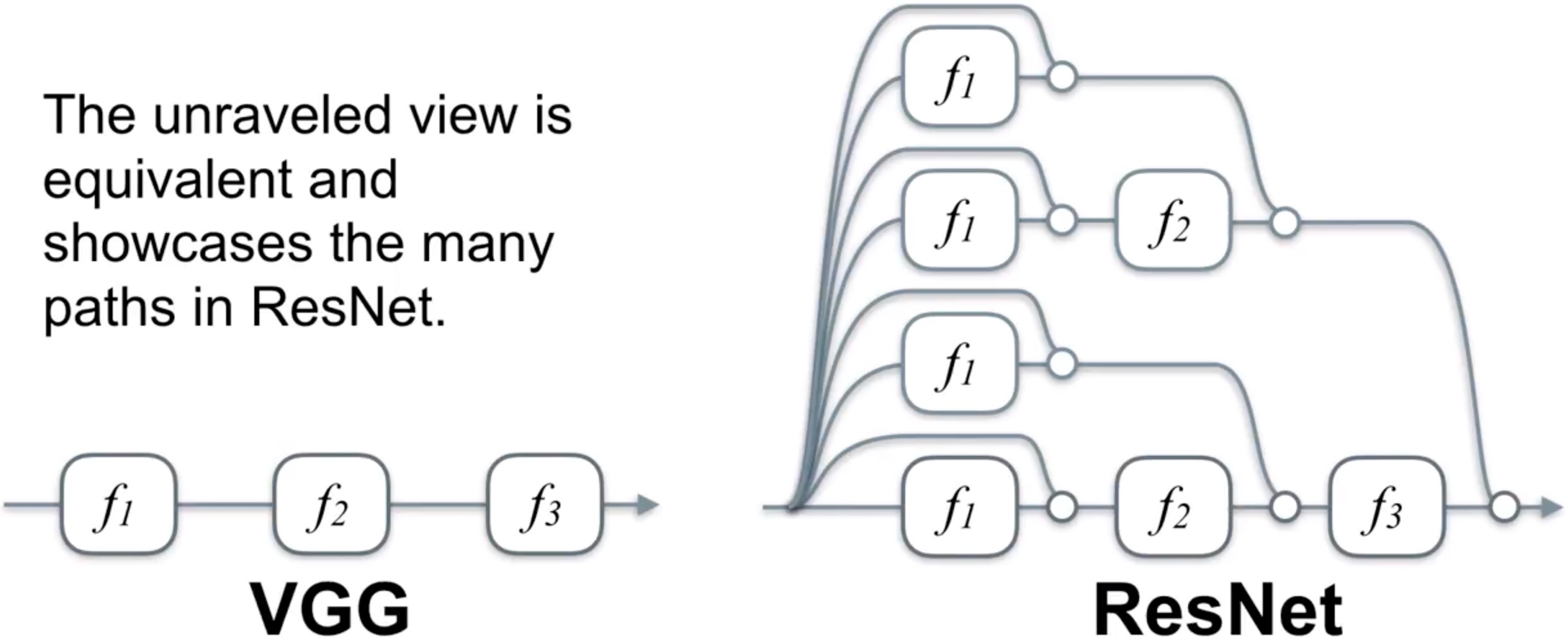


ResNet

Adapted from Veit et al

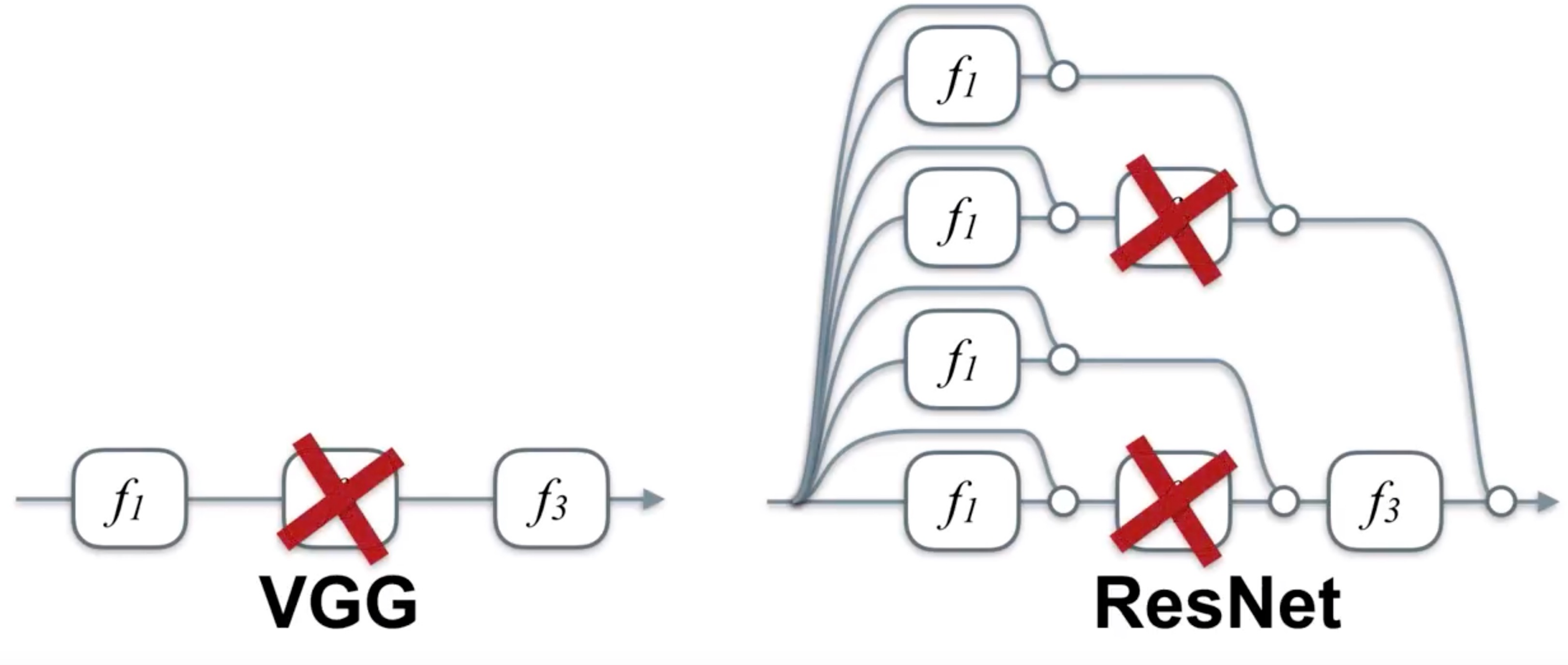
Why does this happen?

The unraveled view is equivalent and showcases the many paths in ResNet.



Adapted from Veit et al

Deletion of a Layer

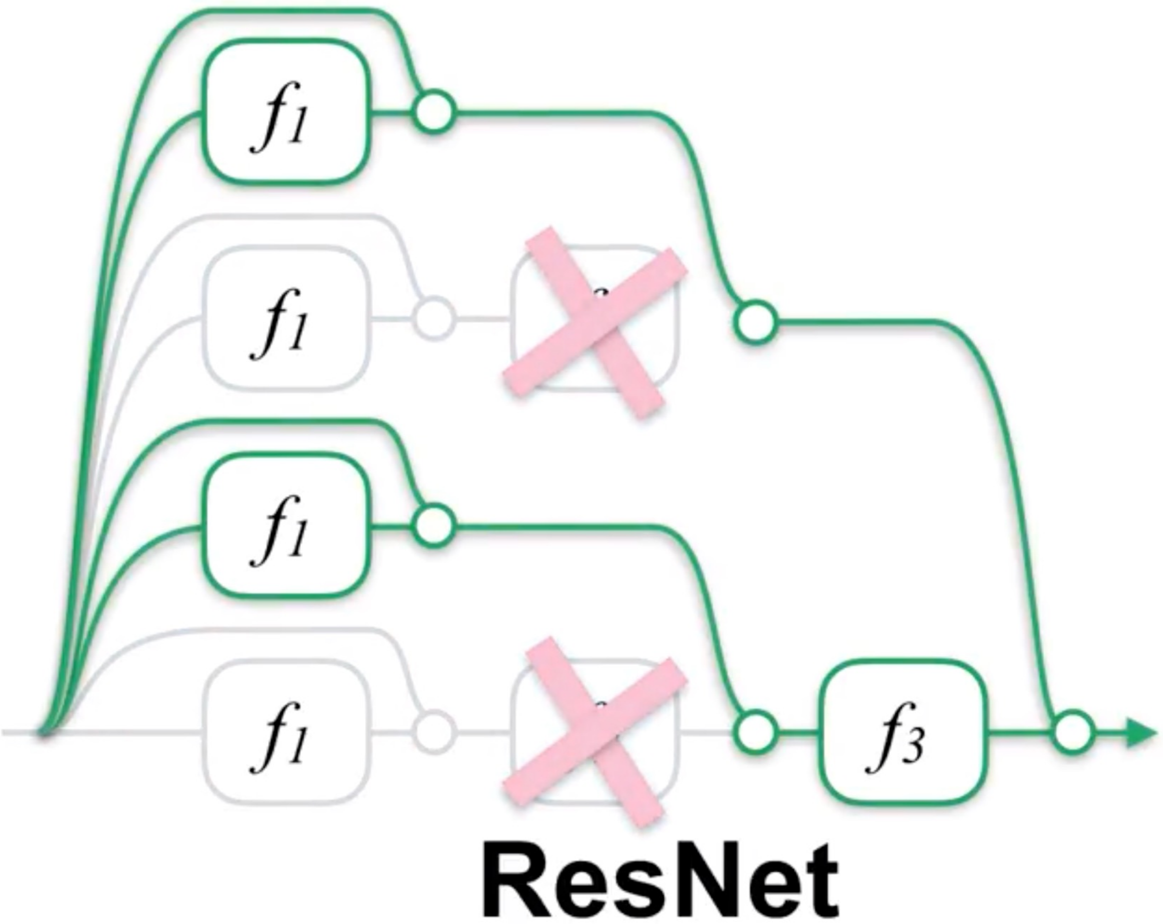


Adapted from Veit et al

Deletion of a Layer

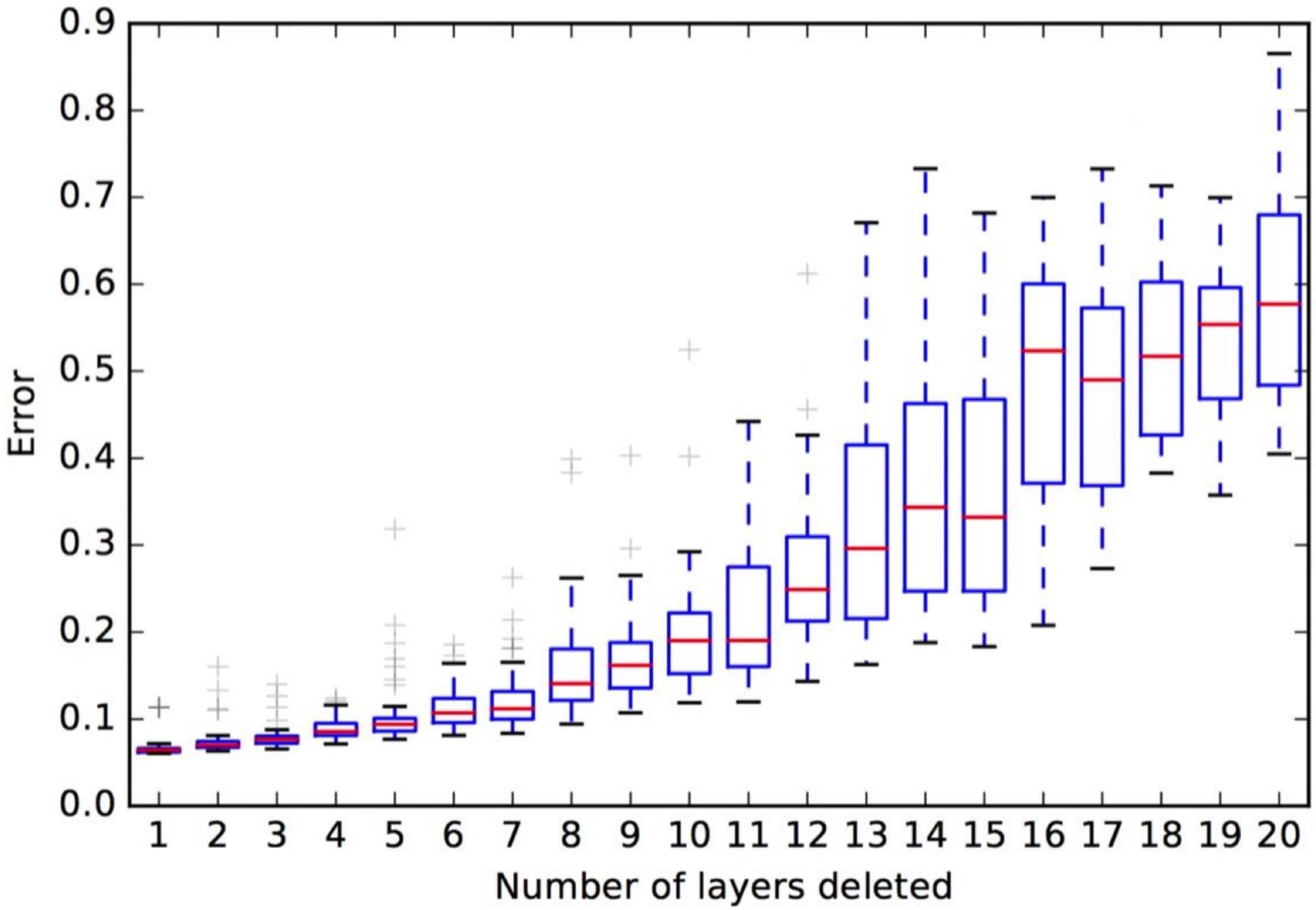
Only half of the paths are affected

All paths are affected



Adapted from Veit et al

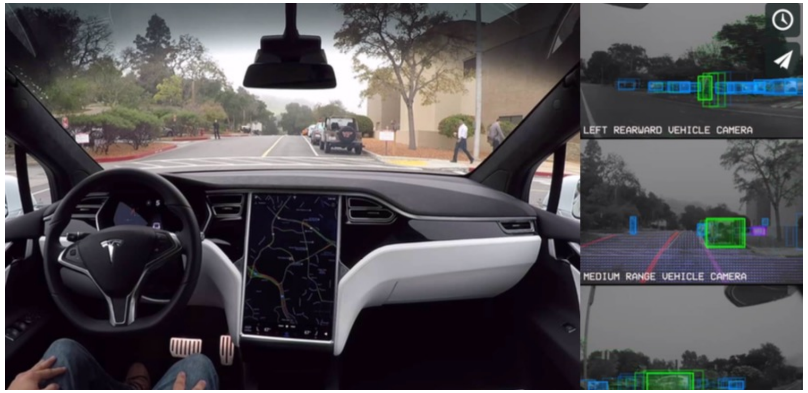
Performance varies smoothly when deleting <u>several</u> layers



Adapted from Veit et al

Can we delete a sequence of layers without performance drop?

