Search to Distill: Pearls are Everywhere but not the Eyes

2021.12.10

Yu Liu*[1,3] Xuhui Jia[1] Mingxing Tan[2] Raviteja Vemulapalli[1] Yukun Zhu[1]
Bradley Green[1] Xiaogang Wang[3]

[1]Google AI [2]Google Brain
[3]Multimedia Laboratory, The Chinese University of Hong Kong
liuyuisanai@gmail.com
{xhjia, tanmingxing, ravitejavemu}@google.com

cvpr 2020

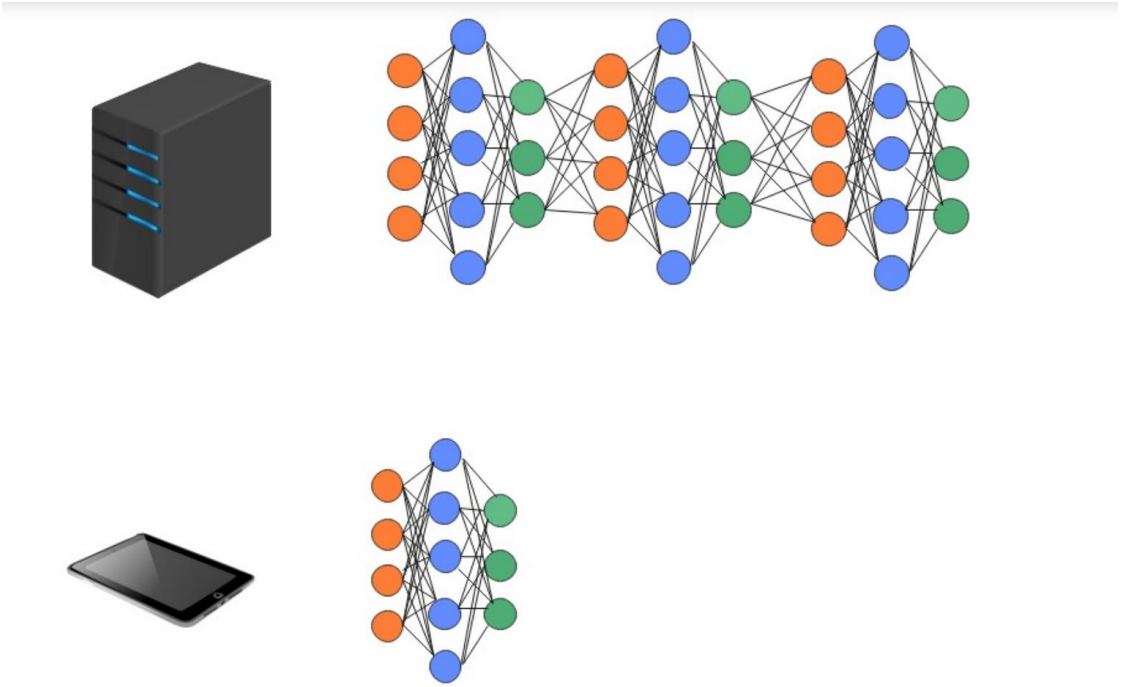
Background

Why do we need Knowledge distillation ?

Inconsistencies between training and deployment environments

During training, we prefer to use large-scale neural networks to fit massive data

