



The University of Sydney

From Manual Design to Automatic

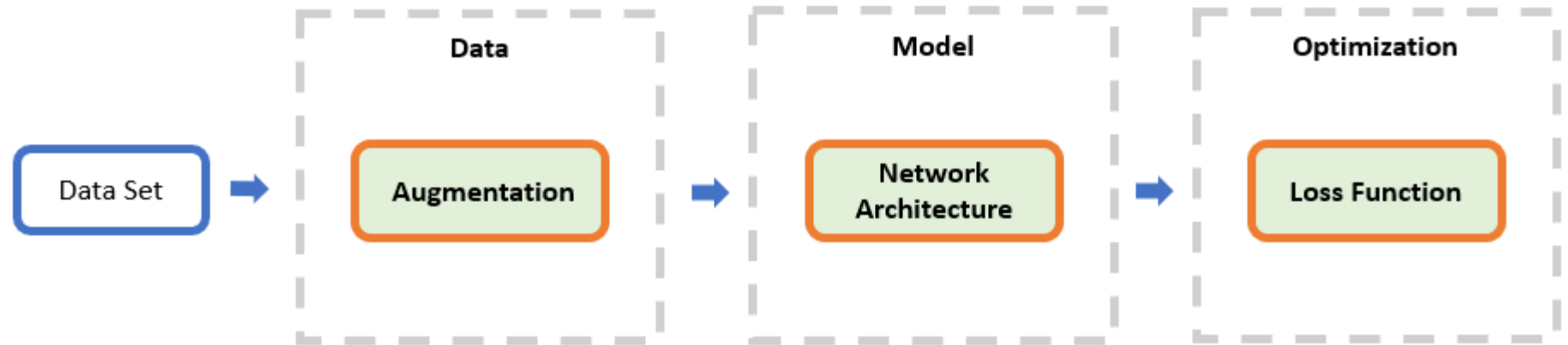
Deep Learning

Wanli Ouyang

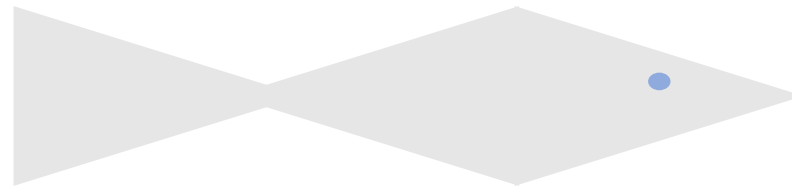
Outline

- Introduction

- Auto-ML



- Manual design



- Conclusion

Outline

- Introduction

Image classification

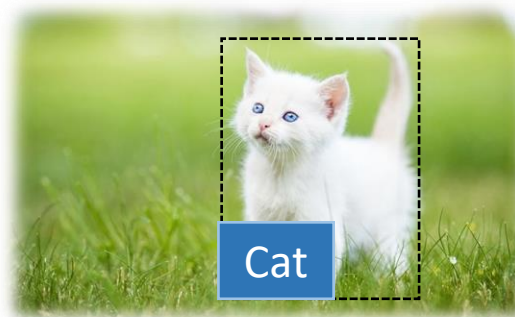
Cat



Dog



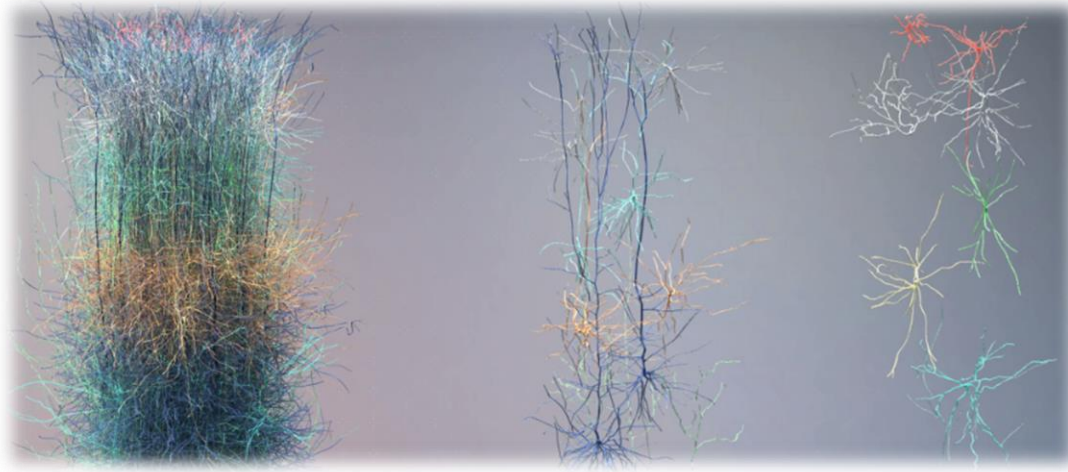
Object detection



DOG, DOG, CAT

Deep learning

Simulate brain activities and employ **millions of neurons** to fit **billions of training samples**. Deep neural networks are trained with GPU clusters with **tens of thousands of processors**



MIT Tech Review
Top 10 Breakthroughs 2013
Ranking No. 1

Deep Learning

With massive amounts of computational power, machines can now recognize objects and translate speech in real time. Artificial intelligence is finally getting smart.



REVOLUTIONARY

Web-scale visual search, self-driving cars, surveillance, multimedia ...

Hold records on most of the computer vision problems

Biology, Physics, Chemistry, Remote Sensing, ...

Turing award 2018



From Andrej Karpathy's Twitter (09/Jan/2020)

- A recent post on Reddit: "Is it theoretically possible to do object recognition with classification algorithms other than NN's?".
- Just ~8 years ago you'd be more likely to find "Is it theoretically possible to do object recognition with NN's?". That was a fun few years.

Deep learning vs non-deep learning

- Automatically learn features from data

Deep Learning – What's Next?

Auto-ML

Deep learning vs non-deep learning

- Automatically learn features from data

Deep learning – not fully automatic

- Automatically learn features from data



Achieved by deep learning

- Data augmentation?

- Loss function?

- Number of layers?

- What kind of operation in each layer?

- How one layer is connected to another layer?

- Number of channels at each layer?

- ...



Manual tuning is required

Automatically learning them is possible by

AutoML

Auto-ML

- The problem of automatically (without human input) producing test set predictions for a new dataset within a fixed **computational budget** [a].
- Target: low error rate with low computational budget

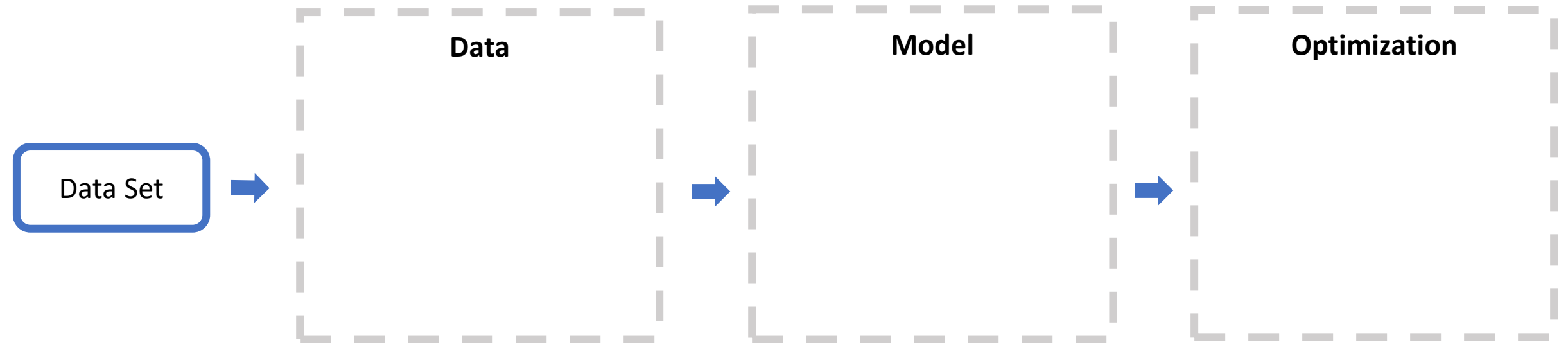
Architecture	GPU Days	Error Rate	Method
NASNet-A [b]	1800	2.65	Reinforcement Learning
AmoebaNet-A [c]	3150	3.34	Evolution

[a] Feurer, Matthias, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. "Efficient and robust automated machine learning." In *Advances in neural information processing systems*, pp. 2962-2970. 2015.

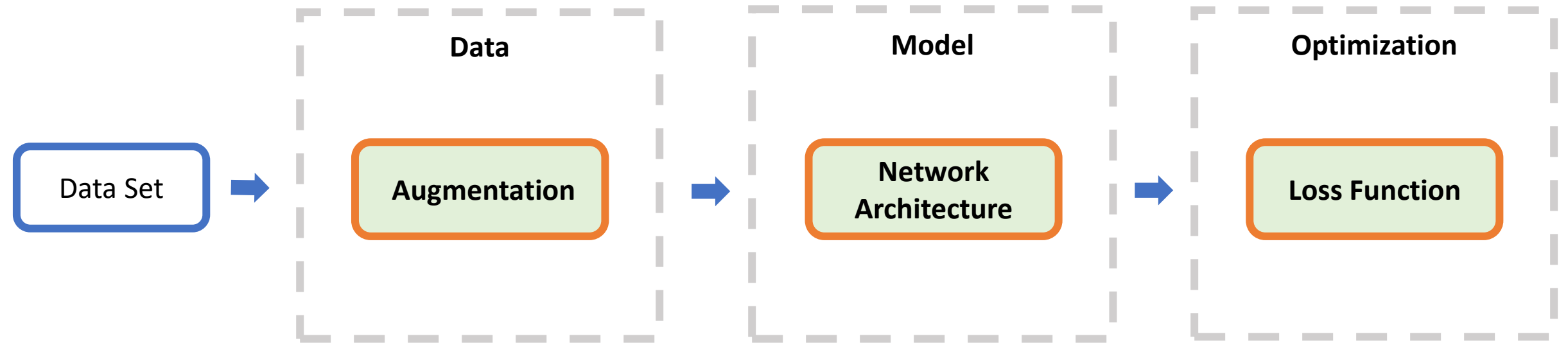
[b] Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V.: Learning transferable architectures for scalable image recognition. In: CVPR (2018)

[c] Shah, Syed Asif Raza, Wenji Wu, Qiming Lu, Liang Zhang, Sajith Sasidharan, Phil DeMar, Chin Guok et al. "AmoebaNet: An SDN-enabled network service for big data science." *Journal of Network and Computer Applications* 119 (2018): 70-82.