Important for applications where fast inference is essential



Can we delete a sequence of layers without performance drop?

In the experiment of [Veit et al, 2016]:

- Layers were dropped randomly
- Same layers were dropped for all images

BlockDrop: Dynamic Inference Paths in Residual Networks

CVPR 2018


Zuxuan Wu


Tushar Nagarajan


Abhishek Kumar

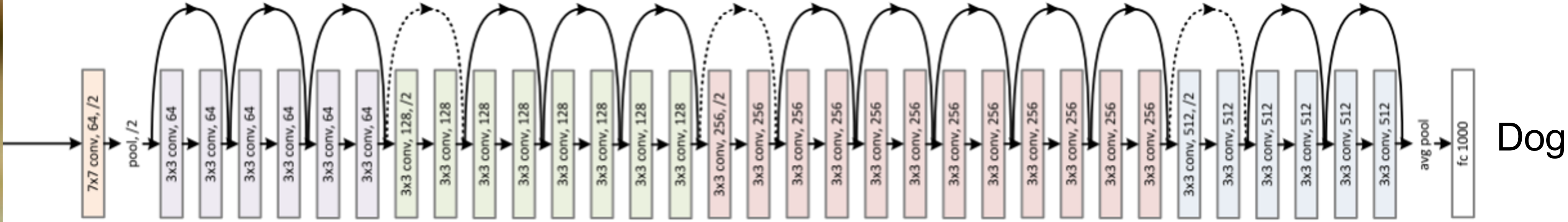

Steve Rennie


Larry Davis


Kristen Grauman


Rogerio Feris

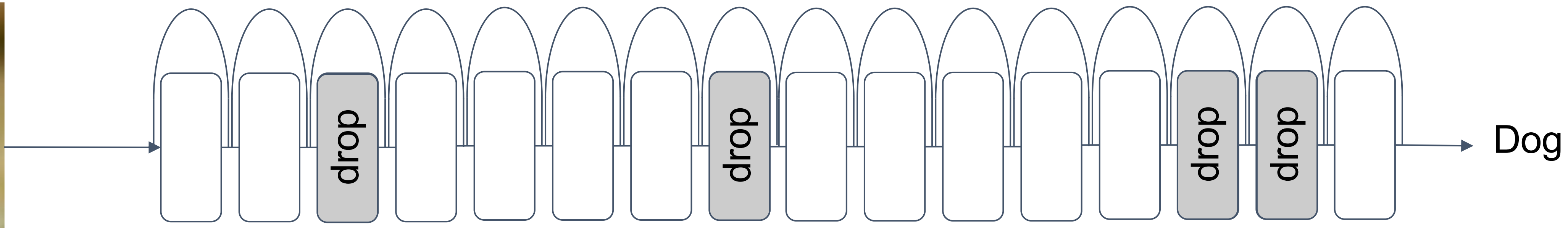
BlockDrop: Dynamic Inference Paths in Residual Networks



Do we really need to run 100+ layers / residual blocks of a neural network (which is expensive) if we have an “easy” input image?

[Wu & Nagarajan et al, CVPR 2018]

BlockDrop: Dynamic Inference Paths in Residual Networks



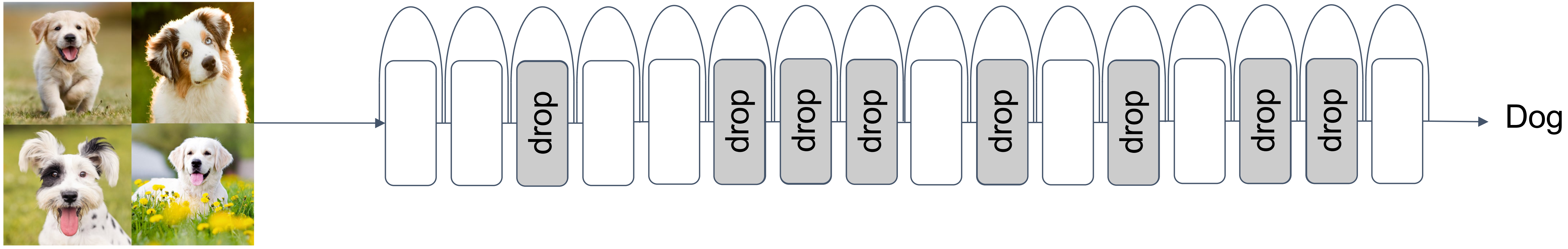
“Dropping some blocks during testing doesn’t hurt performance much”

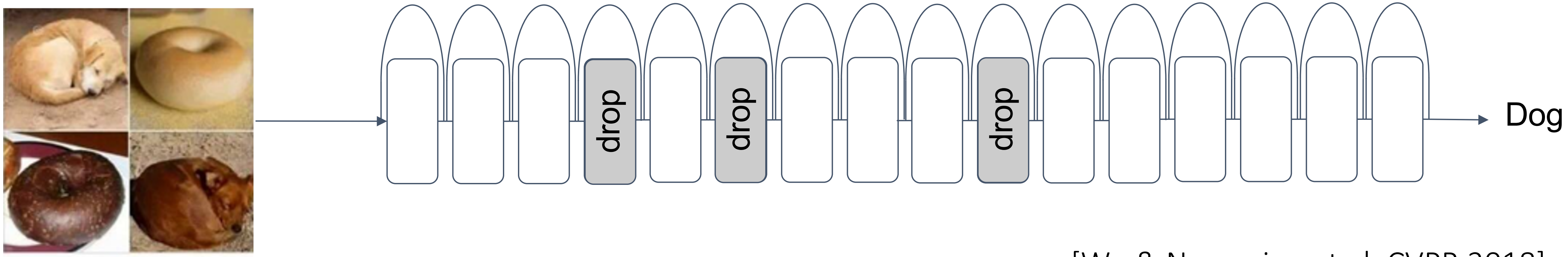
(Veit et al., NIPS 16)

[Wu & Nagarajan et al, CVPR 2018]

BlockDrop: Dynamic Inference Paths in Residual Networks

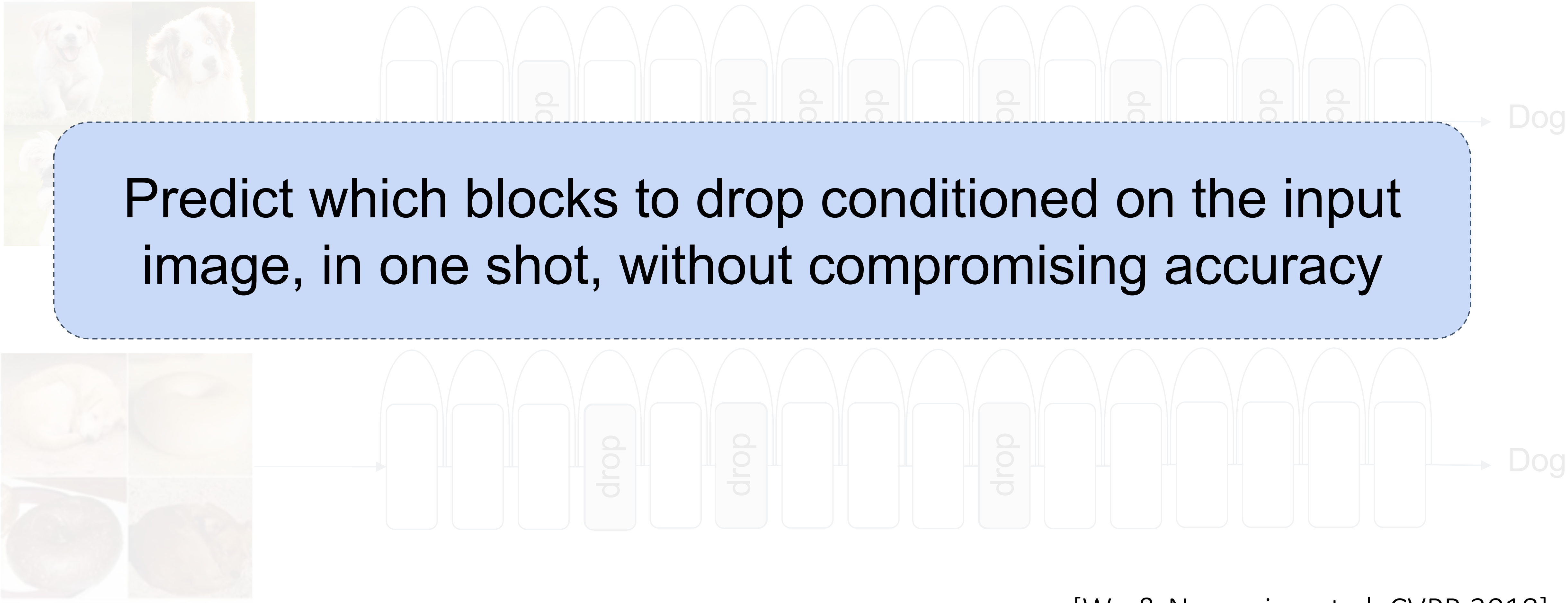
How to determine which blocks to drop depending on the input image?





[Wu & Nagarajan et al, CVPR 2018]

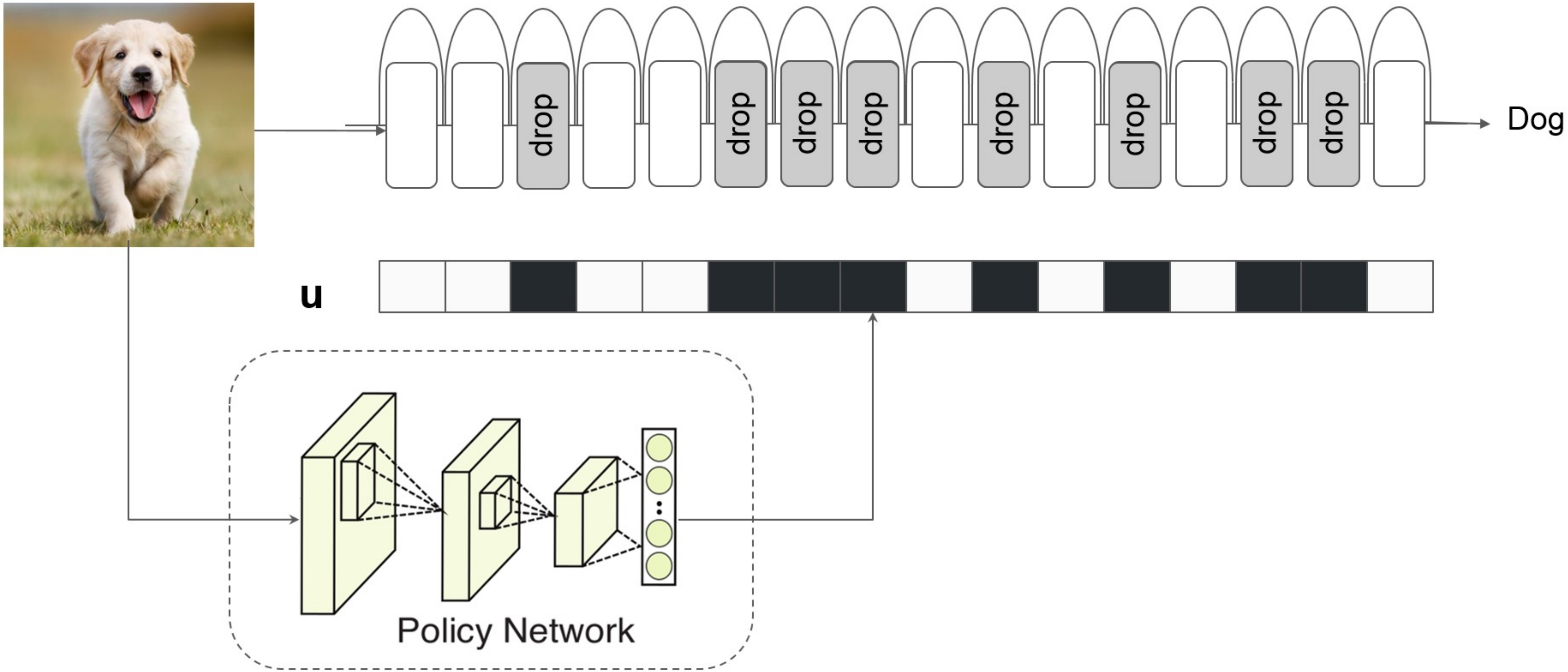
Our Idea: BlockDrop



Predict which blocks to drop conditioned on the input image, in one shot, without compromising accuracy

[Wu & Nagarajan et al, CVPR 2018]