When deployed, we prefer to use small networks to save memory and Increase speed



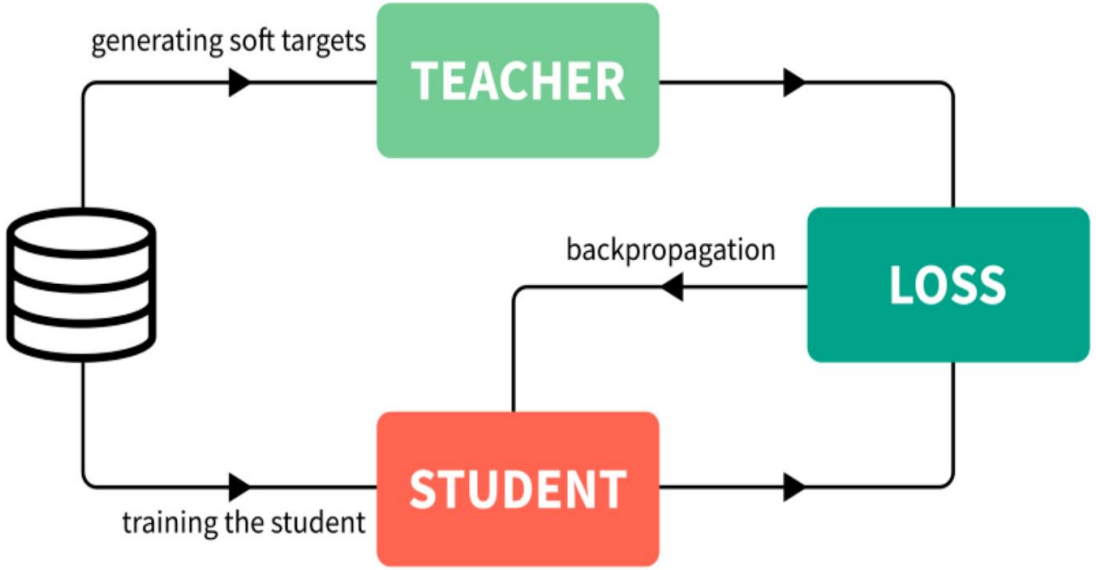
Background

What is Knowledge distillation ?

A large-scale model is pre-trained as teacher model, and the output q of the teacher model is the target of the student model

$$\boldsymbol{L = CE(y, p) + \alpha CE(q, p)}$$

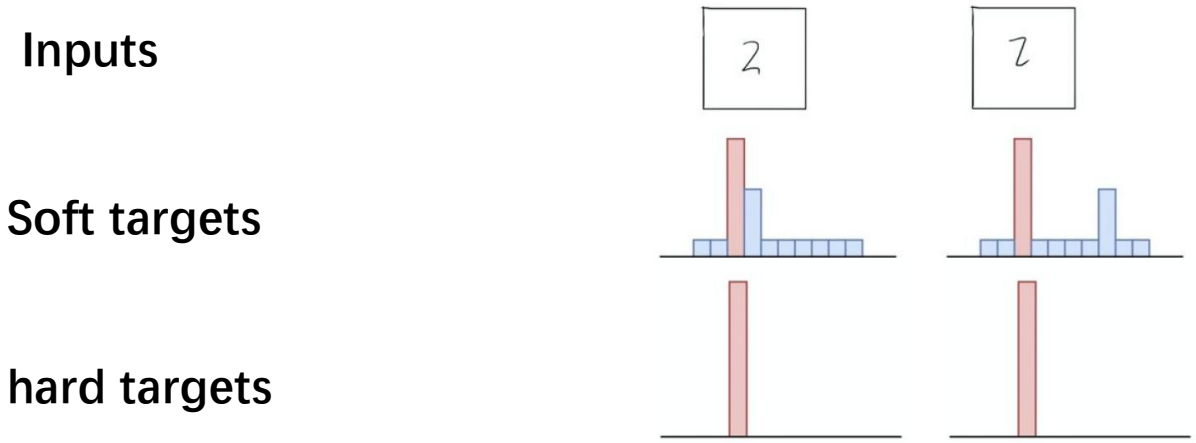
$$q_i = \frac{exp(z_i/T)}{\sum_j exp(z_j/T)}$$



Knowledge Distillation

Why dose Knowledge Distillation work?

Soft target provided by teacher model would have more information than hard target from ground truth



It is not considered from the perspective of network structure

Motivation

A confusing experimental results

Is the learning relative performance of student model related to the structure of the teacher model‘s architecture ?

Teachers	Student1	Student2	Comparison
EfficientNet-B7 [35]	65.8%	66.6%	student1 < student2
Inception-ResNet-v2 [32]	67.4%	66.1%	student1 > student2