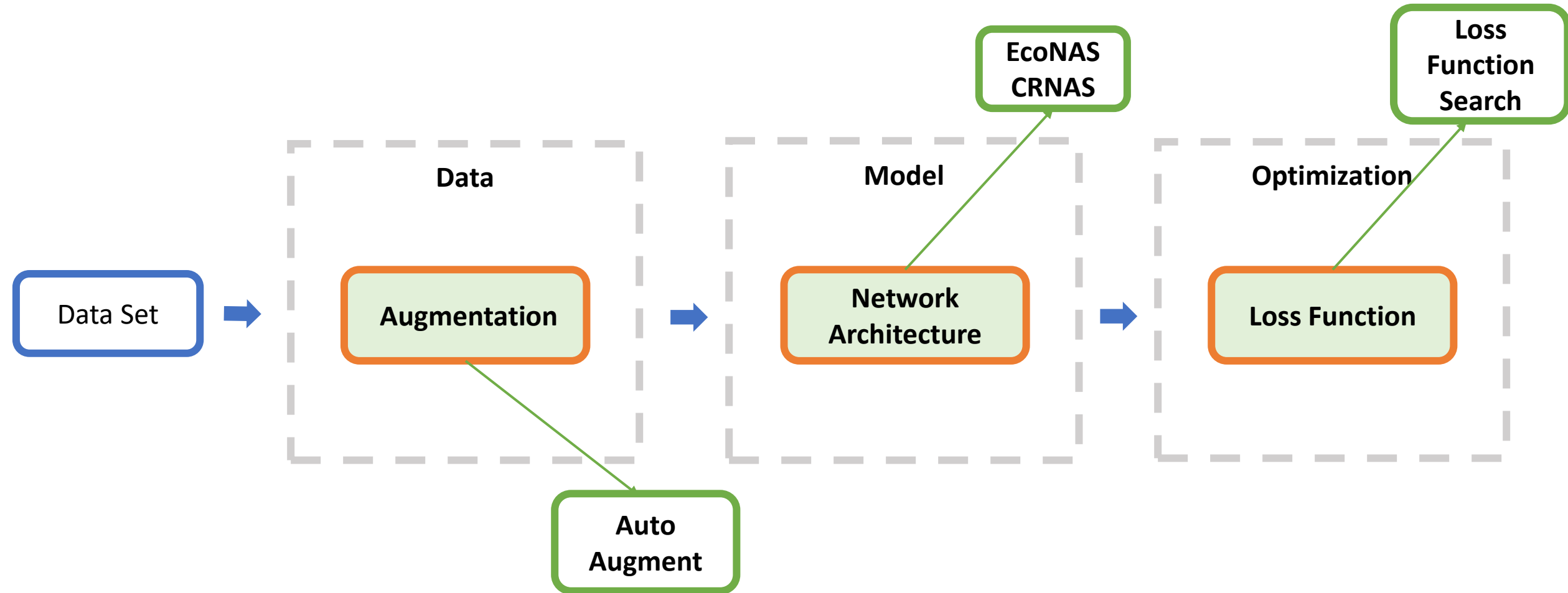
Pipeline of Machine Learning



Pipeline of Deep Learning

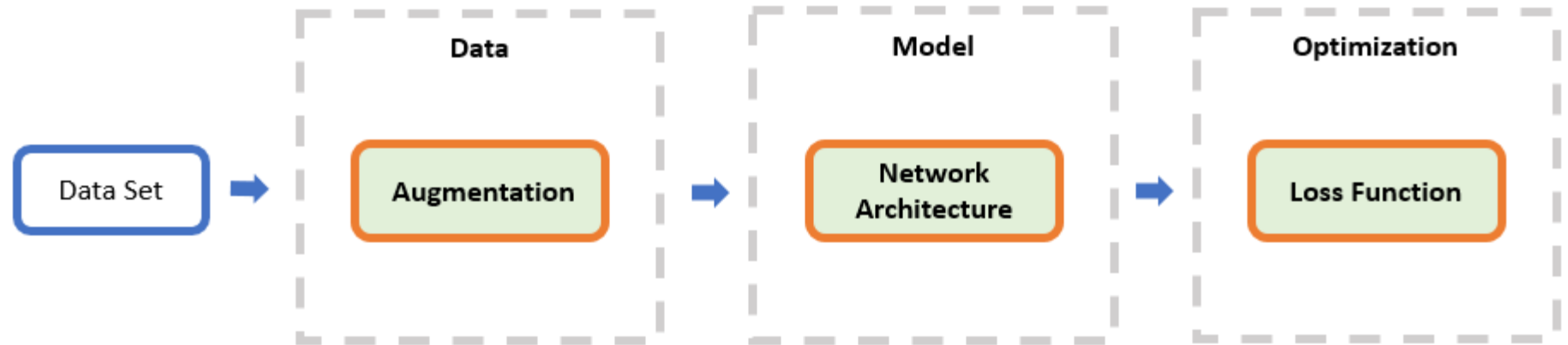


Towards Auto Training System

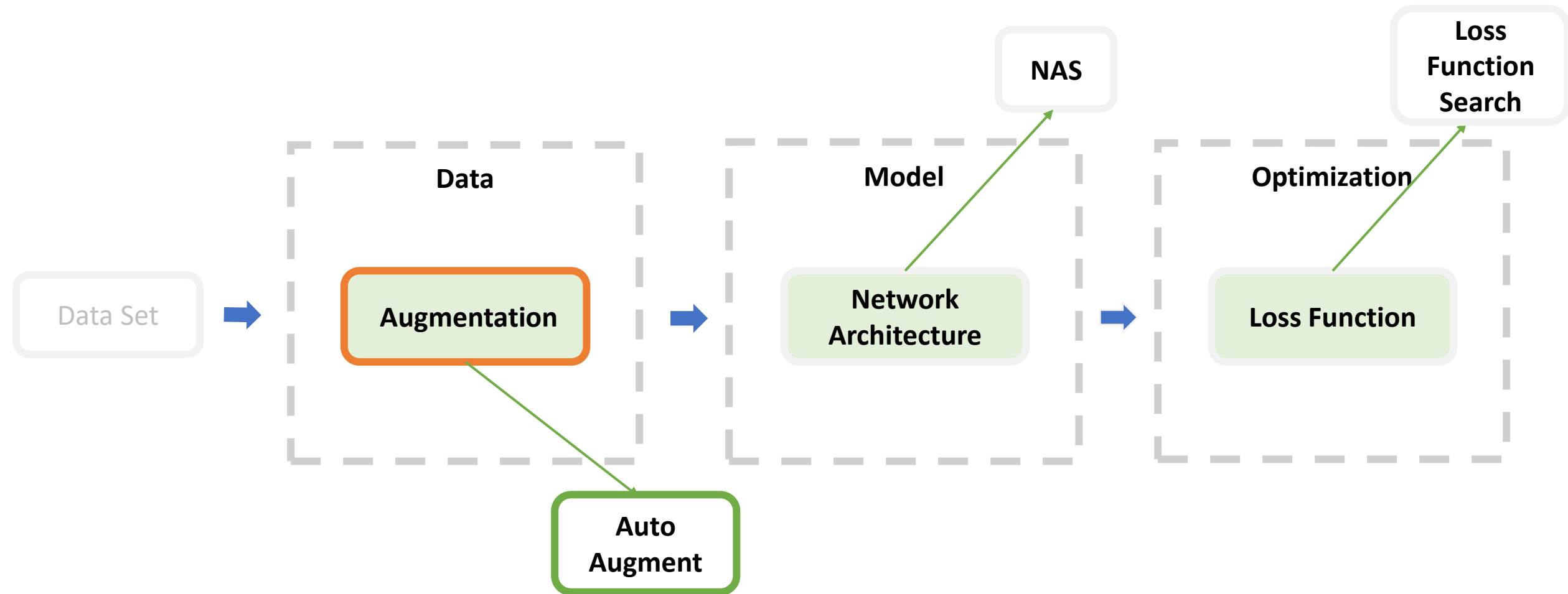


Outline

- Auto-ML



Towards Auto Training System



Online Hyper-parameter Learning for Auto-Augmentation Strategy

Lin, Chen, Minghao Guo, Chuming Li, Wei Wu, Dahua Lin, Wanli Ouyang,
and Junjie Yan

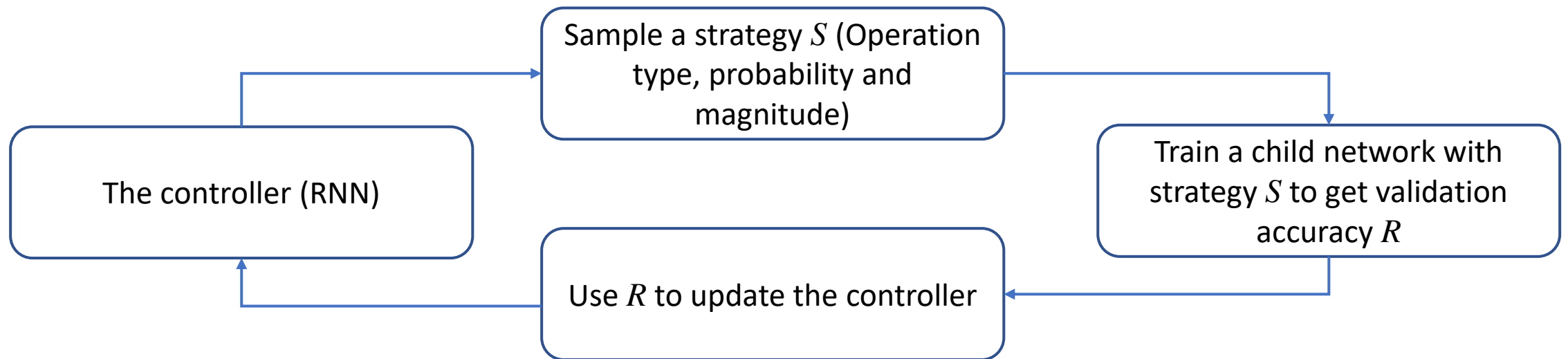
ICCV 2019

Auto-augment search – Existing work

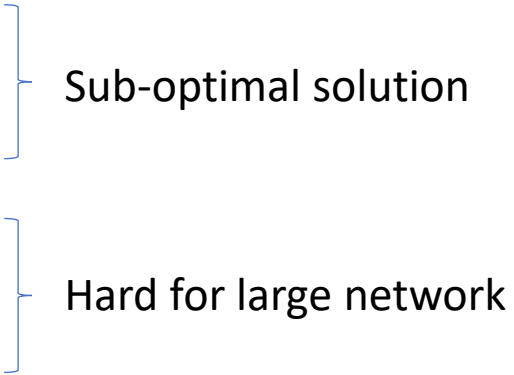
- Search policy on a subsampled dataset and a predefined CNN



- Step 1: The controller generates an augmentation strategy S
- Step 2: Parameters of the CNN are learned using the augmentation strategy S
- Step 3: Get the validation accuracy R for the model with the strategy S
- Step 4: Use the accuracy as the reward for learning the controller



Auto-augment search – Motivation

- Difficulty:
 - Slow evaluation of augmentation policy
 - Slow convergence of RL due to the RNN controller
 - Have to resort to small amount of data:
 - CIFAR-10: 8% subsampled
 - IMAGENET: 0.5% subsampled
 - Have to use very small network:
 - CIFAR-10: WideResNet-40-2 (small)
 - IMAGENET: Wide-ResNet 40-2
 - Solution: Treat augmentation policy search as a hyper-parameter learning
- 

Cubuk, Ekin D., et al. "Autoaugment: Learning augmentation policies from data." *arXiv preprint arXiv:1805.09501* (2018).