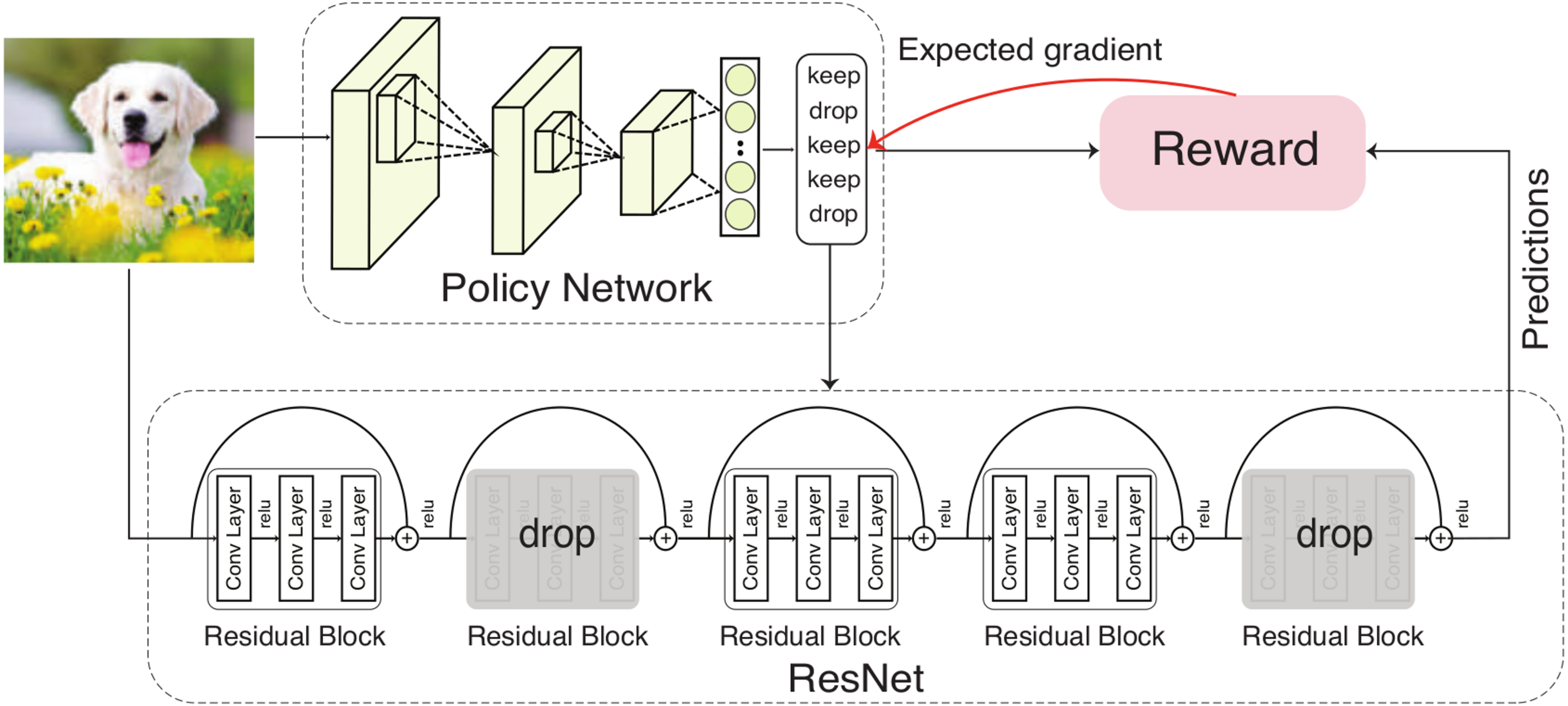
BlockDrop: Dynamic Inference Paths in Residual Networks



[Wu & Nagarajan et al, CVPR 2018]

BlockDrop: Dynamic Inference Paths in Residual Networks

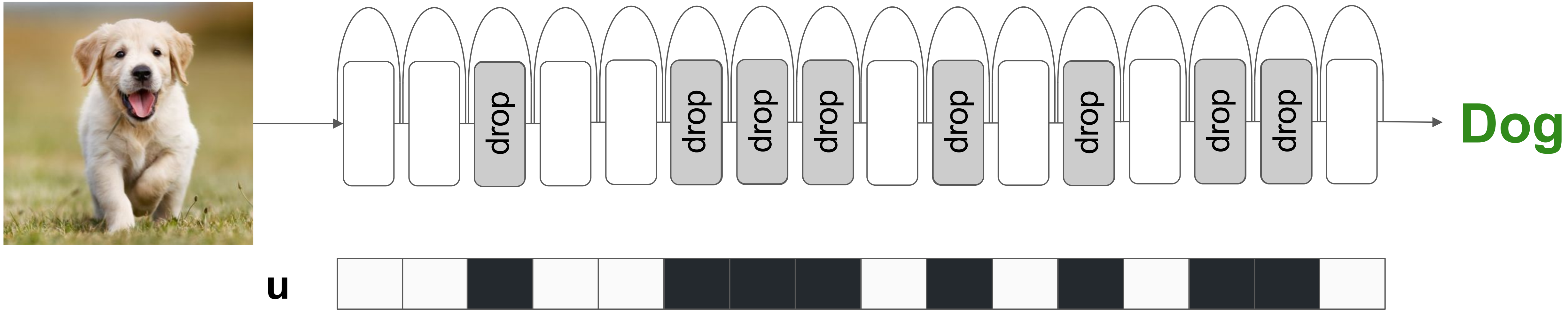
Policy Network Training using Policy Gradients



[Wu & Nagarajan et al, CVPR 2018]

BlockDrop: Dynamic Inference Paths in Residual Networks

- Reward function takes into account both accuracy and block usage

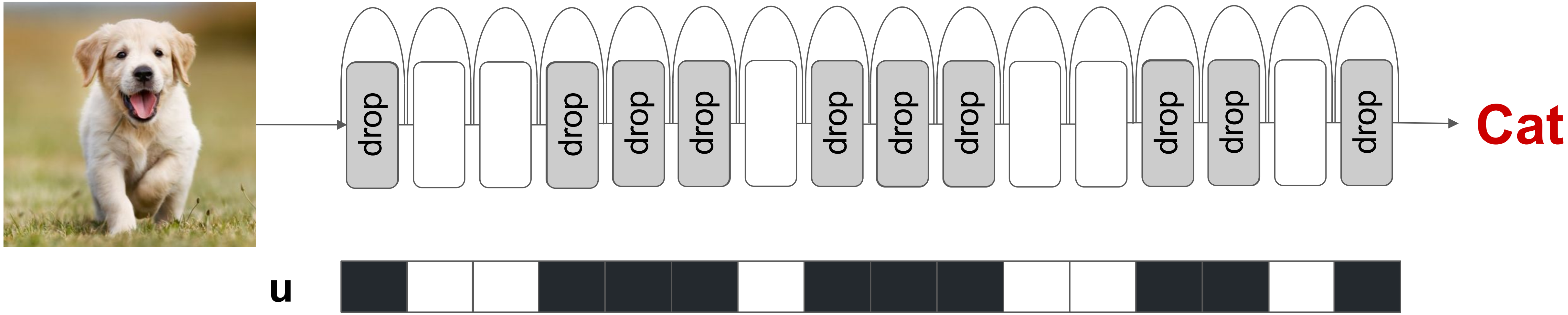


$$R(\mathbf{u}) = \begin{cases} 1 - (\frac{|\mathbf{u}|_0}{K})^2 & \text{if correct} \\ -\gamma & \text{otherwise.} \end{cases}$$

$$R(\mathbf{u}) = \mathbf{1} - \left(\frac{\mathbf{8}}{\mathbf{16}}\right)^{\mathbf{2}} = \mathbf{0.75}$$

[Wu & Nagarajan et al, CVPR 2018]

BlockDrop: Dynamic Inference Paths in Residual Networks



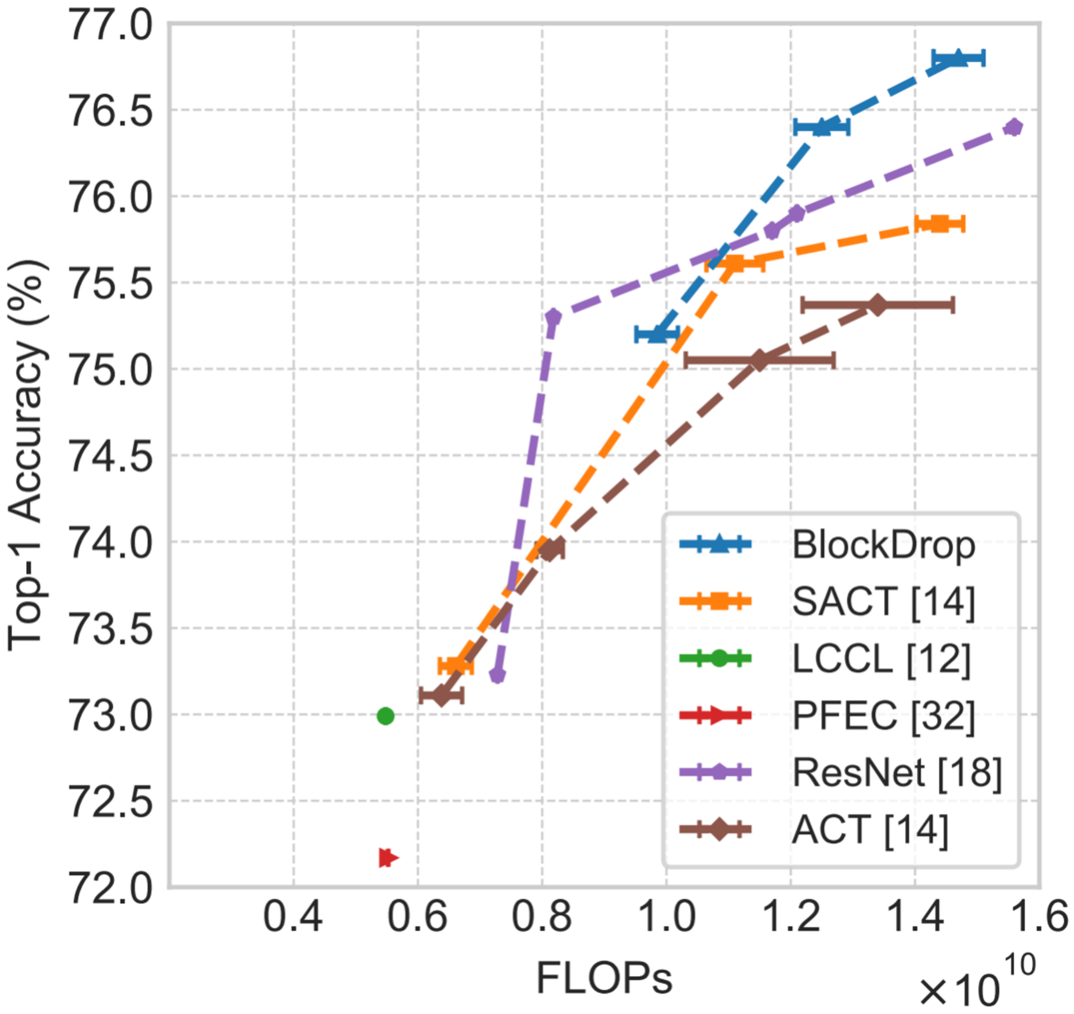
$$R(\mathbf{u}) = \begin{cases} 1 - \left(\frac{|\mathbf{u}|_0}{K}\right)^2 & \text{if correct} \\ -\gamma & \text{otherwise.} \end{cases}$$

$$R(\mathbf{u}) = 1 - \left(\frac{8}{16}\right)^2 = 0.75$$

$$R(\mathbf{u}) = -\mathbf{10}$$

[Wu & Nagarajan et al, CVPR 2018]

BlockDrop: Dynamic Inference Paths in Residual Networks



Results on ImageNet:

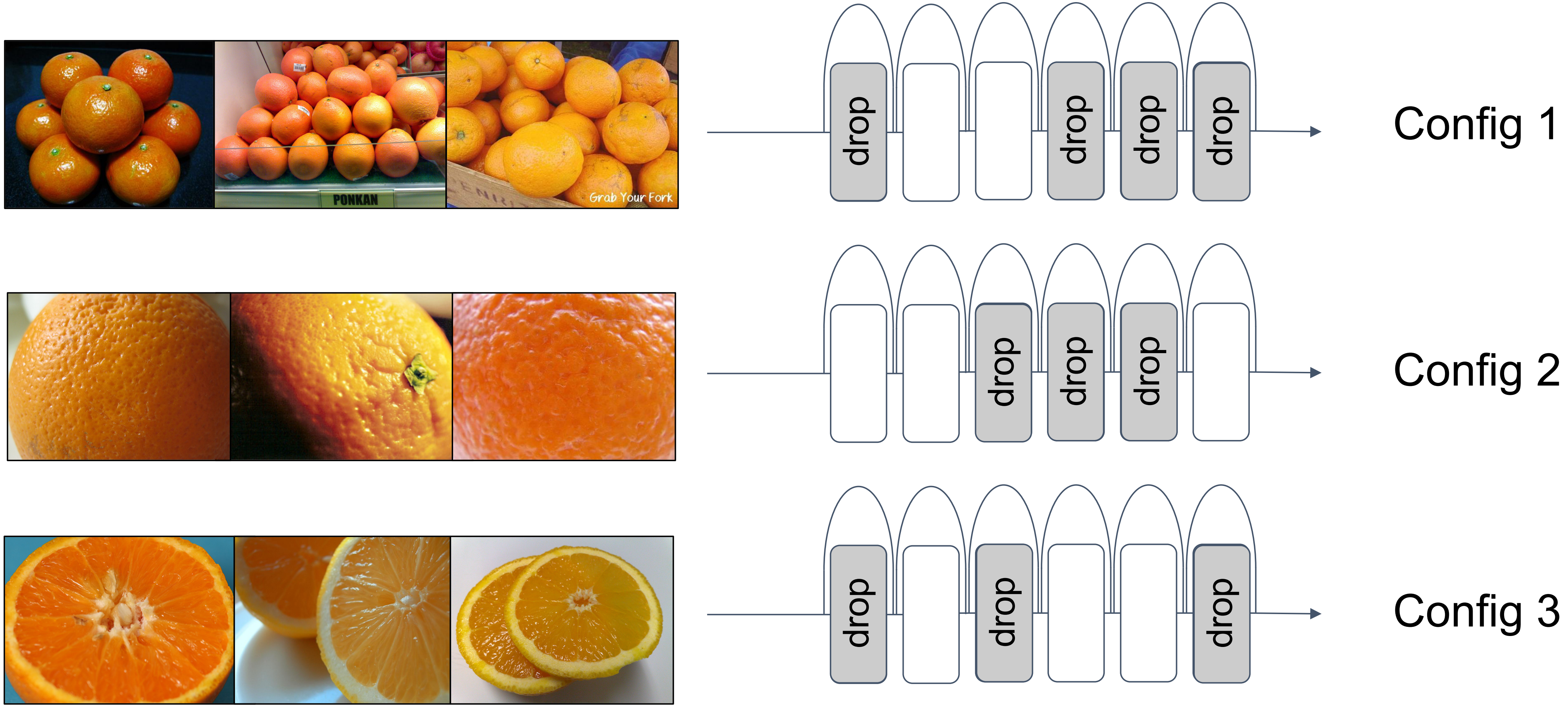
20% - 36% computational savings (FLOPs)

Complementary to other model compression techniques

[Wu & Nagarajan et al, CVPR 2018]