Motivation

Large-scale experimental validation

Different teacher architectures favor different student architectures

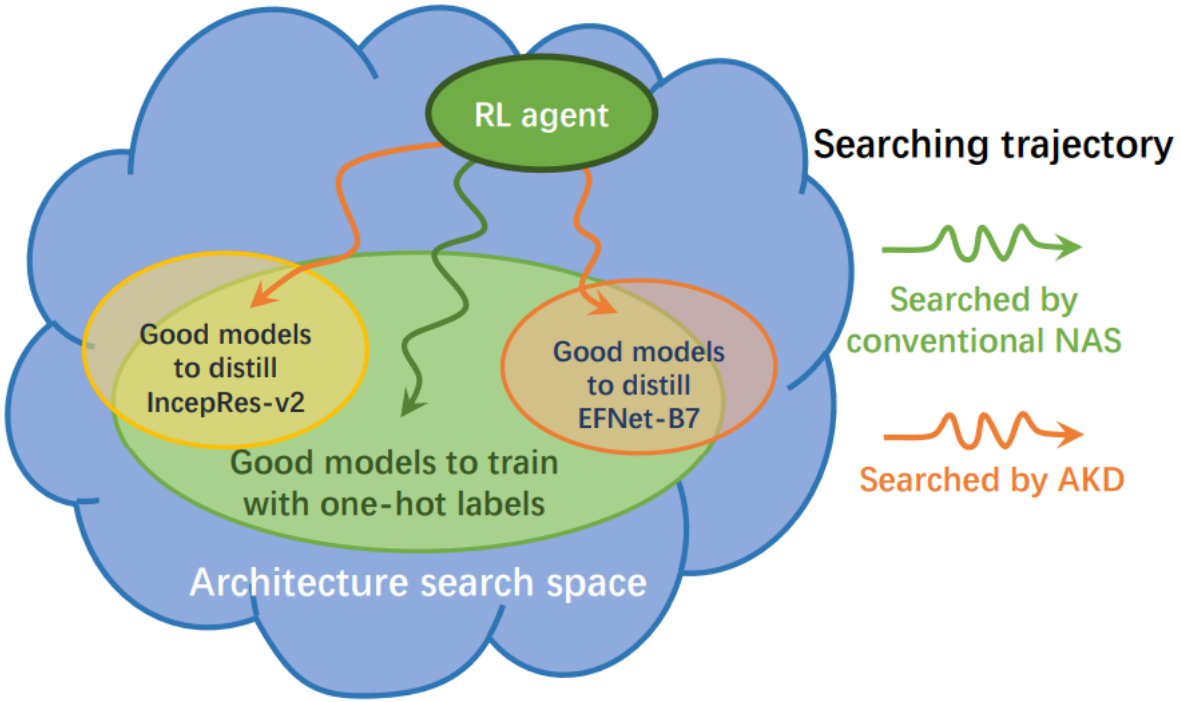
Tag	Model name	Input size	Top-1 accuracy
T(A)	EfficientNet-B7 [35]	600	84.4
T(B)	PNASNet-large [18]	331	82.9 74
T(C)	SE-ResNet-154 [11]	224	81.33
T(D)	PolyNet [41]	331	81.23
T(E)	Inception-ResNet-v2 [32]	299	80.217
T(F)	ResNeXt-101 [38]	224	79.431
T(G)	Wide-ResNet-101 [40]	224	78.84
T(H)	ResNet-152 [5]	224	78.31

	Teacher models			
GT	T(A)	T(B)	T(E)	T(F)
S_3 (65.6)	S_2 (66.6)	S_3 (66.9)	S_1 (67.4)	S_5 (67.1)
S_4 (65.6)	S_3 (66.5)	S_5 (66.4)	S_4 (67.0)	S_1 (67.1)
S_5 (65.5)	S_4 (66.3)	S_4 (66.1)	S_5 (66.9)	S_4 (66.6)
S_1 (65.5)	S_5 (66.0)	S_1 (65.7)	S_3 (66.5)	S_3 (66.3)
S_2 (65.4)	S_1 (65.8)	S_2 (65.4)	S_2 (66.1)	S_2 (66.0)

Idea

For a given teacher model, search the best student model structure suitable the teacher model , the student model can learn the structural knowledge of the teacher model to the maximum extent.