Lin, Chen, Minghao Guo, Chuming Li, Wei Wu, Dahua Lin, Wanli Ouyang, and Junjie Yan. "Online Hyper-parameter Learning for Auto-Augmentation Strategy." *ICCV19*.

Hyperparameter Learning

- To learn the hyperparameters from data
- Hyperparameters can be
 - Sampling strategy
 - Loss weights
- Different from CNN architecture
 - CNN architecture is transferable across different dataset
 - Hyper-parameters in training strategy are KNOWN to be deeply coupled with specific dataset and underlying network architecture.

Hyperparameter Learning -- Challenges

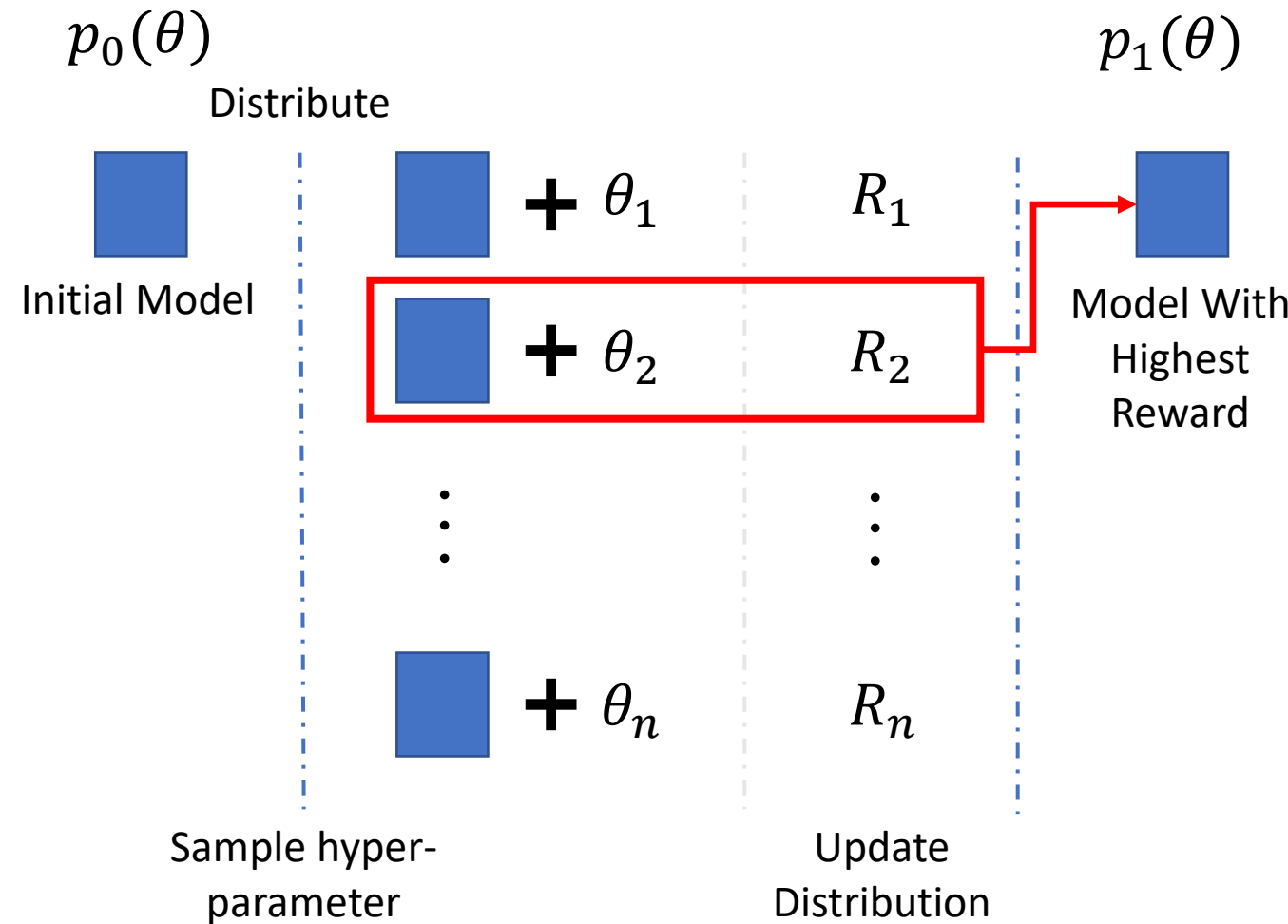
- Usually the hyper-parameters are not differentiable wrt validation loss.
- Full evaluation-based method using reinforcement learning, evolution, or Bayesian optimization is **computationally expensive** and implausible to be applied on industrial-scaled dataset.

Our Solution: Online Hyperparameter Learning (OHL)

- What is OHL
 - Online Hyper-parameter Learning aims to learning the best hyper-parameter within only a **single** run.
 - While learning the hyper-parameters, it improves the performance of the model at mean time.

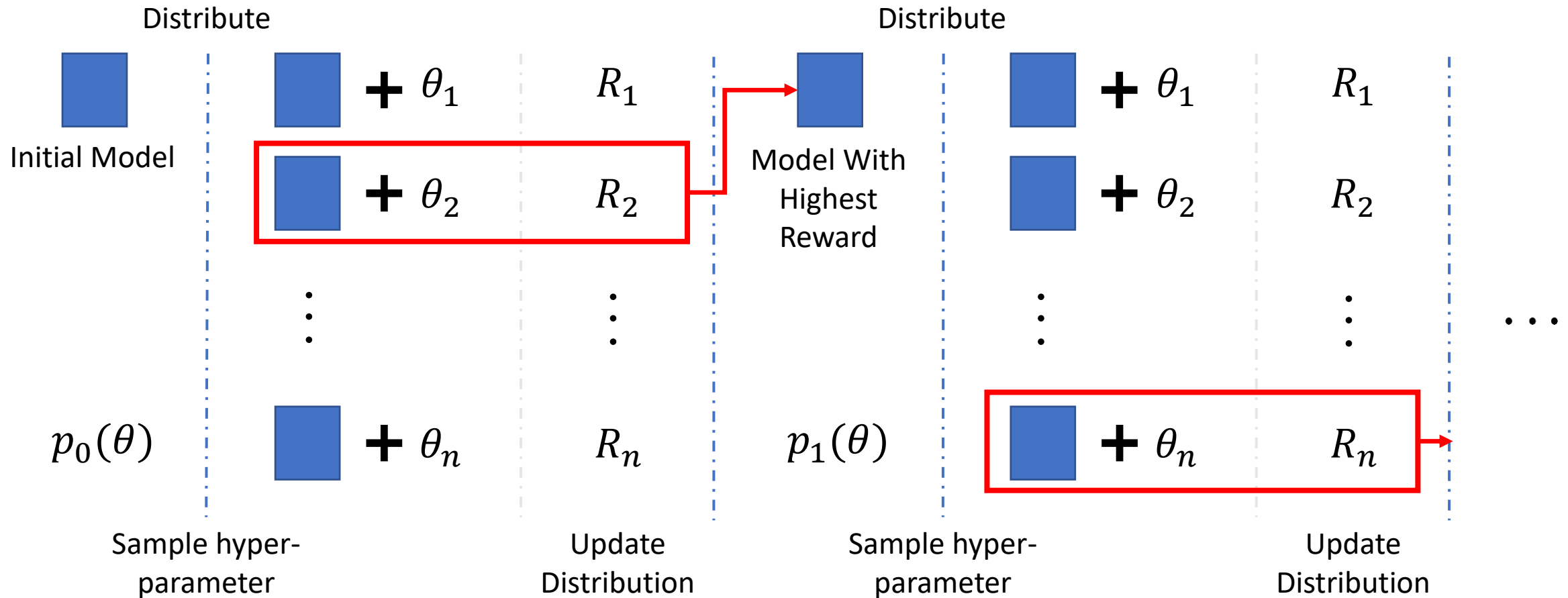
Our Solution: Online Hyperparameter Learning (OHL)

- Hyper-parameter is modeled as random variables.
- Split the training stage into trunks
- Run multiple copies of current model, with different sampled hyper-parameters.
- At the end of each trunk, we compute the reward of each copy by its performance on validation set.
- Update the hyper-parameter distribution using RL.
- Distribute the best performing model



Our Approach: Online Hyperparameter Learning (OHL)

$p_0(\theta)$: Initial Distribution



Augmentation as hyperparameter

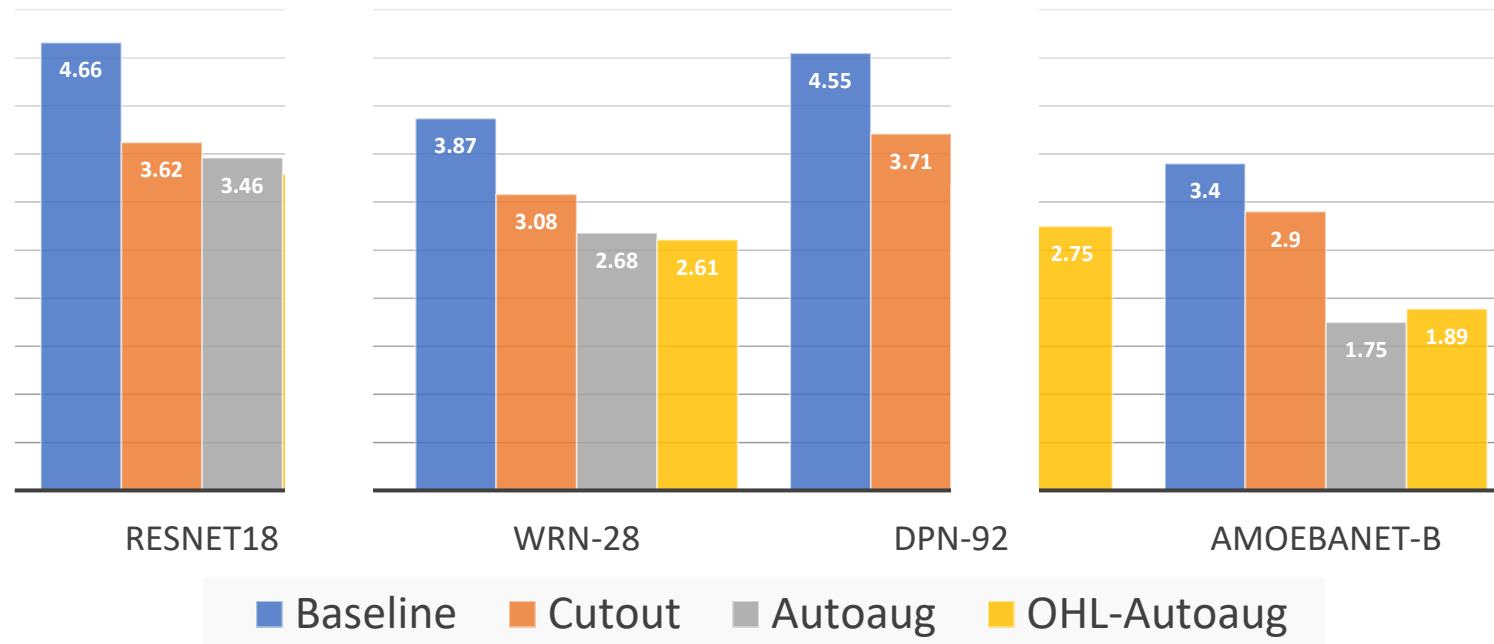
- For fair comparison, we apply the same search space with original auto-augment, with minor modification