BlockDrop: Dynamic Inference Paths in Residual Networks

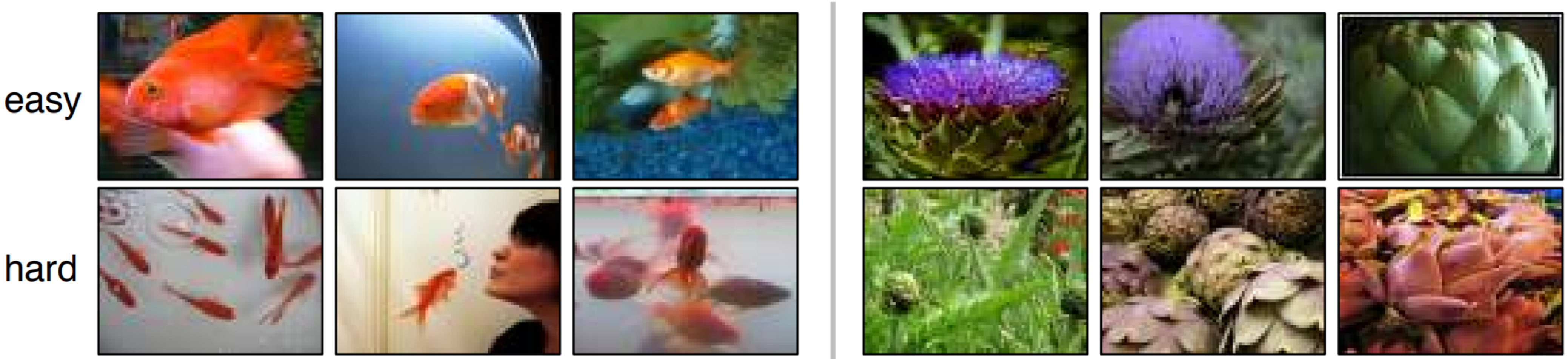
- Different policies capture different visual patterns



orange

[Wu & Nagarajan et al, CVPR 2018]

BlockDrop: Dynamic Inference Paths in Residual Networks



Goldfish - easy (23 blocks) vs. hard (29 blocks)

Artichoke - easy (18 blocks) vs. hard (28 blocks)

Block usage in neural networks agrees with our perception of *difficulty*

[Wu & Nagarajan et al, CVPR 2018]

Adashare: Learning What To Share For Efficient Deep Multi-Task Learning

NeurIPS 2020

Ximeng Sun

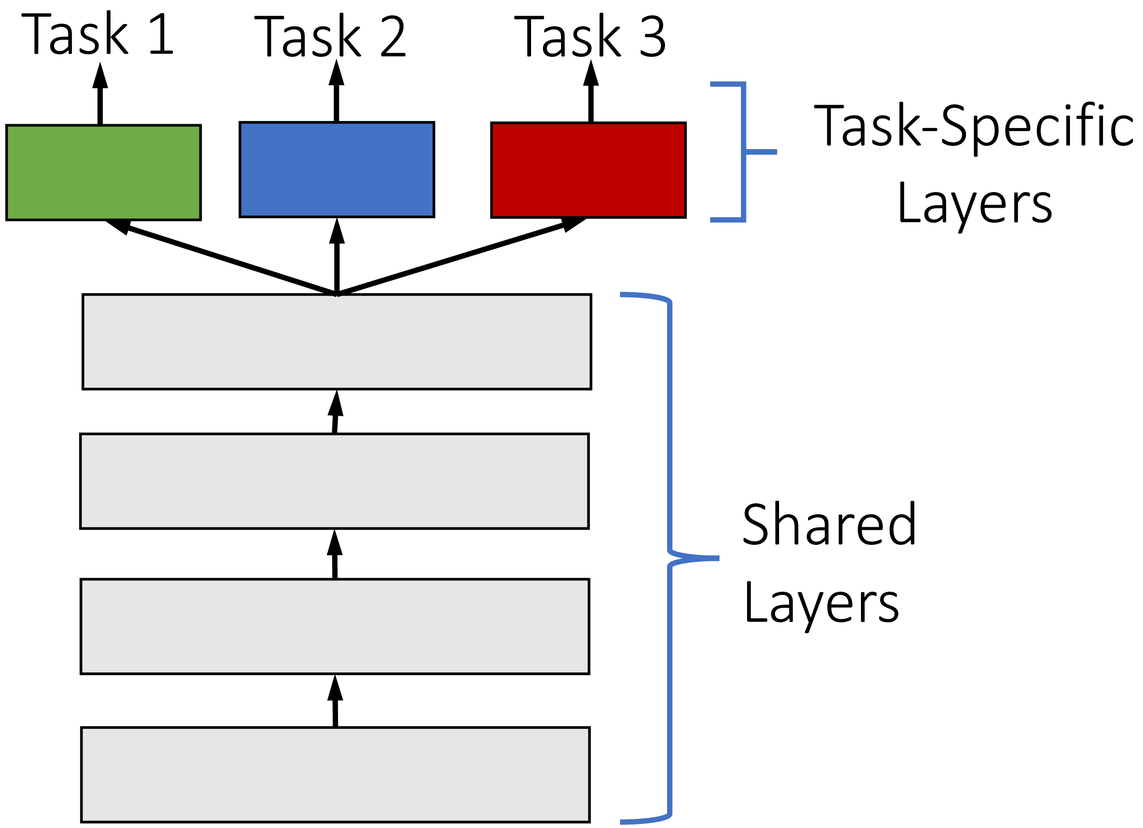
Rameswar Panda

Kate Saenko

Rogerio Feris

Hard Parameter Sharing

- Hand-designed architectures composed of base layers that are shared across tasks and specialized branches that learn task-specific features.

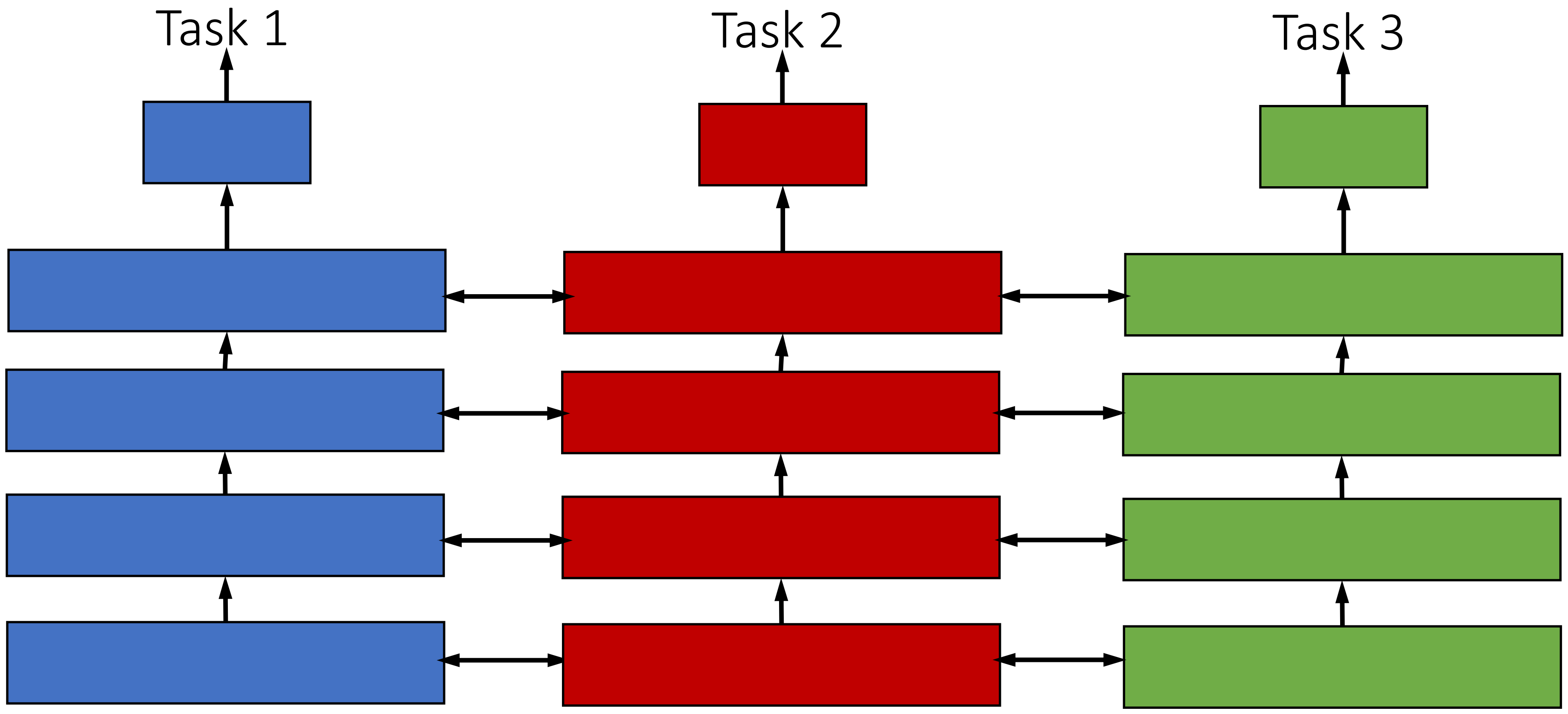


- Performance depends on “where to branch” in the network [Misra et al, 2016]
- The space of possible branching architectures is combinatorially large !

Soft Parameter Sharing

- Network column for each task and a mechanism for feature sharing between columns.

Number of parameters grow linearly with the number of tasks !

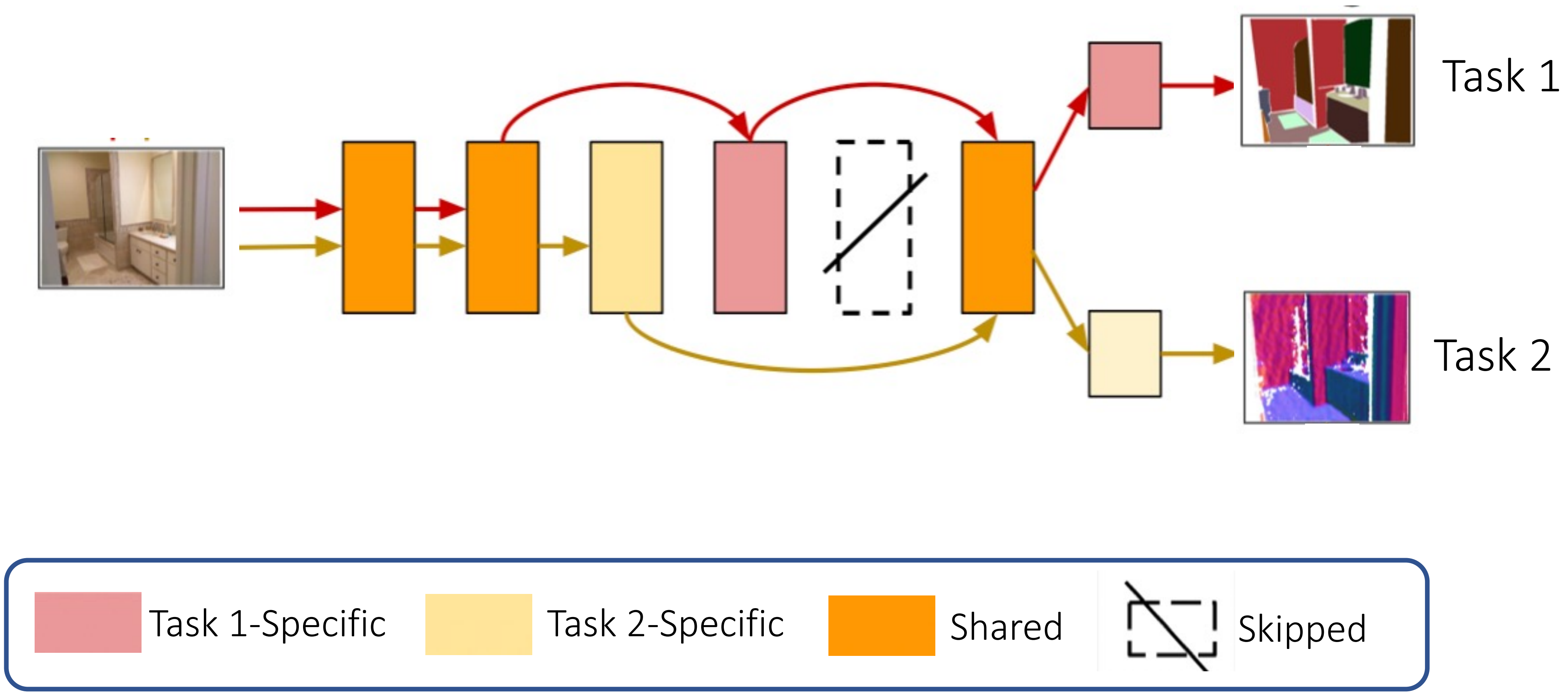


Problem

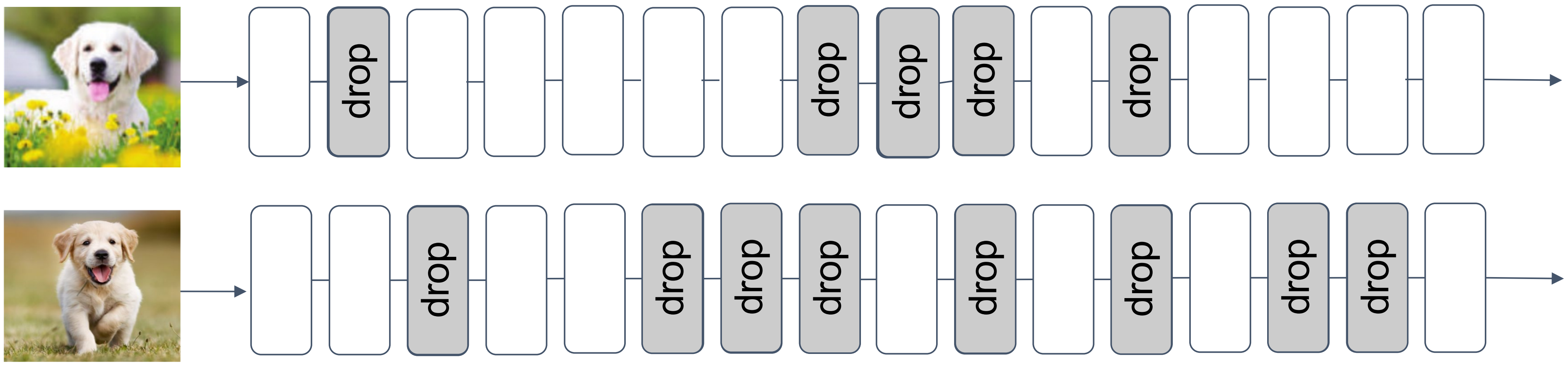
Can we determine which layers in the network should be shared across which tasks and which layers should be task-specific to achieve the best accuracy/memory footprint trade-off for scalable and efficient multi-task learning?