NAS: Neural Architecture Search
AKD: Architecture-aware Knowledge Distillation



NAS via RNN + RL

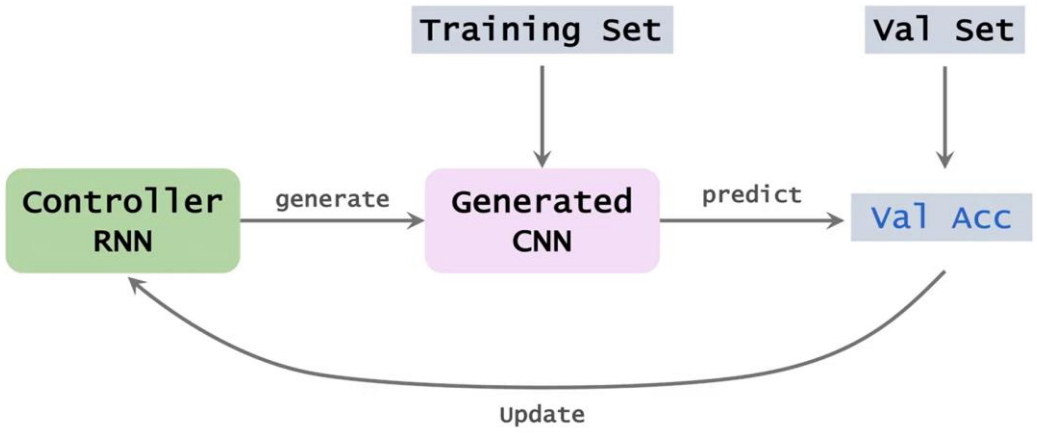
Toy example: using RNN to search a CNN structure with 20 Conv layers

Hyper-parameter Types	Candidates
# of filters	{24, 36, 48, 64}
size of filters	{3×3, 5×5, 7×7}
stride	{1, 2}

- Searching space: the set containing all the possible architectures

$$\{24, 36, 48, 64\}^{20} \times \{3 \times 3, 5 \times 5, 7 \times 7\}^{20} \times \{1, 2\}^{20}$$

How to train the controller RNN?



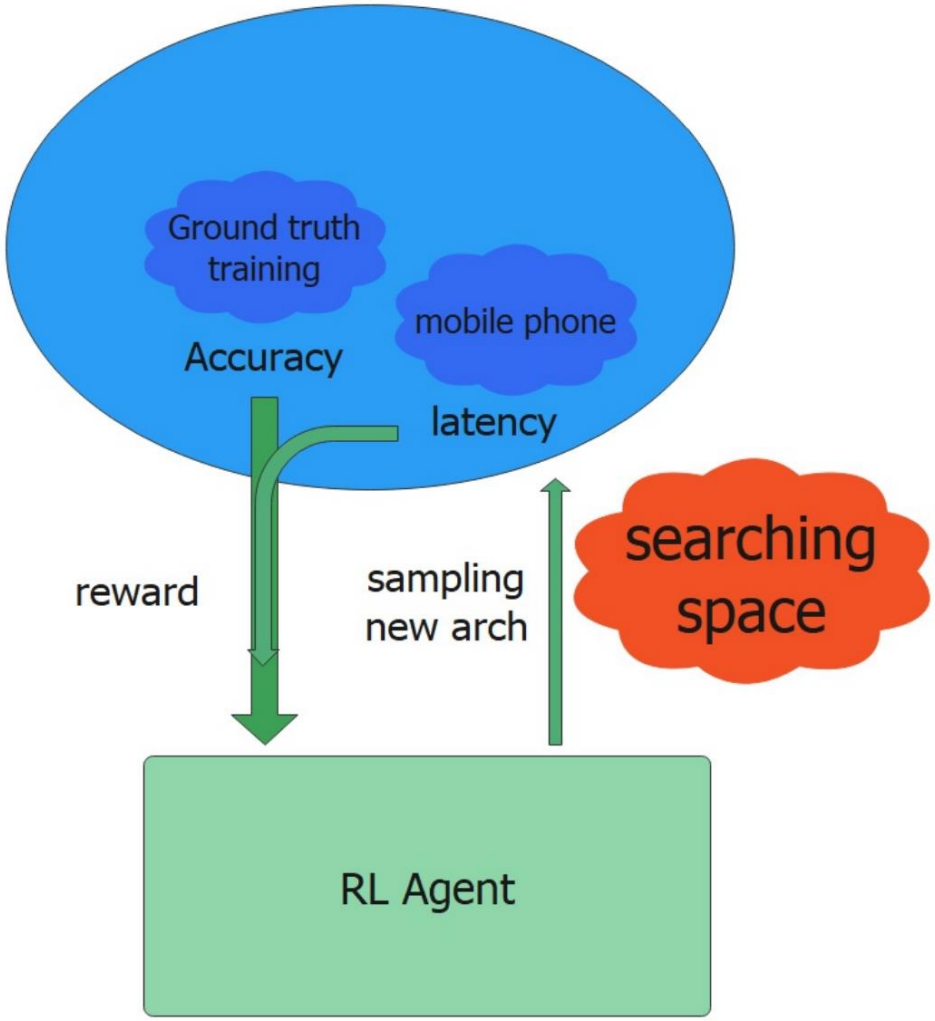
- r: objective function (to maximize). ⟷ Validation Accuracy
- θ: optimization variable. ⟷ Controller RNN Parameters