Elements Name	Range of magnitude
Horizontal Shear	{0.1, 0.2, 0.3}
Vertical Shear	{0.1, 0.2, 0.3}
Horizontal Translate	{0.15, 0.3, 0.45}
Vertical Translate	{0.15, 0.3, 0.45}
Rotate	{10, 20, 30}
Color Adjust	{0.3, 0.6, 0.9}
Posterize	{4.4, 5.6, 6.8}
Solarize	{26, 102, 179}
Contrast	{1.3, 1.6, 1.9}
Sharpness	{1.3, 1.6, 1.9}
Brightness	{1.3, 1.6, 1.9}

- Each augmentation is a pair of operations eg.
 - (HorizontalShear0.1, ColorAdjust0.6)
 - (Rotate30, Contrast1.9)
 - ...
- In a stochastic point of view, the augmentation is a random variable:
 - $p_{\theta}(Aug)$
 - α is the weight parameter controls augmentation distribution.
- Learning augmentation strategy is learning θ

Experimental Results - CIFAR10

- Using OHL, we train our performance model while learning alpha at the same time.
 - On CIFAR10 (Top1 Error)

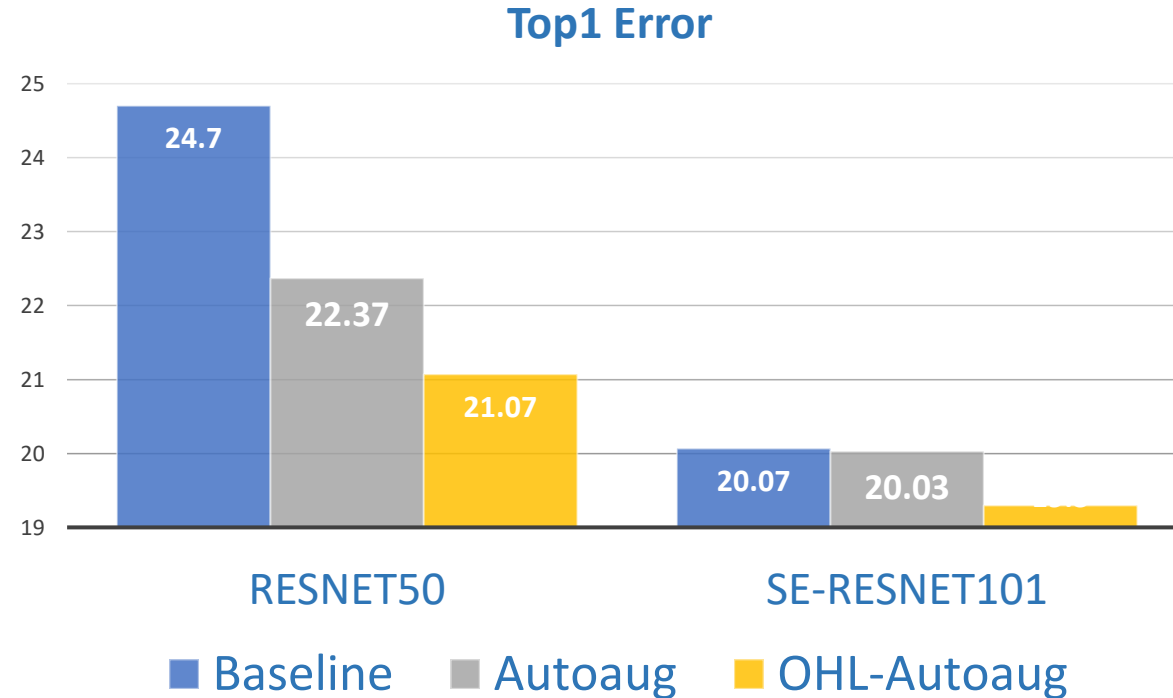


Cubuk, Ekin D., et al. "Autoaugment: Learning augmentation policies from data." *arXiv preprint arXiv:1805.09501* (2018).

Lin, Chen, Minghao Guo, Chuming Li, Wei Wu, Dahua Lin, Wanli Ouyang, and Junjie Yan. "Online Hyper-parameter Learning for Auto-Augmentation Strategy." *ICCV19*.

Experimental Results - ImageNet

- On ImageNet (Top1/Top5 Error)

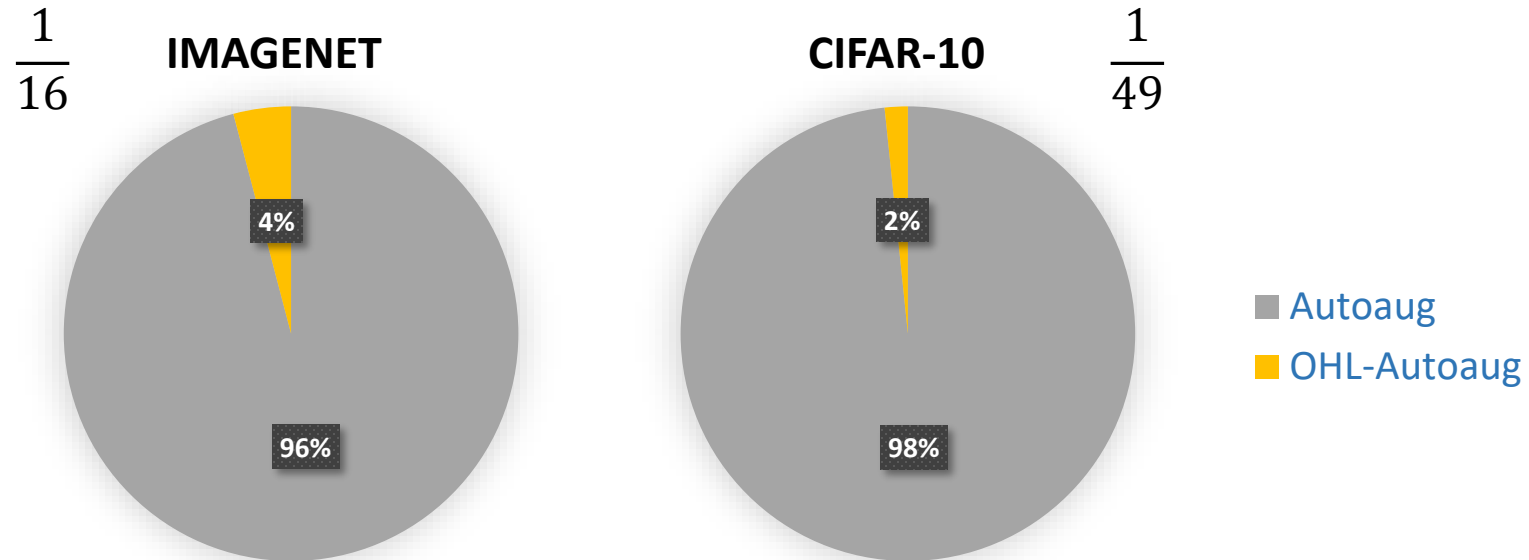


Cubuk, Ekin D., et al. "Autoaugment: Learning augmentation policies from data." *arXiv preprint arXiv:1805.09501* (2018).

Lin, Chen, Minghao Guo, Chuming Li, Wei Wu, Dahua Lin, Wanli Ouyang, and Junjie Yan. "Online Hyper-parameter Learning for Auto-Augmentation Strategy." *ICCV19*.

Computation Required vs Offline Learning

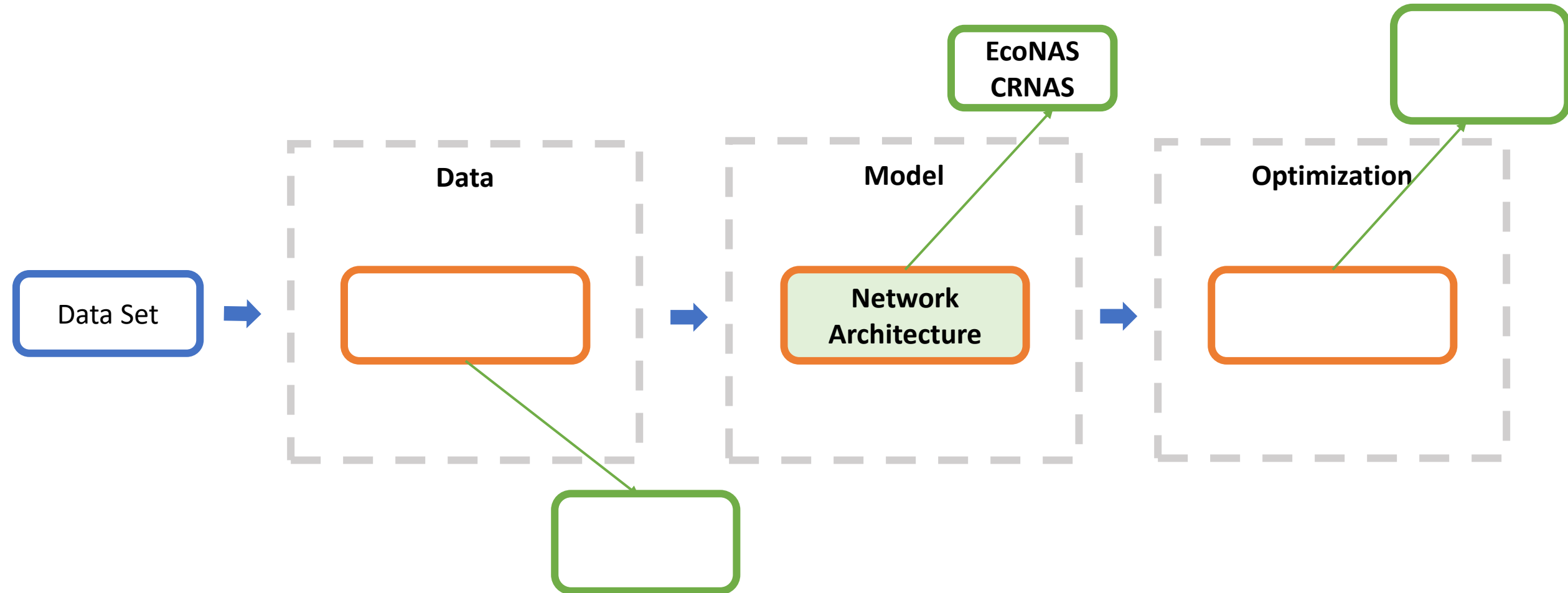
Dataset	Auto-Augment [a]		OHL-Auto-Aug	
	#Iterations	Usage of Dataset (%)	#Iterations	Usage of Dataset(%)
CIFAR-10	4.03×10^6	8%	1.17×10^5	100%
ImageNet	1.76×10^7	0.5%	7.5×10^5	100%
Need to retrain?	Y		N	



[a] Cubuk, Ekin D., et al. "Autoaugment: Learning augmentation policies from data." *arXiv preprint arXiv:1805.09501* (2018).