Proposed Approach: AdaShare

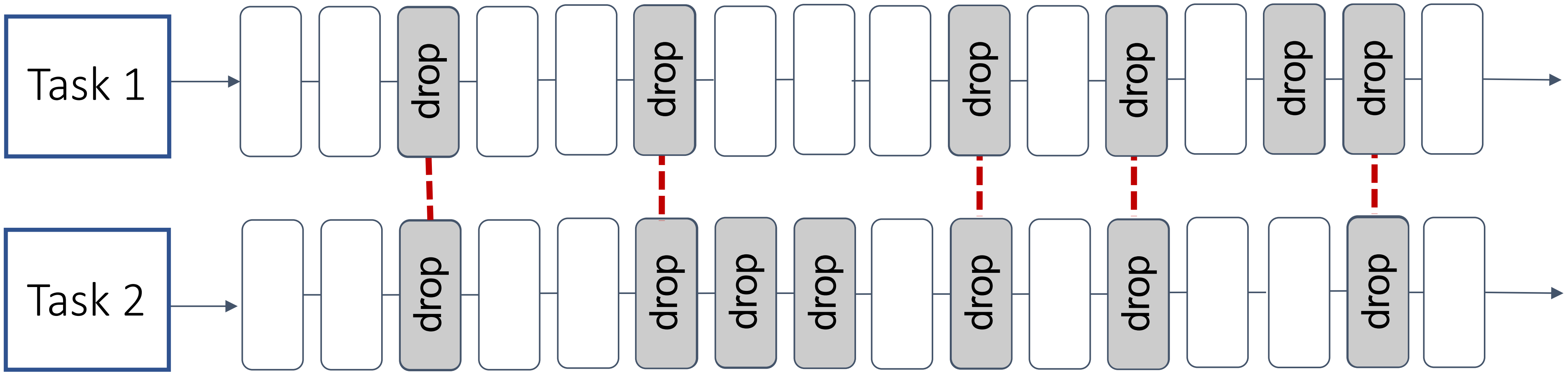
- Single network that supports separate execution paths for different tasks



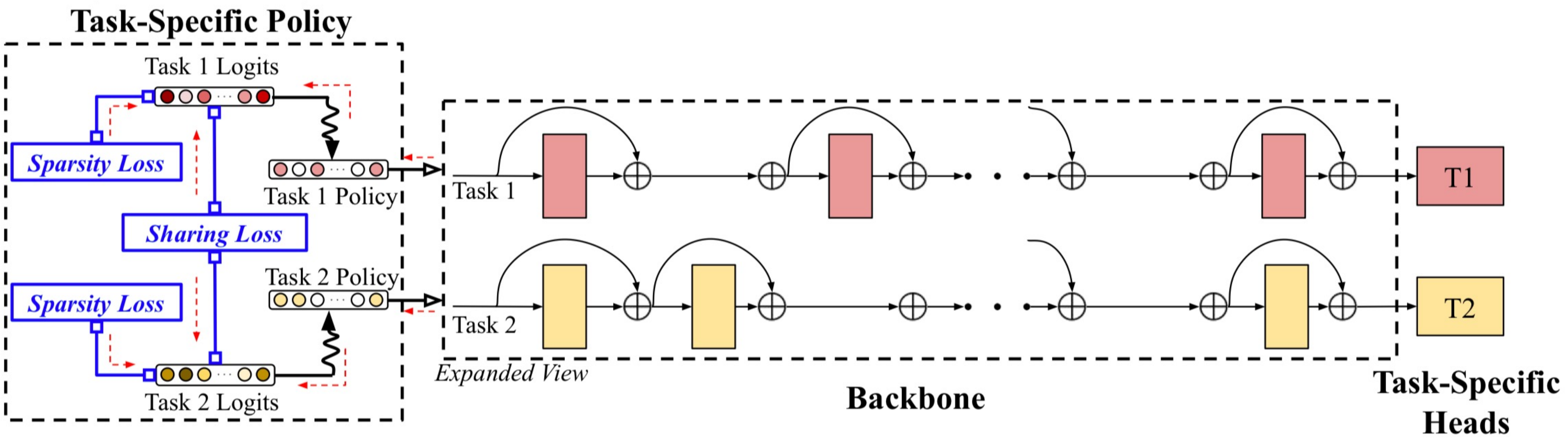
BlockDrop: Per-instance routing; Accuracy + Sparsity reward



AdaShare: Per-task routing; Accuracy + Sparsity + Sharing reward

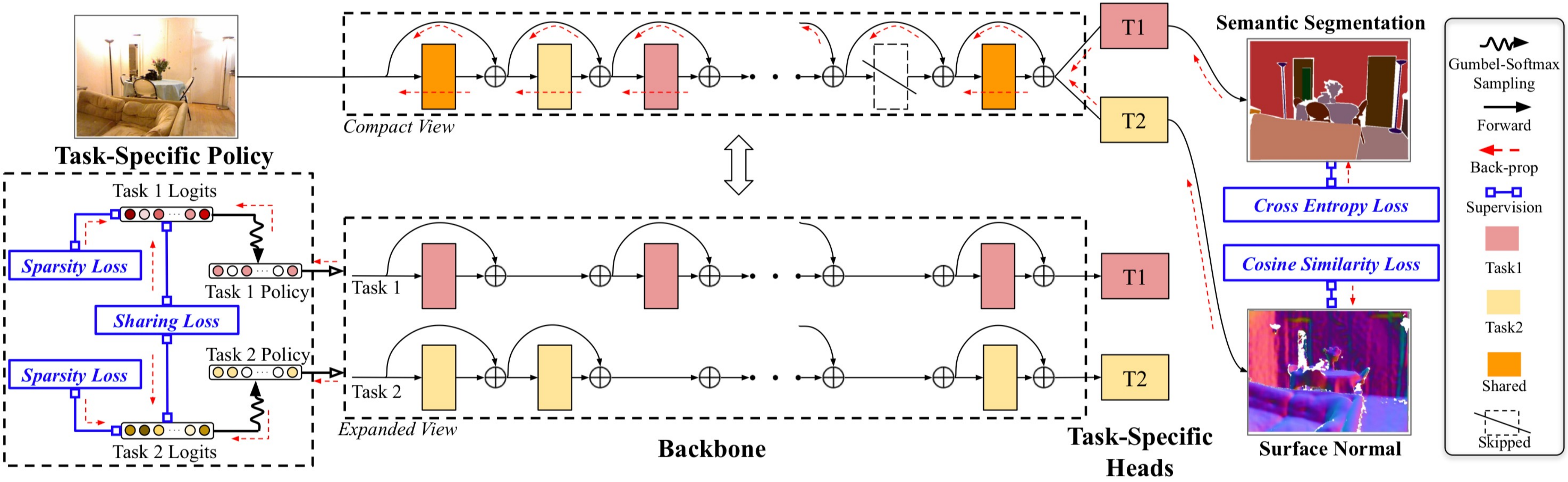


AdaShare: Learning what to Share in Multi-Task Learning




Gumbel-Softmax Sampling

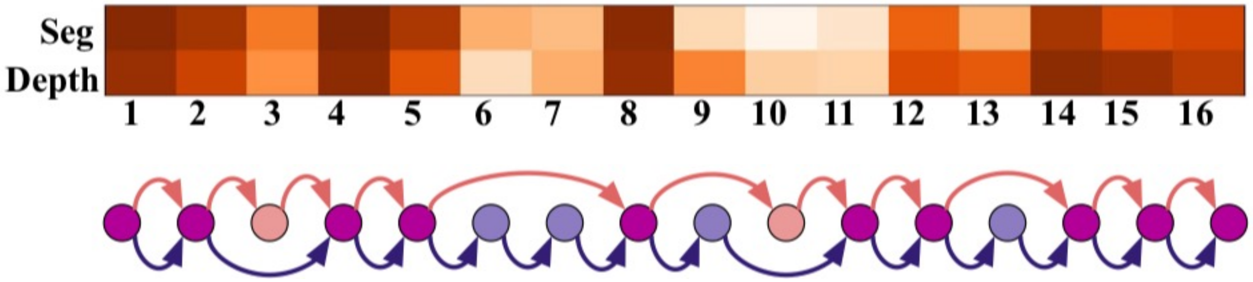
AdaShare: Learning what to Share in Multi-Task Learning



AdaShare: Experimental Results

- **CityScapes [2 tasks].** *AdaShare* achieves the best performance on 5 out of 7 metrics using less than 1/2 parameters of most baselines.

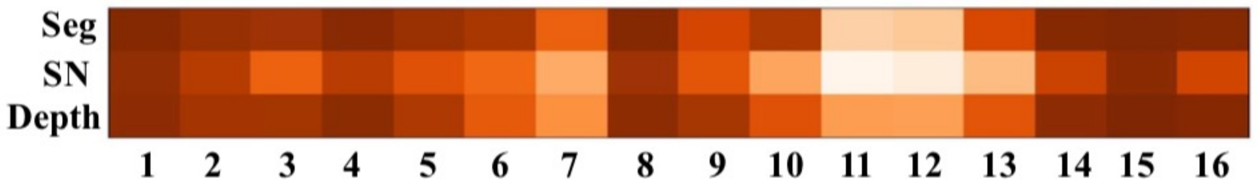
Model	# Params ↓	Semantic Seg.		Depth Prediction				
		mIoU ↑	Pixel Acc ↑	Error↓		δ, within ↑		
				Abs	Rel	1.25	1.25^2	1.25^3
Single-Task	2	40.2	74.7	0.017	0.33	70.3	86.3	93.3
Multi-Task	**1**	37.7	73.8	0.018	0.34	72.4	88.3	94.2
Cross-Stitch	2	40.3	74.3	**0.015**	**0.30**	74.2	89.3	**94.9**
Sluice	2	39.8	74.2	0.016	0.31	73.0	88.8	94.6
NDDR-CNN	2.07	**41.5**	74.2	0.017	0.31	74.0	89.3	94.8
MTAN	2.41	40.8	74.3	**0.015**	0.32	75.1	89.3	94.6
AdaShare	**1**	**41.5**	**74.9**	0.016	0.33	**75.5**	**89.8**	**94.9**

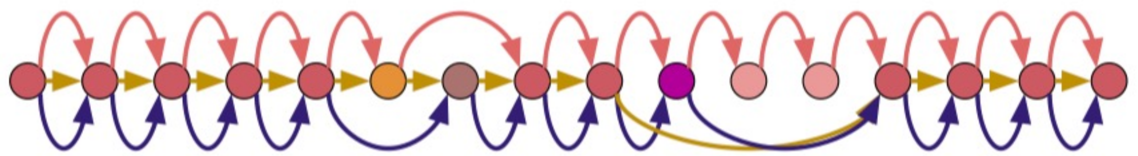


AdaShare: Experimental Results

- **NYU v2 [3 tasks].** AdaShare achieves the best performance on 10 out of 12 metrics using less than 1/3 parameters of most baselines.

Model	# Params ↓	Semantic Seg.		Surface Normal Prediction					Depth Prediction				
				Error ↓		θ, within ↑			Error ↓		δ, within ↑		
		mIoU ↑	Pixel Acc ↑	Mean	Median	11.25°	22.5°	30°	Abs	Rel	1.25	1.25^2	1.25^3
Single-Task	3	27.5	58.9	17.5	15.2	34.9	73.3	85.7	0.62	0.25	57.9	85.8	95.7
Multi-Task	**1**	24.1	57.2	**16.6**	13.4	42.5	73.2	84.6	0.58	0.23	62.4	88.2	96.5
Cross-Stitch	3	25.4	57.6	17.2	14.0	41.4	70.5	82.9	0.58	0.23	61.4	88.4	95.5
Sluice	3	23.8	56.9	17.2	14.4	38.9	71.8	83.9	0.58	0.24	61.9	88.1	96.3
NDDR-CNN	3.15	21.6	53.9	17.1	14.5	37.4	**73.7**	**85.6**	0.66	0.26	55.7	83.7	94.8
MTAN	3.11	26.0	57.2	**16.6**	13.0	43.7	73.3	84.4	0.57	0.25	62.7	87.7	95.9
AdaShare	**1**	**30.2**	**62.4**	**16.6**	**12.9**	**45.0**	71.7	83.0	**0.55**	**0.20**	**64.5**	**90.5**	**97.8**

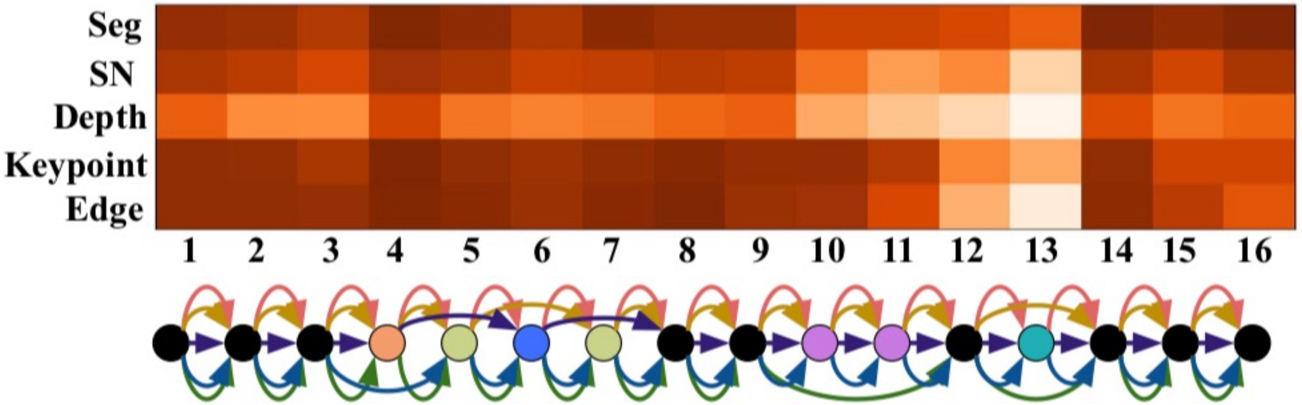




AdaShare: Experimental Results

- **Tiny-Taskonomy [5 Tasks].** AdaShare outperforms the baselines on 3 out of 5 tasks using less than 1/5 parameters of most baselines.