- Validation accuracy (r) is **not** a differentiable function of the controller RNN parameters (θ).

NAS via RNN + RL

Using RL to train RNN

- Objective : Improve the controller RNN so that validatio
- Rewards : validation accuracies + latency.
- Policy function: the controller RNN
- Improve the policy function by policy gradient ascent.



AKD

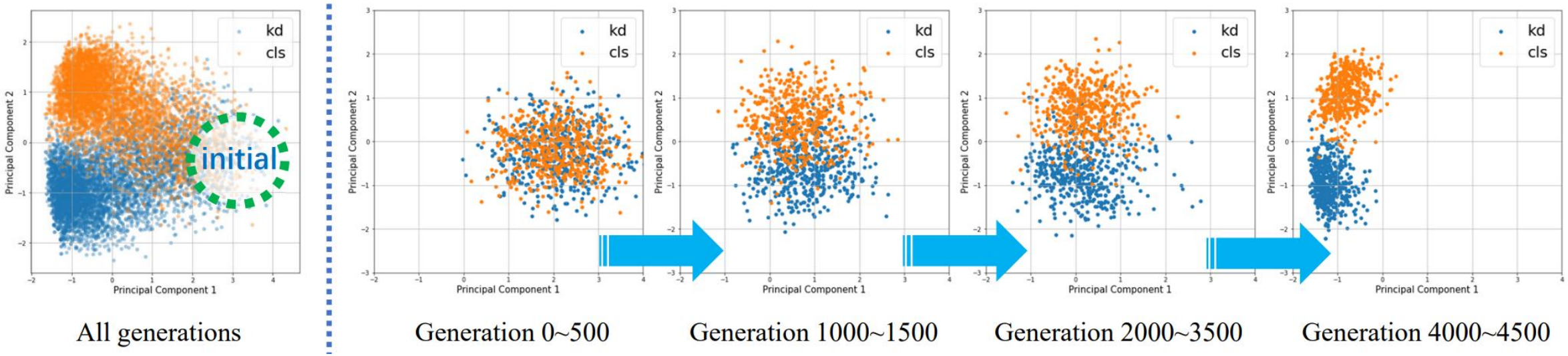
The difference between NAS and AKD is that AKD uses the teacher model's soft label to train the student model，while NAS lets the student model train directly on the training set.

Search process

Green circle : the structure initialization parameter of RNN represents

Orange dot: the Structure that NAS search for

Blue dot: the Structure that AKD search for



How to prove the existence of architecture knowledge ?

Neural network knowledge = learned parameters + architecture knowledge?

1. Do optimal architectures for training with GT and KD are same? No
2. Do optimal architectures of different teachers are same? No
3. Will tow different RL agent converge to the same architecture when distill the same model? Yes