Lin, Chen, Minghao Guo, Chuming Li, Wei Wu, Dahua Lin, Wanli Ouyang, and Junjie Yan. "Online Hyper-parameter Learning for Auto-Augmentation Strategy." *ICCV19*.

Towards Auto Training System



EcoNAS: Finding Proxies for Economical Neural Architecture Search

Dongzhan Zhou, Xinchu Zhou, Wenwei Zhang, Chen Change Loy,
Shuai Yi, Xuesen Zhang, Wanli Ouyang

EcoNAS: Finding Proxies for Economical Neural Architecture Search

CVPR 2020

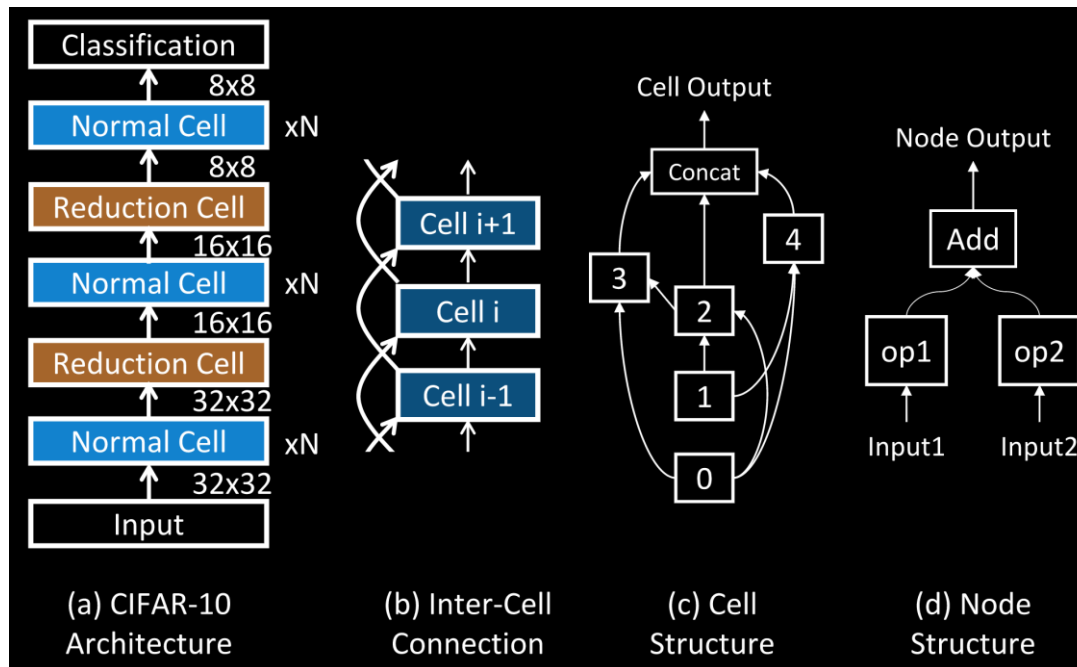
Dongzhan Zhou, Xinchu Zhou, Wenwei Zhang, Chen Change Loy,
Shuai Yi, Xuesen Zhang, Wanli Ouyang

Proxy

- A proxy is a computationally reduced setting, e.g.
 - Reduced input resolution
 - Reduced number of channels
 - Reduced number of samples
 - Reduced number of training epochs
- Compared with the original network, the proxy has the same
 - Operation
 - Number of layers
 - Relative ratio for the numbers of channels between two layers

Exploration study on CIFAR-10

Network Structure (from DARTS [a])



Ops:

3x3 avg pooling	3x3 Separable Conv
3x3 max pooling	5x5 Separable Conv
5x5 max pooling	7x7 Separable Conv
7x7 max pooling	3x3 Dilated Conv
Identity	1x3 then 3x1 Conv
1x1 Conv	1x7 then 7x1 Conv
3x3 Conv	

[a] Liu, H., Simonyan, K., & Yang, Y. (2018). Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*.

Zhou, Dongzhan, Xinchu Zhou, Wenwei Zhang, Chen Change Loy, Shuai Yi, Xuesen Zhang, and Wanli Ouyang. "EcoNAS: Finding Proxies for Economical Neural Architecture Search." *CVPR*, 2020.

Exploration study on CIFAR-10

Specific reduction factors for CIFAR-10

Reduction Factor	0	1	2	3	4
Computation	$\frac{1}{2^0}$	$\frac{1}{2^1}$	$\frac{1}{2^2}$	$\frac{1}{2^3}$	

Training Epochs (e)	120	60	30	
-------------------------	-----	----	----	--

Exploration study on CIFAR-10

Specific reduction factors for CIFAR-10

Reduction Factor	0	1	2	3	4
Channels for CNN (c)	36	24	18	12	8
Resolution of input images (r)	32	24	16	12	8
Sample ratio of full training set (s)	1.0	0.5	0.25	0.125	

Training Epochs (e)	120	60	30	
-------------------------	-----	----	----	--

Motivation

Existing proxies behave differently in maintaining rank consistency.

Example:

	Real Ranking	Ranking in Proxy 1	Ranking in Proxy 2
Network A	1	1	3
Network B	2	2	4
Network C	3	3	1
Network D	4	4	2
		Good Proxy	Bad Proxy

Finding reliable proxies is important for Neural Architecture Search.

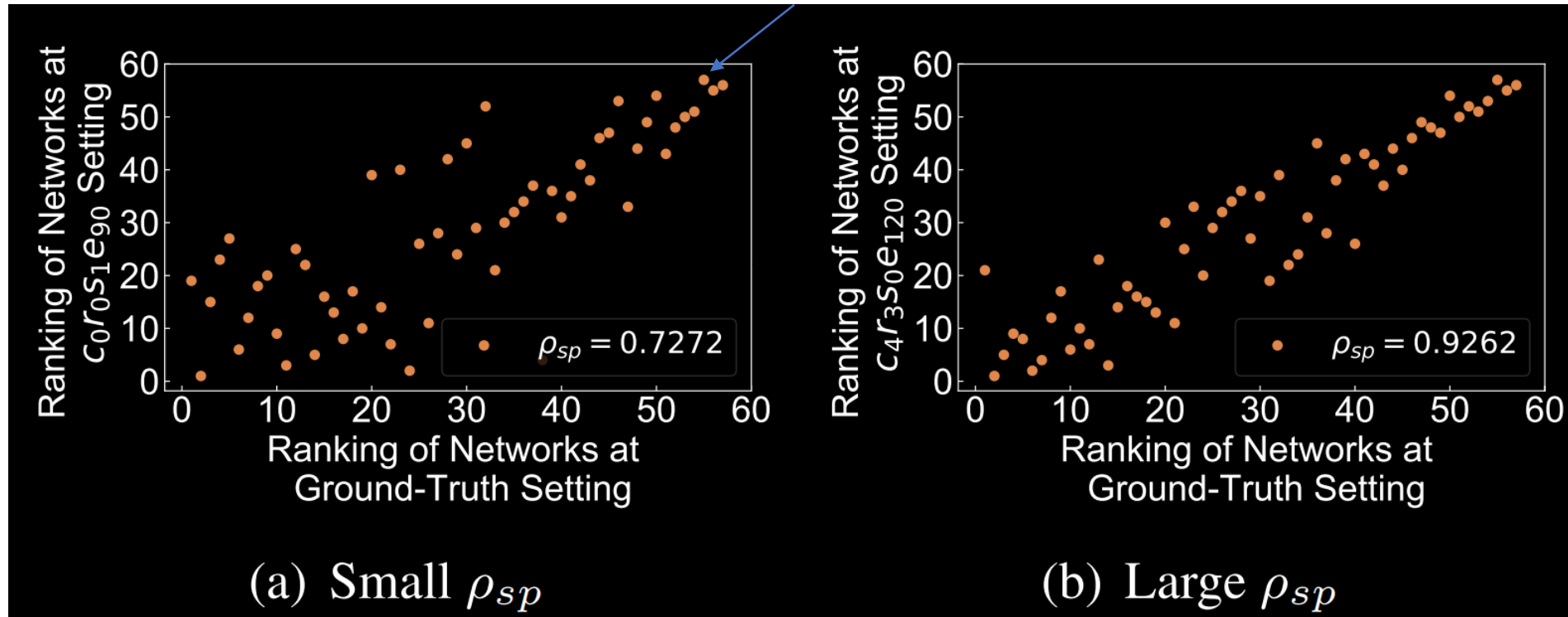
Zhou, Dongzhan, Xinchu Zhou, Wenwei Zhang, Chen Change Loy, Shuai Yi, Xuesen Zhang, and Wanli Ouyang. "EcoNAS: Finding Proxies for Economical Neural Architecture Search." *CVPR*, 2020.

How to evaluate Proxies?

Spearman Coefficient of original ranking (Ground-Truth Setting) and proxy ranking (reduced setting).

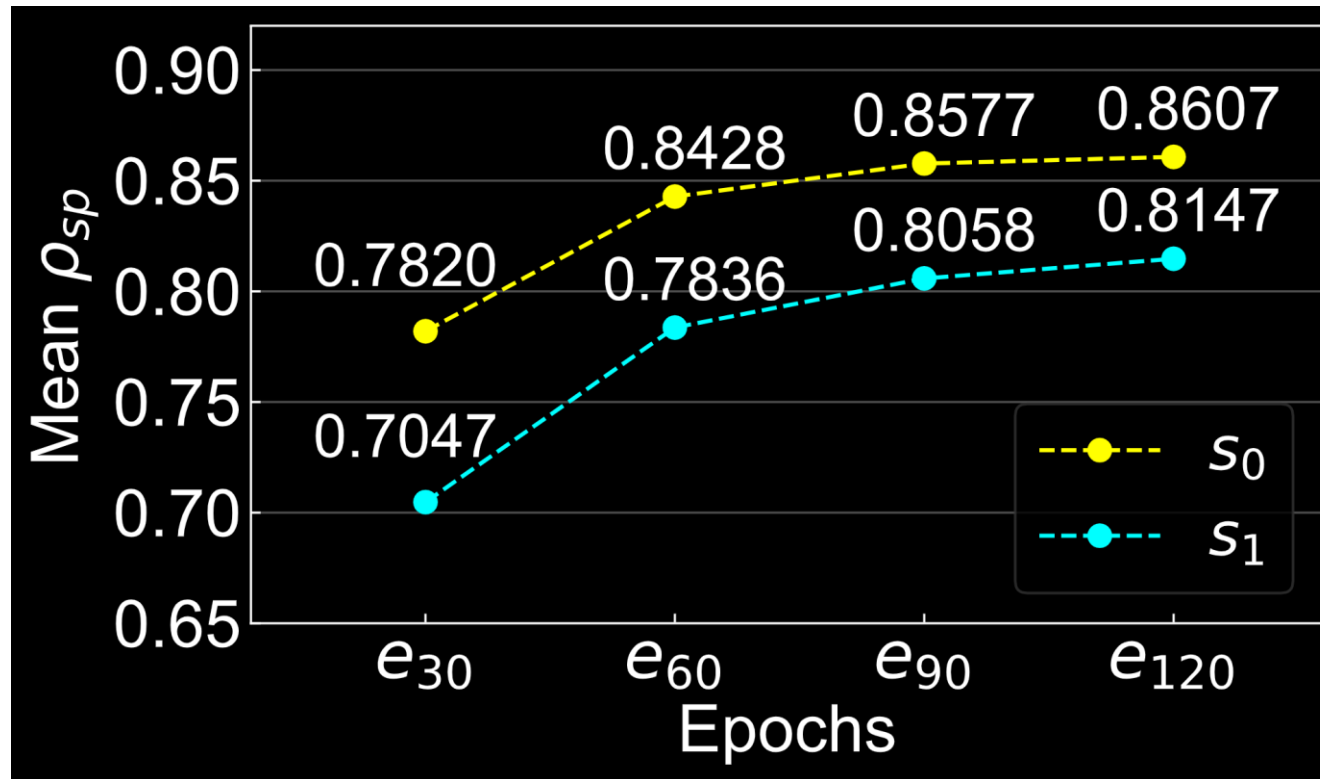
- Value range $[-1, 1]$, higher absolute value indicates stronger correlation.
- Positive value for positive correlation, vice versa.