Models	# Params ↓	Seg ↓	SN ↑	Depth ↓	Keypoint ↓	Edge ↓
Single-Task	5	0.575	**0.707**	**0.022**	0.197	0.212
Multi-Task	**1**	0.587	0.702	0.024	0.194	0.201
Cross-Stitch	5	0.560	0.684	**0.022**	0.202	0.219
Sluice	5	0.610	0.702	0.023	**0.192**	0.198
NDDR-CNN	5.41	**0.539**	0.705	0.024	0.194	0.206
MTAN	4.51	0.637	0.702	0.023	0.193	0.203
AdaShare	**1**	0.566	**0.707**	0.025	**0.192**	**0.193**



Task2Sim: Towards Effective Pre-training and Transfer from Synthetic Data

Arxiv 2021


Samarth Mishra


Rameswar Panda


Cheng Phoo


Richard Chen


Leonid Karlinsly

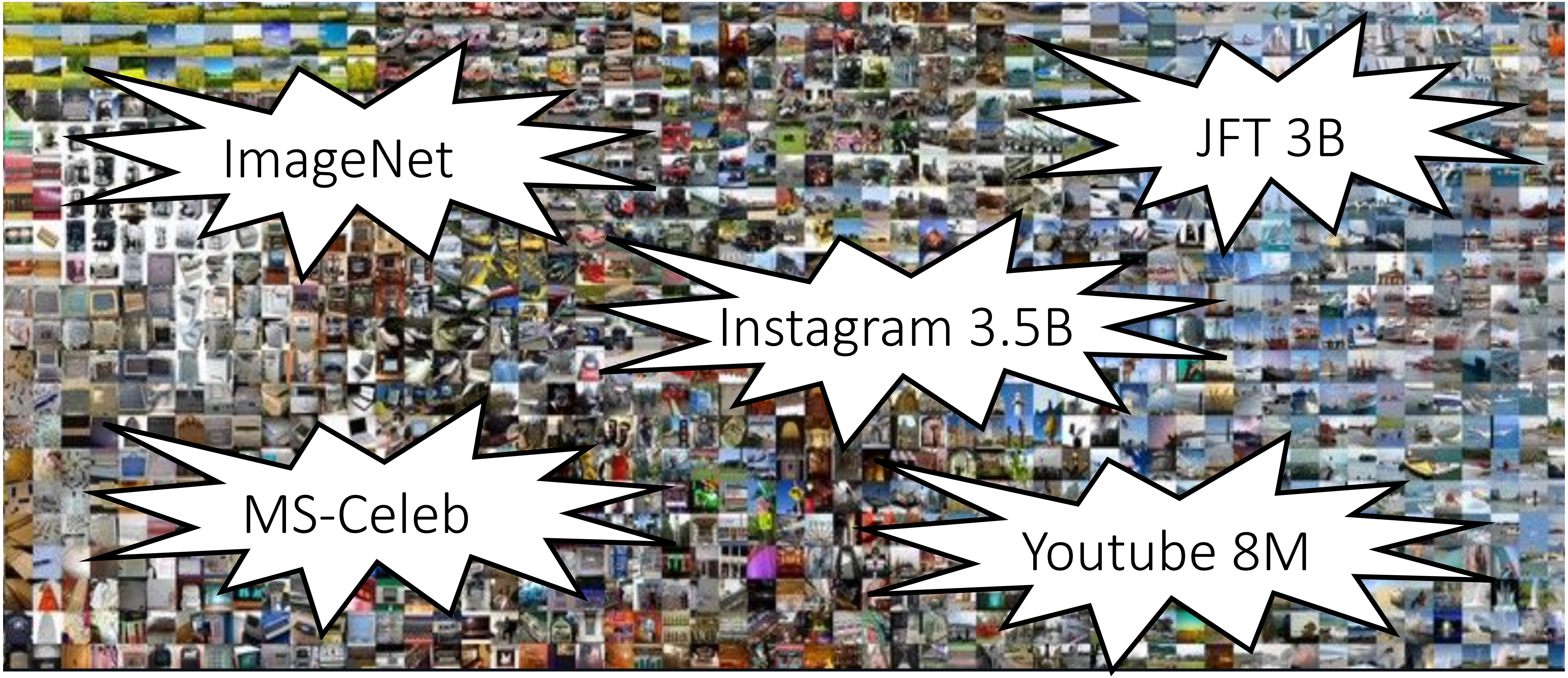

Kate Saenko


Venkatesh Saligrama


Rogerio Feris

Status Quo: Pre-train Models with Massive Datasets

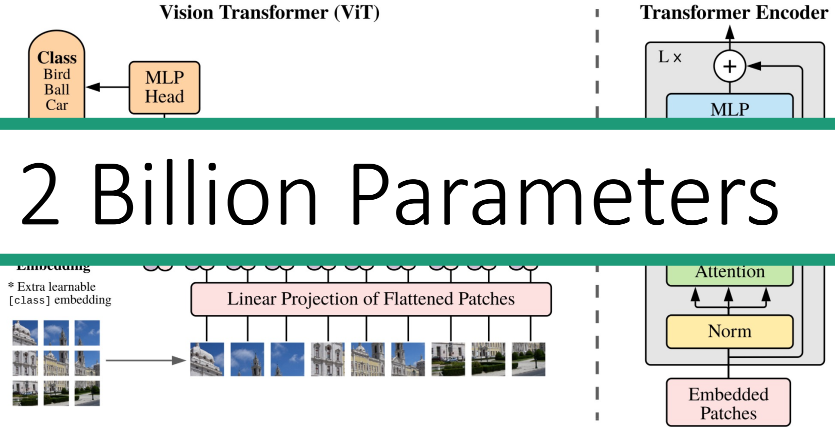
(Labeled/Unlabeled/Weakly-Labeled)



Larger Pre-training → Better Results

3 Billion Images (weakly labelled)

+



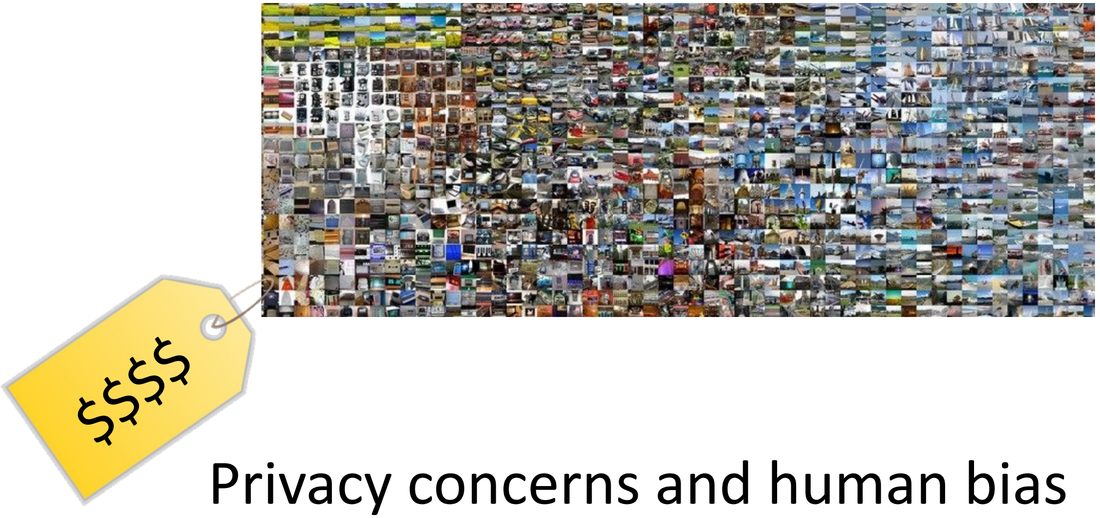
2 Billion Parameters

90.45% Top-1 Accuracy in ImageNet

Xiaohua Zhai et al. "Scaling Vision Transformers", Arxiv 2021

Issues with Large-scale Pre-training

Expensive Curation



Private Access

Google's JFT – 300 M

Facebook's IG – 1B

Privacy concerns and human bias

Issues with usage rights

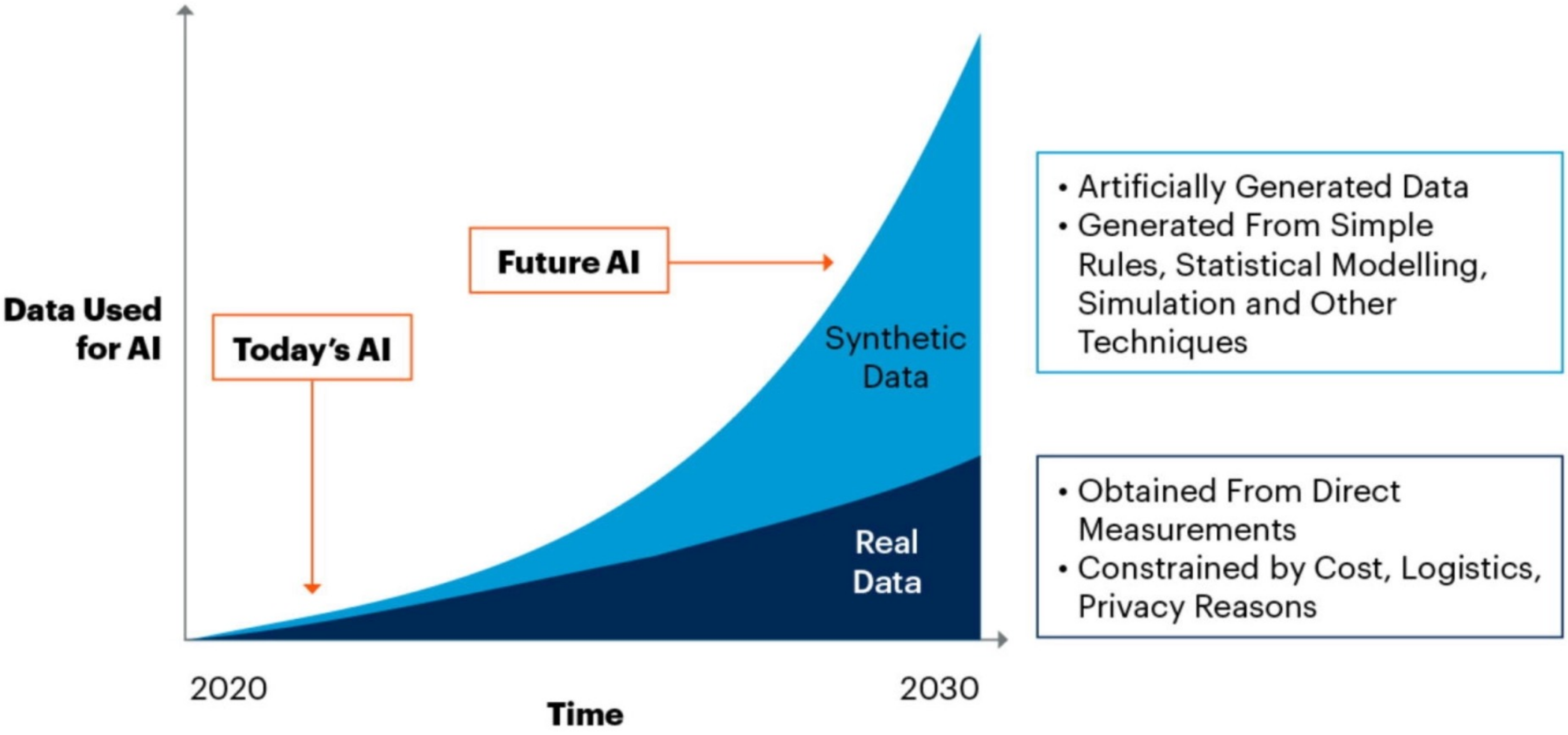
Promising way to address these issues: synthetic data

Face simulation

Embodied Perception

Autonomous Driving

By 2030, Synthetic Data Will Completely Overshadow Real Data in AI Models



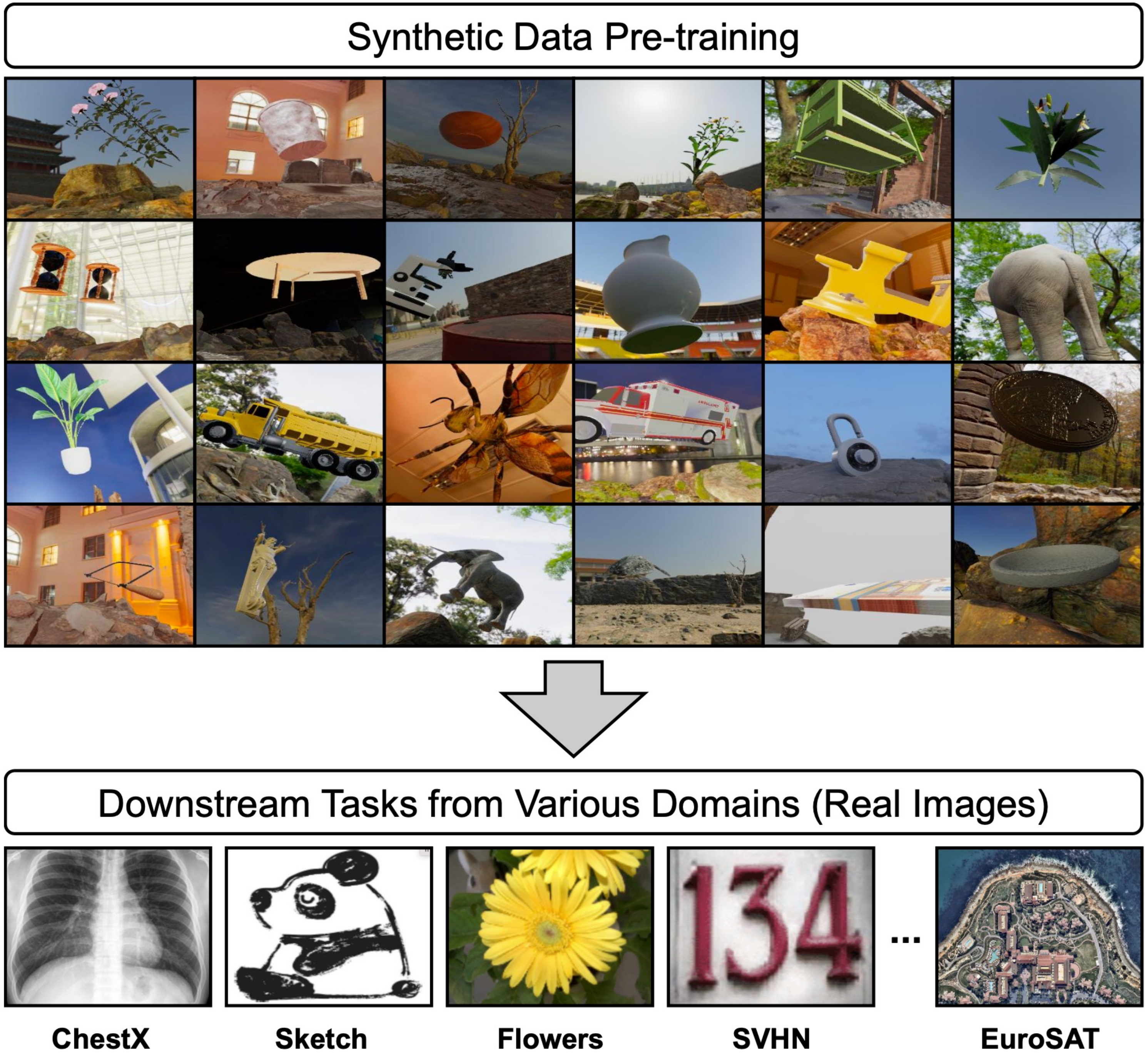
Source: Gartner

750175_C

- Reality Gap:
  Many works on Sim2Real domain adaptation

New Problem:

- Synthetic Data Pretraining and Transfer to Diverse Downstream Tasks

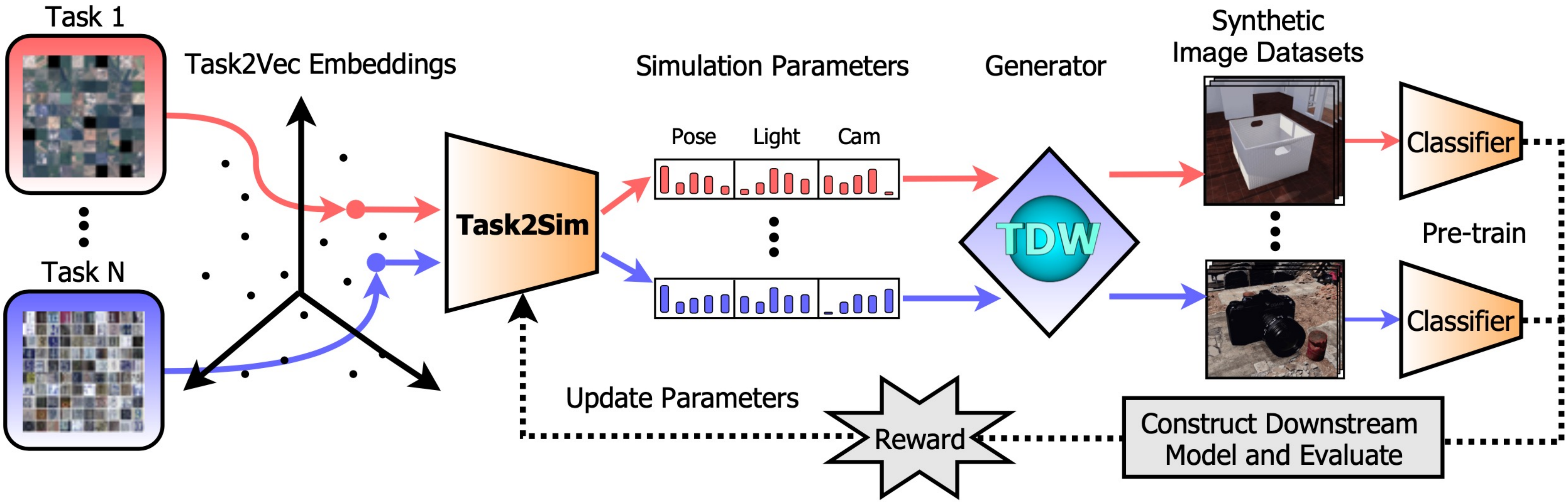


Observation: Different simulation parameters have different effects on different downstream tasks

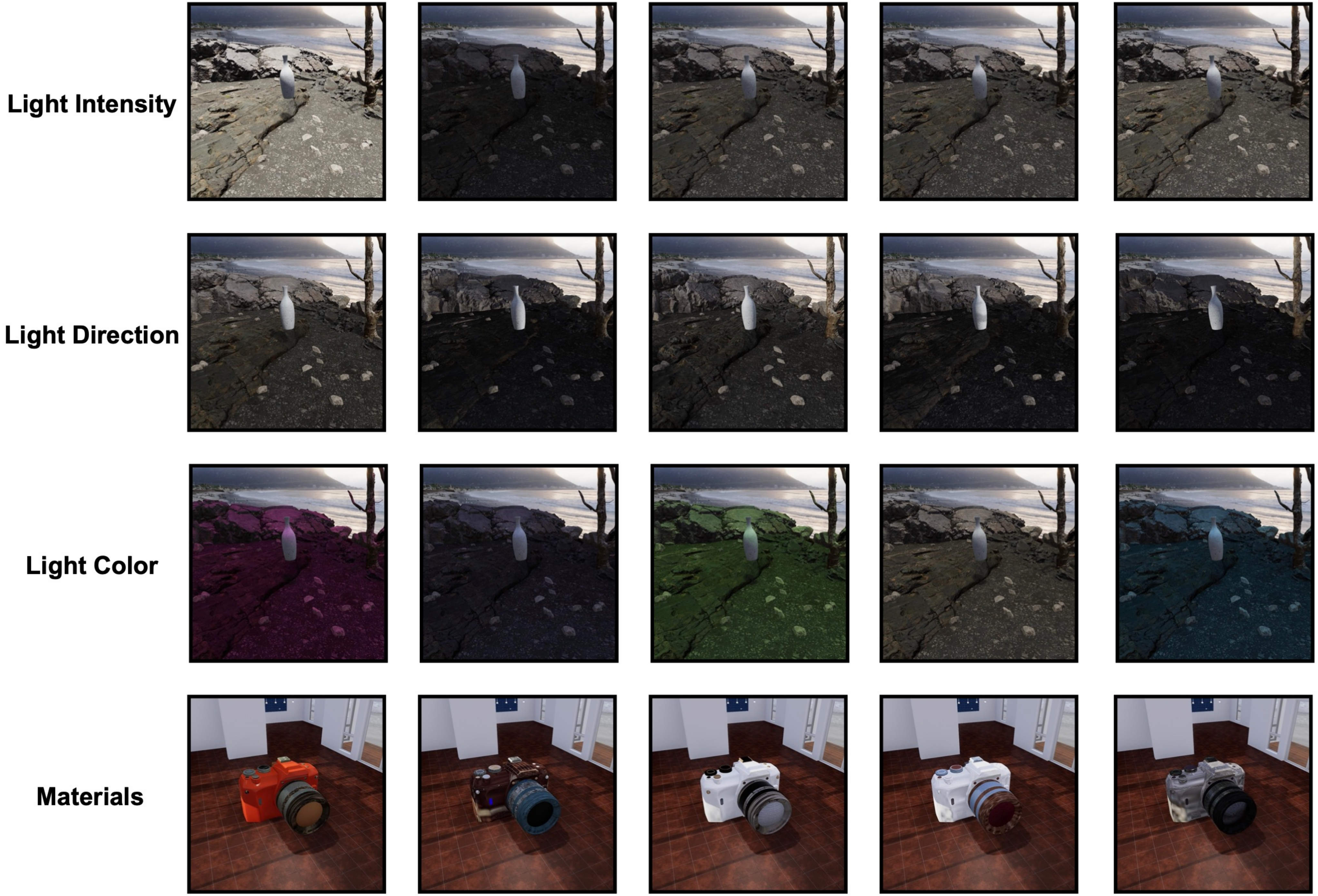
Resnet-50, linear probing

Pretraining Data Variations	Downstream Accuracy			
	EuroSAT	SVHN	Sketch	DTD
Pose	87.01	28.49	37.89	37.39
+Lighting	88.57	32.36	**38.81**	40.32
+Blur	**90.20**	35.58	35.53	37.66
+Materials	84.54	**44.84**	30.81	**38.51**
+Background	80.44	29.93	14.60	32.39

Proposed Approach: Task2Sim



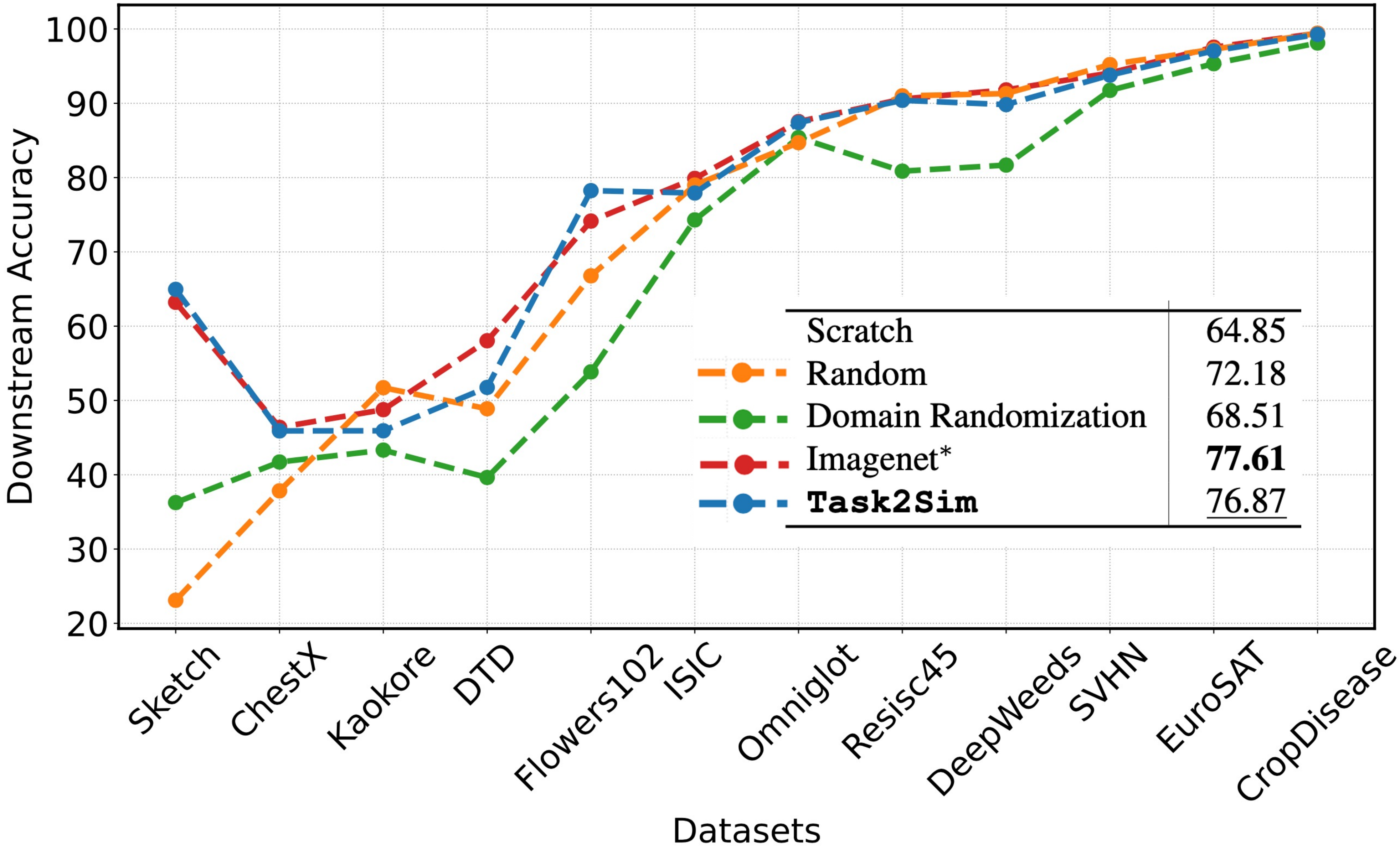

Obj Rotation
Cam Distance
Focus Blur
Background


Light Intensity
Light Direction
Light Color
Materials

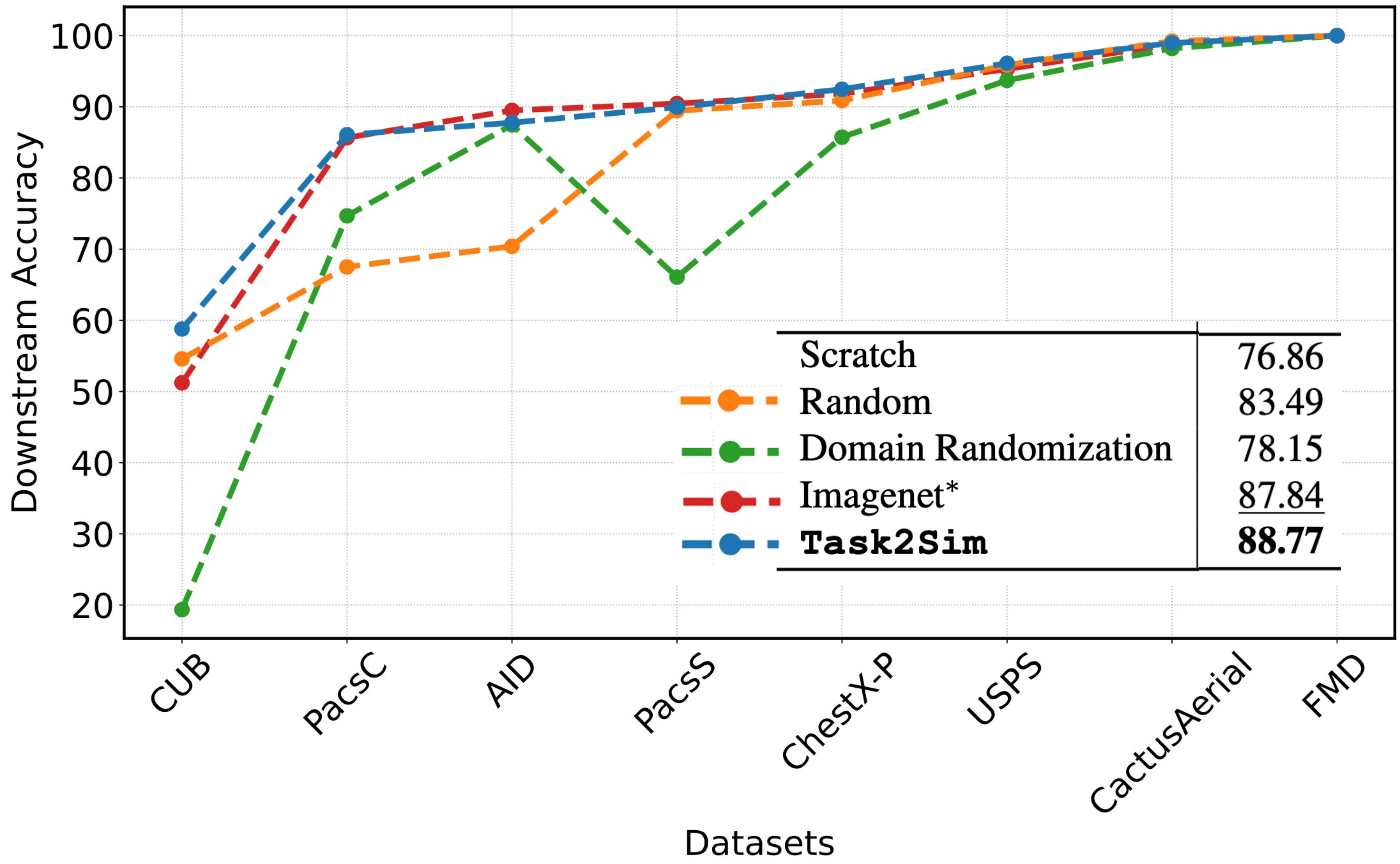
Experiments:
20 downstream tasks from various domains