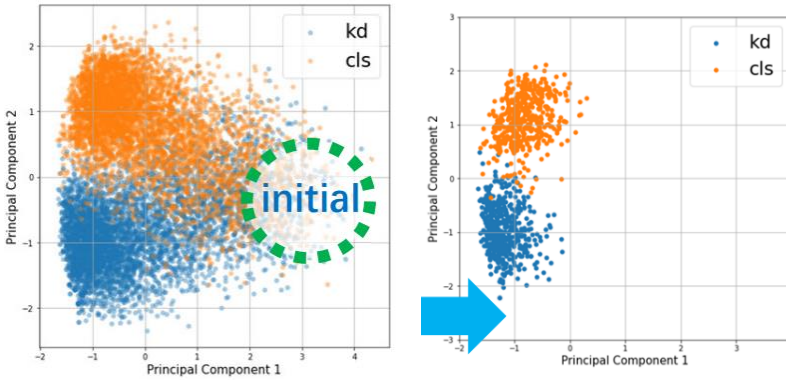


Figure 1

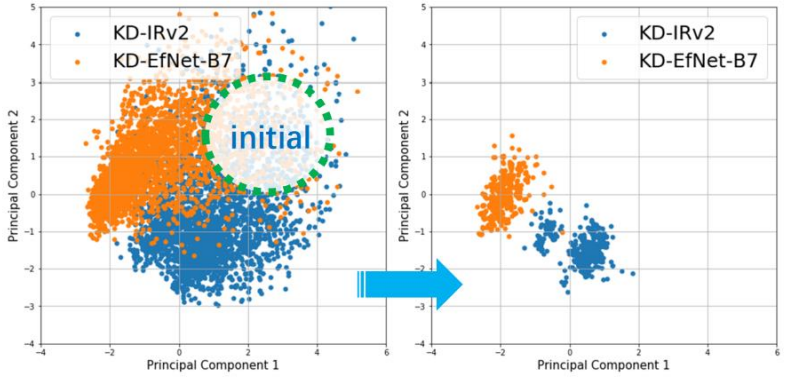
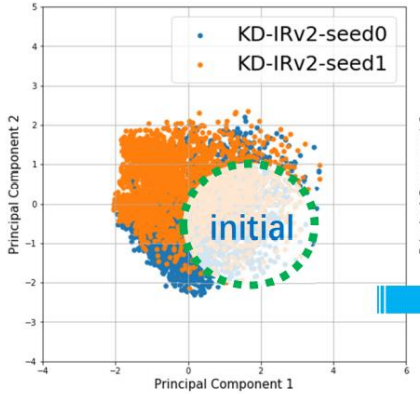


Figure 2



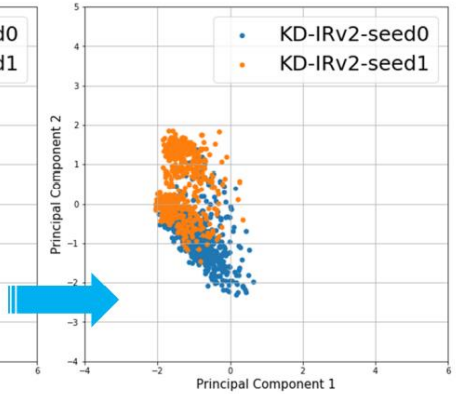


Figure 3

Performance

A metric to measure how AKD improves KD result

$$[\mathrm{KD(AKDNet)} - \mathrm{CLS(AKDNet)}] - [\mathrm{KD(NASNet)} - \mathrm{CLS(NASNet)}], \quad (1)$$

15±1 ms : (66.5 -61.4) – (63.9 – 57.7) = 0.93

Latency	searching by	training by	top-1	top-5
15±1 ms	hard label	hard label	59.73	81.39
	hard label	distillation	63.9	84.26
	distillation	hard label	61.4	83.1
	distillation	**distillation**	**66.5**	**87.5**
25±1 ms	hard label	hard label	67.0	87.4
	hard label	distillation	68.1	88.0
	distillation	hard label	67.2	87.5
	distillation	**distillation**	**69.6**	**89.1**
75±1 ms	hard label	hard label	73.0	92.1
	hard label	distillation	74.7	92.54
	distillation	hard label	73.6	92.2
	distillation	**distillation**	**75.5**	**93.1**

Compare with SOTA architectures

Latency	architecture	with KD?	top-1	top-5
15∼20 ms	AKDNet		61.4	83.1
	AKDNet	✓	↑2.6	↑3.24
	AKDNet	RCO-KD	↑3.1	↑3.8
	MNet-v2-a		59.2	79.8
	MNet-v2-a	✓	↑1.4	↑2.1
	MNASNet-a		62.2	83.5
	MNASNet-a	✓	↑1.49	↑2.3
	MNet-v3-a		64.1	85.0
	MNet-v3-a	✓	↑1.3	↑ 2.2
25∼27 ms	AKDNet		67.2	87.5
	AKDNet	✓	↑2.4	↑1.6
	AKDNet	RCO-KD	↑2.8	↑1.5
	MNASNet-b		66.0	86.1
	MNASNet-b	✓	↑1.1	↑0.6

Thanks