A model sampled from the search space



Influence of sample ratio (s) and epochs (e)

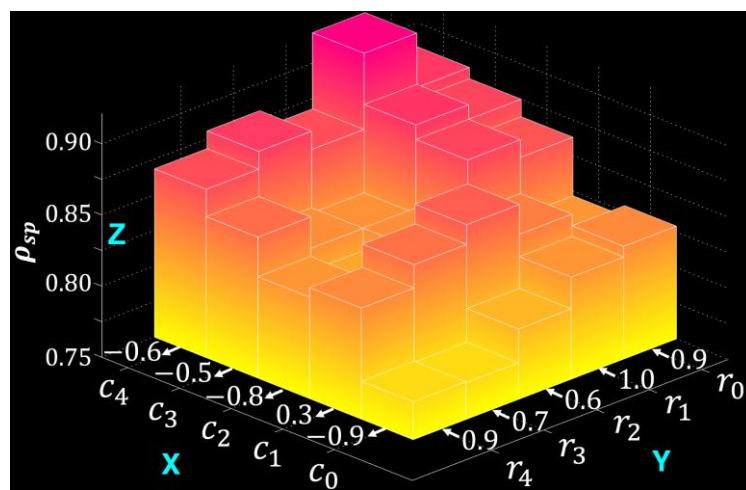
With the same iteration numbers, using more training samples with fewer training epochs could be more effective than using more training epochs and fewer training samples.



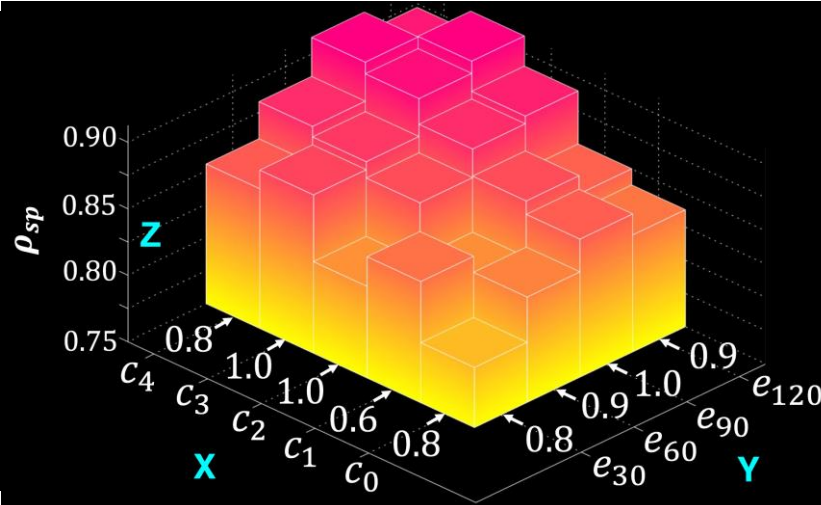
Influence of channels (c) and resolution (r)

Reducing the resolution of input images is sometimes feasible

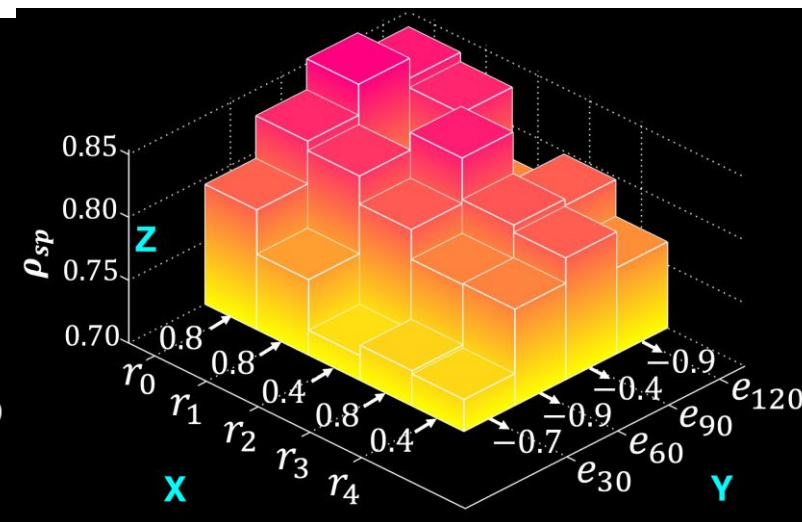
Reducing the number of channels of networks is more reliable than reducing the resolution.



$c_x r_y s_0 e_{60}$



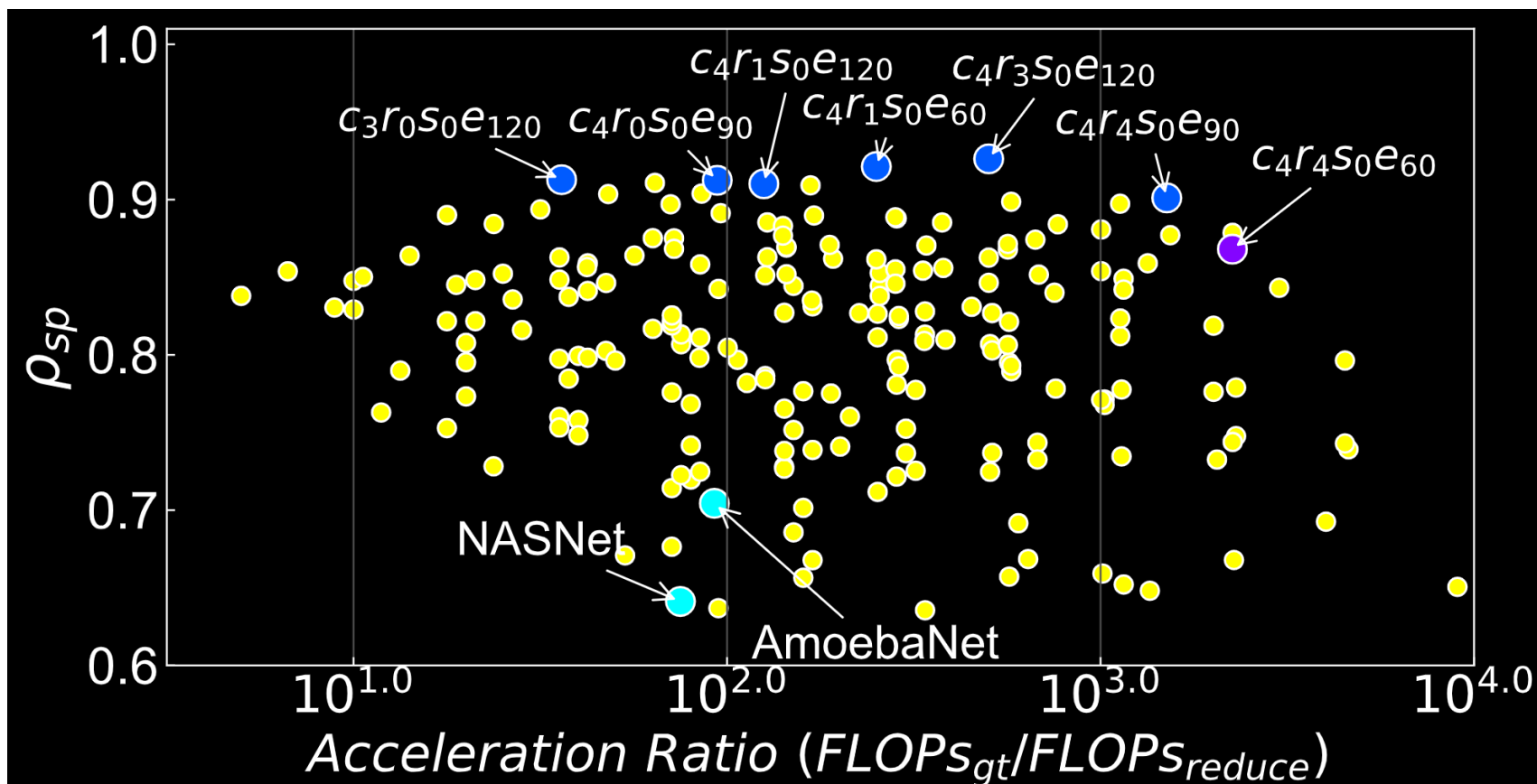
$c_x r_0 s_0 e_y$



$c_0 r_x s_0 e_y$

Efficient proxies

An efficient proxy does not necessarily have a poor rank consistency.




EcoNAS: Economical evolutionary-based NAS

1. Select a more efficient and consistent reduced setting as proxy.
2. Hierarchical proxy strategy: train networks with different proxies based on their accuracy.

Setting: Three population sets P_E , P_{2E} , P_{3E} , which store networks trained for E , $2E$, $3E$ epochs, respectively.

For each cycle:



Step 1. A batch of networks are randomly sampled from P_E , P_{2E} , P_{3E} and mutated. Networks with higher accuracy are more likely to be chosen. Train the mutated networks for E epochs and add them to P_E .

Step 2. Choose top networks from P_E , P_{2E} , load from checkpoints and train E more epochs, then add to P_{2E} , P_{3E} .

Step 3. Remove dead networks from all populations.