Category	Dataset	Train Size	Test Size	Classes
Natural	CropDisease [39]	43456	10849	38
	Flowers [42]	1020	6149	102
	DeepWeeds [44]	12252	5257	9
	CUB [65]	5994	5794	200
Satellite	EuroSAT [18]	18900	8100	10
	Resisc45 [4]	22005	9495	45
	AID [75]	6993	3007	30
	CactusAerial [34]	17500	4000	2
Symbolic	Omniglot [30]	9226	3954	1623
	SVHN [40]	73257	26032	10
	USPS [21]	7291	2007	10
Medical	ISIC [7]	7007	3008	7
	ChestX [67]	18090	7758	7
	ChestXPneumonia [25]	5216	624	2
Illustrative	Kaokore [60]	6568	821	8
	Sketch [66]	35000	15889	1000
	PACS-C [32]	2107	237	7
	PACS-S [32]	3531	398	7
Texture	DTD [6]	3760	1880	47
	FMD [81]	1400	600	10

Fine-tuning - Seen Tasks (237 classes/100k images)



Fine-tuning - Unseen Tasks (237 classes/100k images)



Next Steps

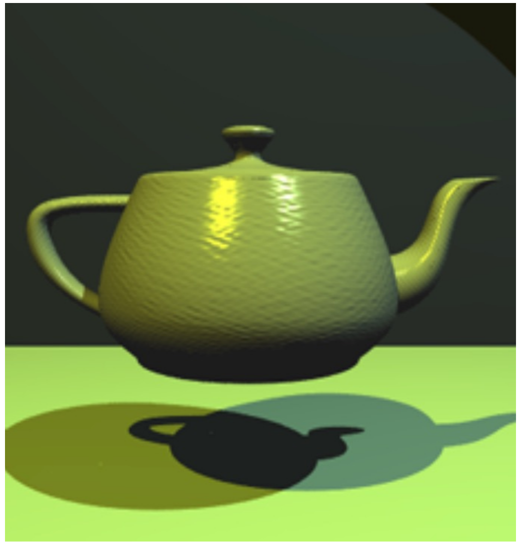
Label = Cat

Pretraining from Images with Labels

Label = ?

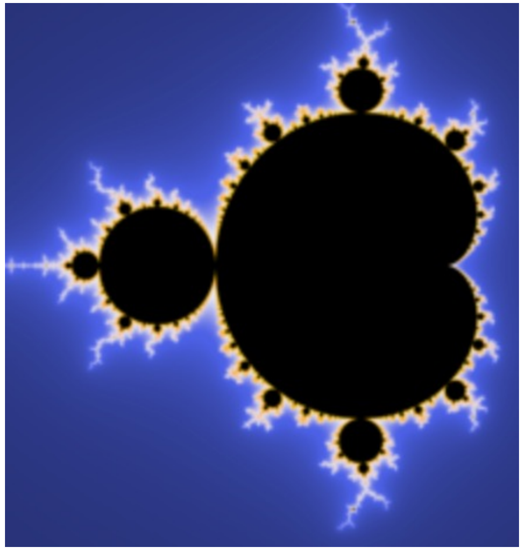
Pretraining from Images without Labels

Lighting = x,
Pose = y, ...



Pretraining from Synthetic Images

$$
\begin{aligned}
x_{k+1} &= x_k^2 - y_k^2 + \operatorname{Re} c \\
y_{k+1} &= 2x_k y_k + \operatorname{Im} c
\end{aligned}
$$

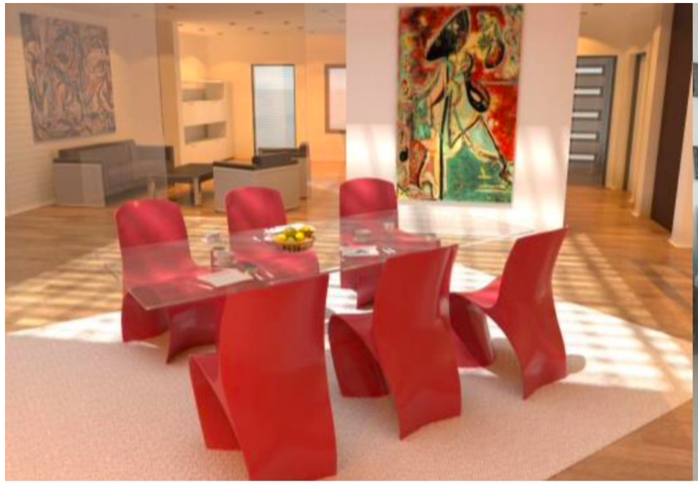


Pretraining from Fractals and Noise Processes

2012

Today

Multimodal Learning from Synthetic Data



Q/ How many chairs are in the room? A/ 6

Q/ What color is the bed cover? A/ white

Q/ Is there a dog in the kitchen? A/ no

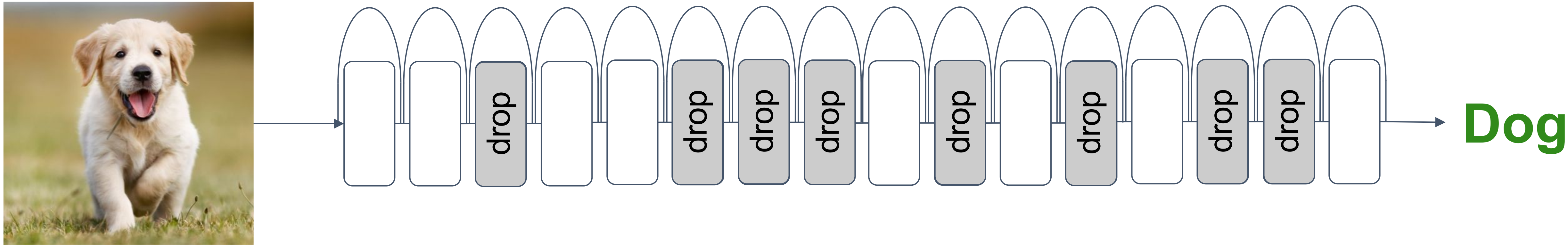
Q/ How many chairs are in the picture? A/ 2

Q/ What color is the fire hydrant? A/ yellow

Q/ Is there a teddy bear on top of the table? A/ yes

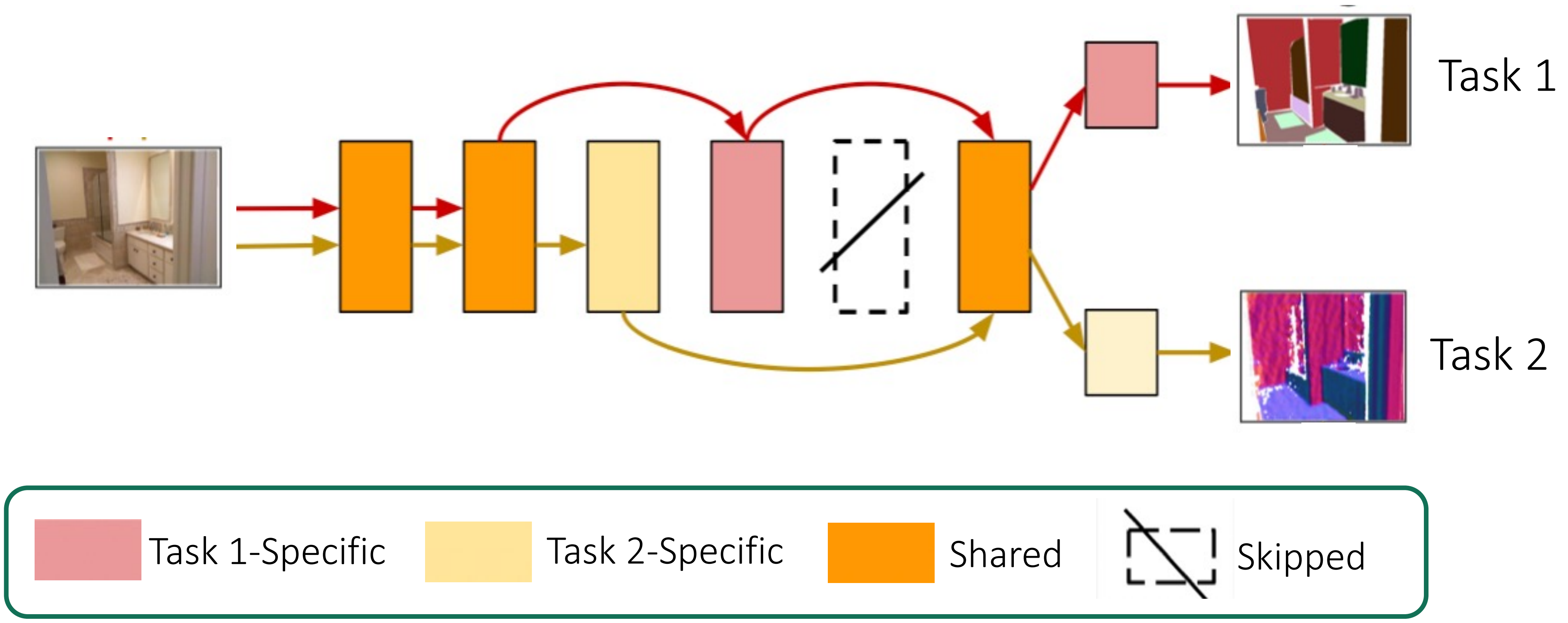
Summary

BlockDrop: Instance-specific Computational Pathways



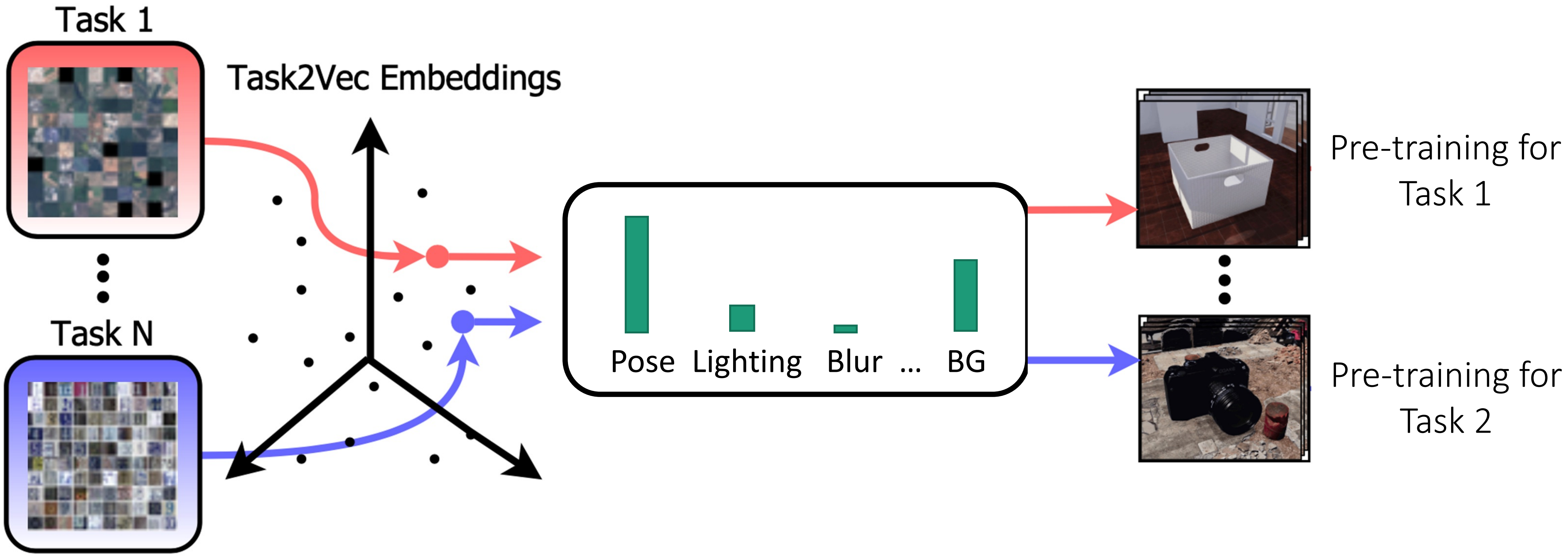
Summary

Adashare: Task-specific Computational Pathways



Summary

Task2Sim: Task-specific Data Simulation Pathways



References

- S. Mishra, R. Panda, C. Phoo, L. Karlinsky, K. Saenko, V. Saligrama, and R. Feris. Task2Sim: Towards Effective Pre-training and Transfer from Synthetic Data. Arxiv 2021 (soon)

- X. Sun, R. Panda, R. Feris, and K. Saenko. AdaShare: Learning What to Share for Efficient Deep Multi-Task Learning. NeurIPS 2020

- Z. Wu*, T. Nagarajan*, A. Kumar, S. Rennie, L. Davis, K. Grauman, and R. Feris. BlockDrop: Dynamic Inference Paths in Residual Networks. CVPR 2018 (* equal contribution)

See more at http://rogerioferis.org