EcoNAS ablation study on CIFAR-10

1. Reliable proxy and hierarchical proxy strategy will reduce both searching cost and error rate.

Reduced Setting (w/o hierarchical proxy)	Cost (GPU days)	Spearman Coefficient	Params. (M)	Error Rate (%)
AmoebaNet	3150	0.70	3.20	3.34 ± 0.06
$C_4r_4s_0e_{35}$ (ours)	12	0.74	3.18	2.94

Reduced Setting (w. hierarchical proxy)	Cost (GPU days)	Spearman Coefficient	Params. (M)	Error Rate (%)
NASNet Proxy	21	0.65	2.89	3.20
$C_3r_2s_1e_{60}$	12	0.79	2.56	2.85
$C_4r_4s_0e_{60}$ (ours)	8	0.85	3.40	2.60

EcoNAS ablation study

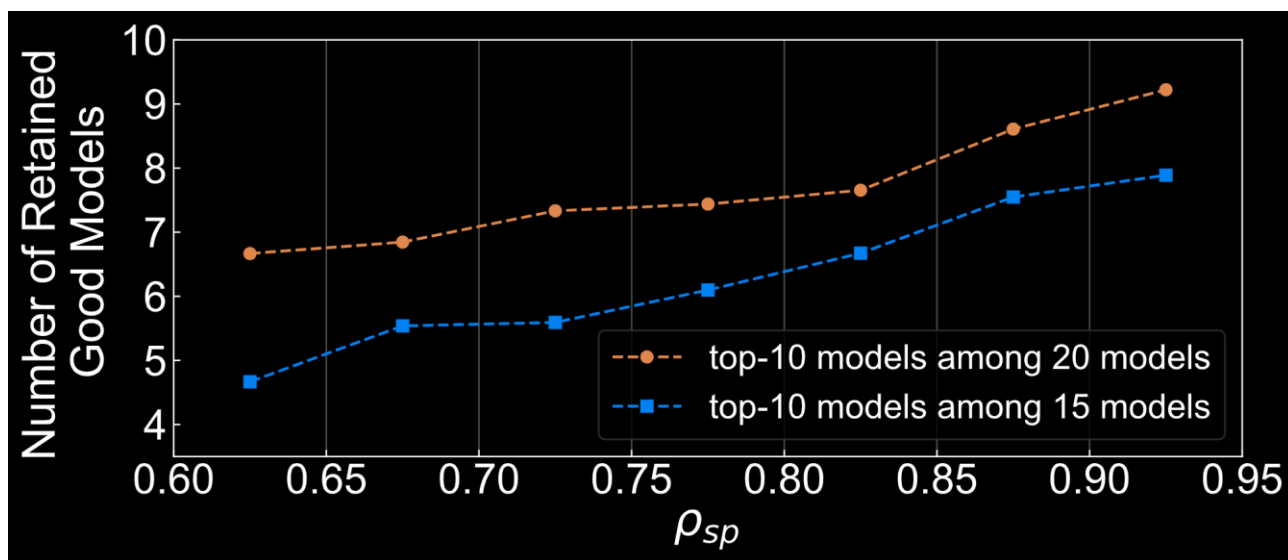
2. Reliable proxy settings can be adopted in other NAS methods.

Method	Setup	Cost (GPU days)	Params. (M)	Error Rate (%)
DARTS (on CIFAR-10)	$c_2r_0s_0$	1.5	3.2	3.0
	$c_4r_2s_0$ (ours)	0.3	5.0	2.8
ProxylessNAS (on ImageNet)	$c_0r_0s_0$ -S	8	4.1	25.4
	$c_0r_0s_0$ -L	8	6.9	23.3
	$c_2r_2s_0$ (ours)	4	5.3	23.2

EcoNAS analysis

Not only save searching costs.

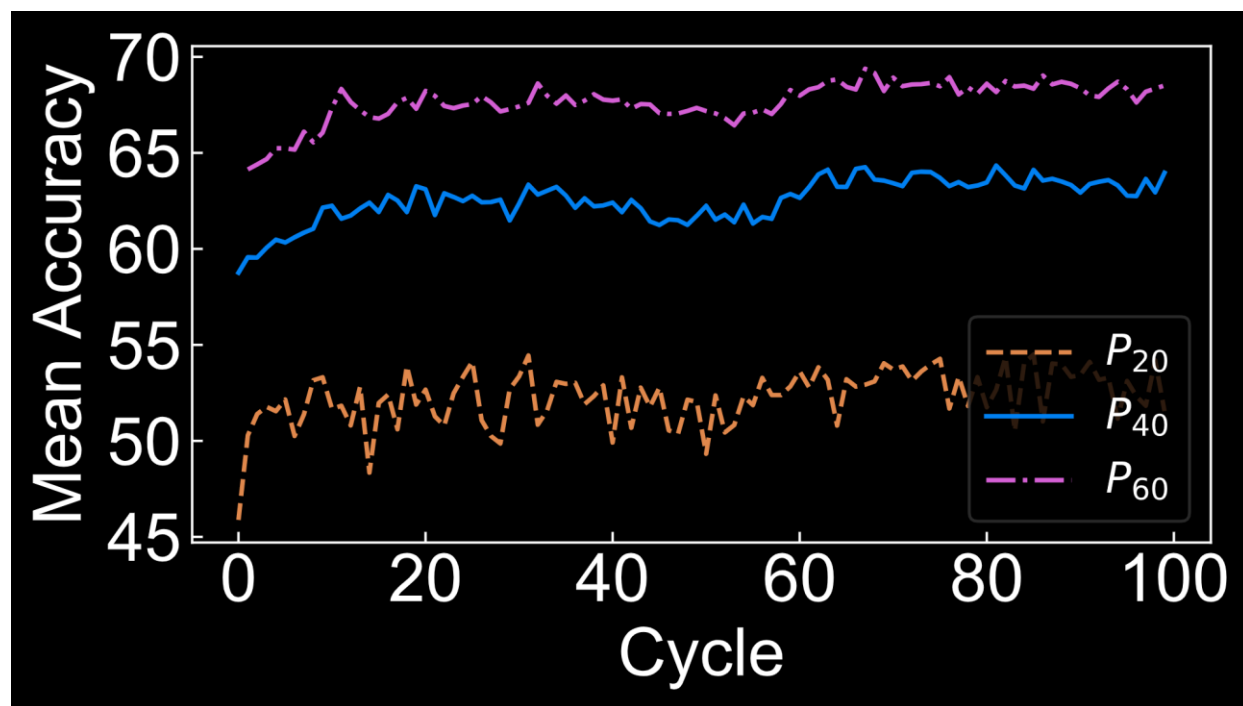
But also save re-training costs. Reliable proxies need not many networks to be re-trained.



Method	Number of Re-training Networks
BlockQNN	100
NASNet	250
AmoebaNet	20
EcoNAS (ours)	5

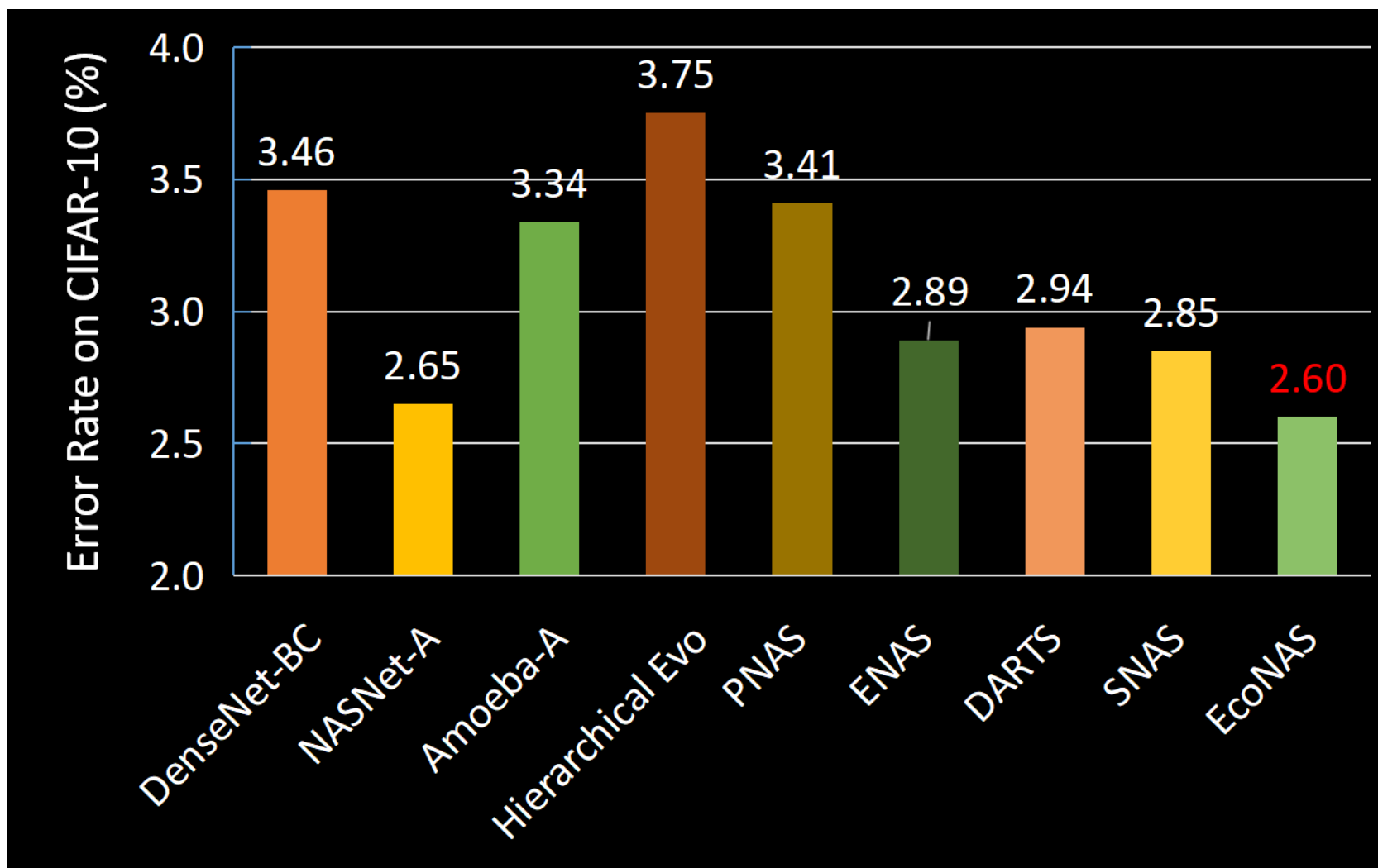
EcoNAS analysis

Provides more diverse structures, which allows searching algorithms to find accurate structures with fewer costs.

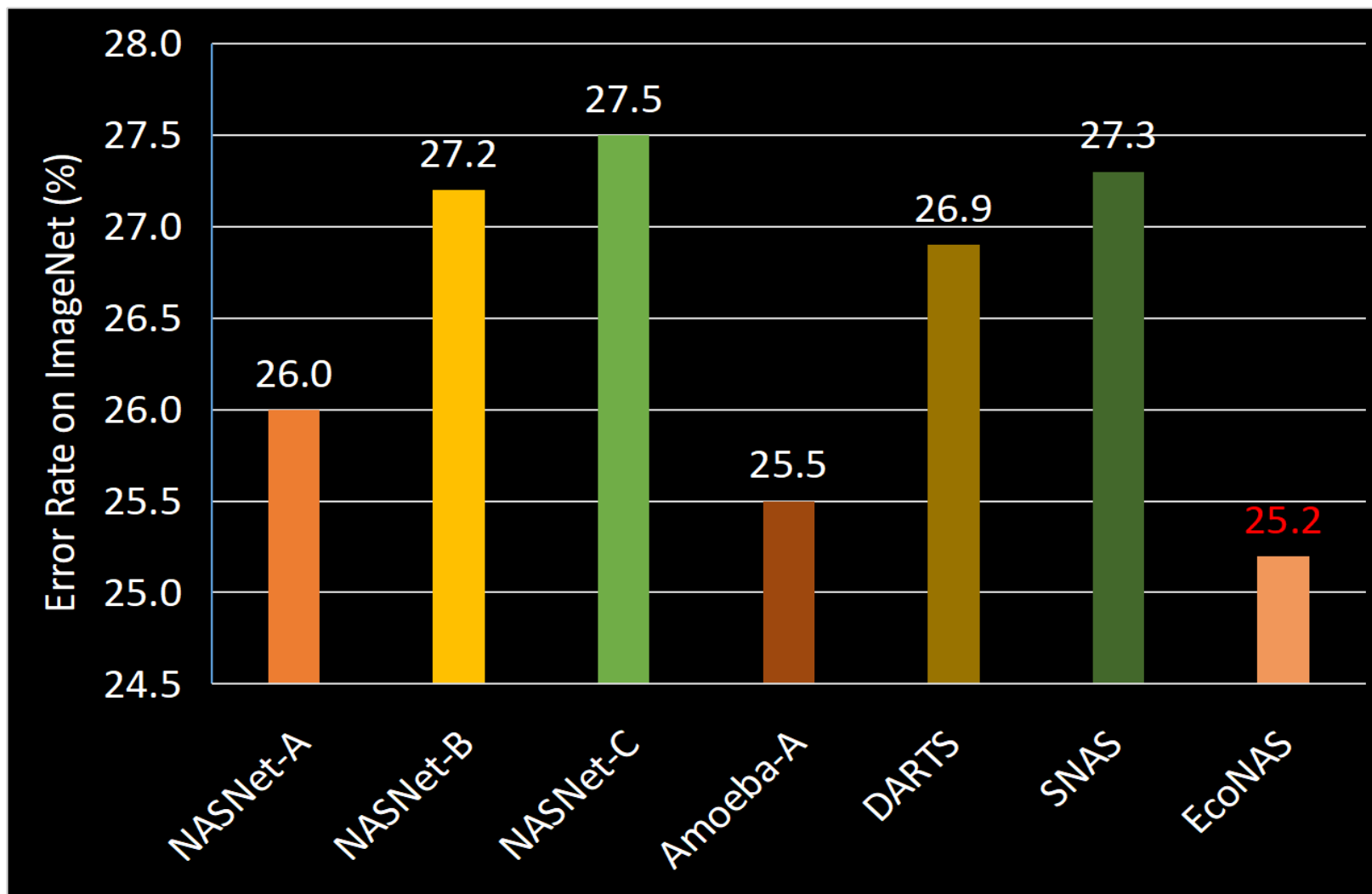


Method	Network numbers
BlockQNN	11k
NASNet	45k
AmoebaNet	20k
EcoNAS (ours)	1k

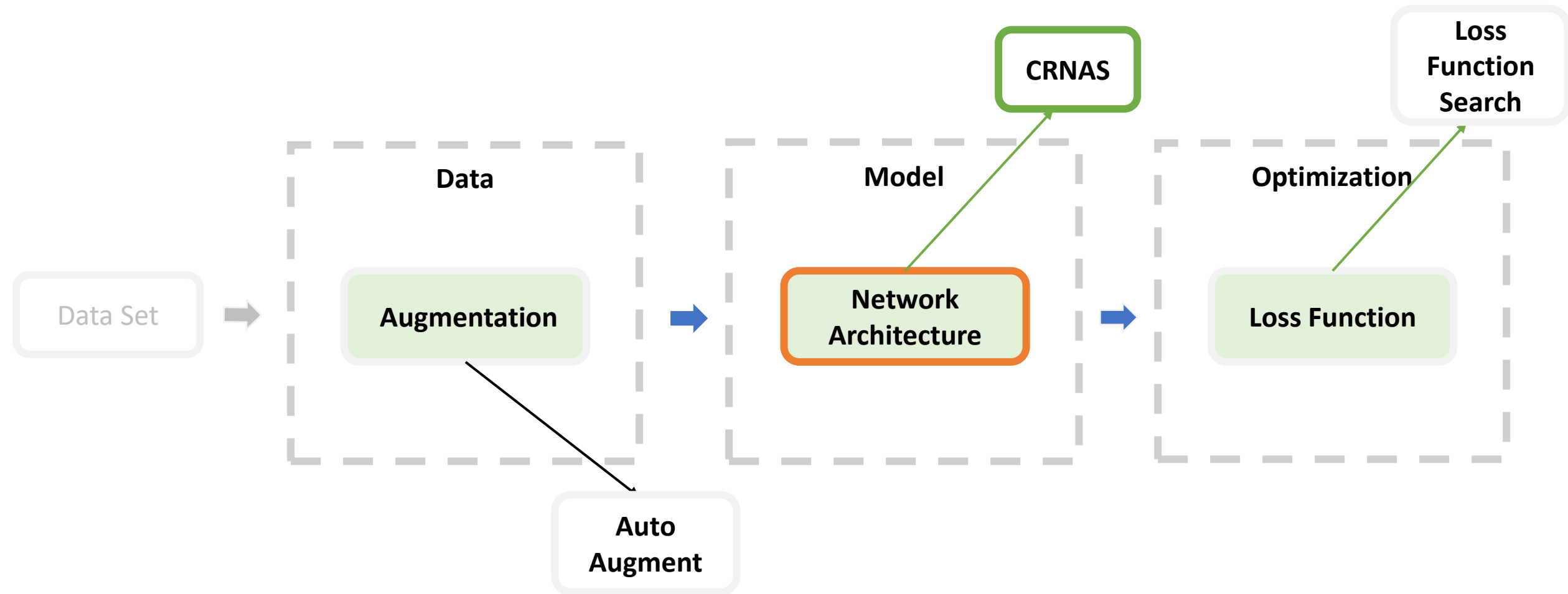
EcoNAS results on CIFAR-10



EcoNAS results on ImageNet



Towards Auto Training System



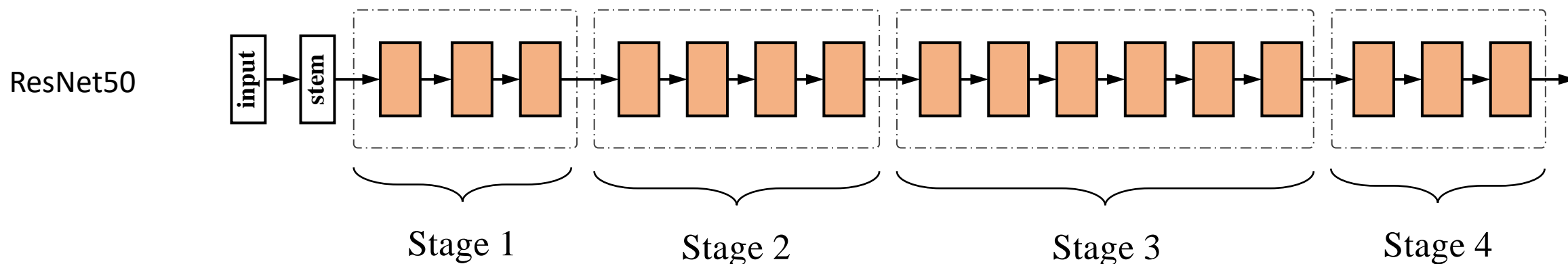
Computation Reallocation for Object Detection

Feng Liang, Chen Lin, Ronghao Guo, Ming Sun, Wei Wu,
Junjie Yan, Wanli Ouyang

ICLR 2020

Motivation

- How to decide the number of layers in each stage?
- The optimal number of layers at a stage is different for different tasks
- Backbone network directly from classification
 - Shown to be sub-optimal, e.g. in DetNet for object detection.



Motivation

- How to decide the number of layers in each stage?
- The optimal number of layers at a stage is different for different tasks
- Backbone network directly from classification
 - Shown to be sub-optimal, e.g. in DetNet for object detection.
- Manual search?
 - Too many choices
 - High computational cost
- Solution: Computation Reallocation (CRNAS)

Computation Reallocation(CRNAS)

We aim to reallocate the engaged computation cost in a more efficient way directly on the detection task. A two-level reallocation space is conducted to reallocate the computation across:

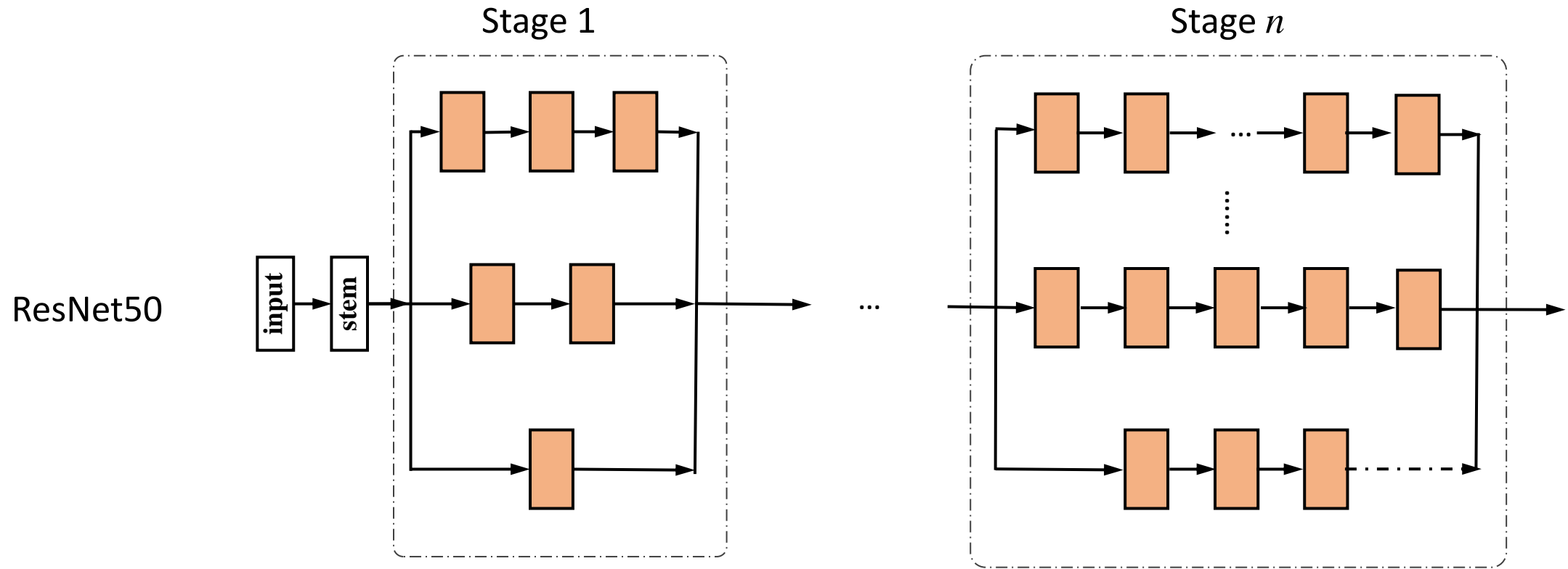
1. different resolution
 - Reallocate the numbers of layers in different stages
 - One-shot NAS + exhausted search
2. spatial position
 - Introducing dilated conv search space
 - One-shot NAS + greedy search

Related work

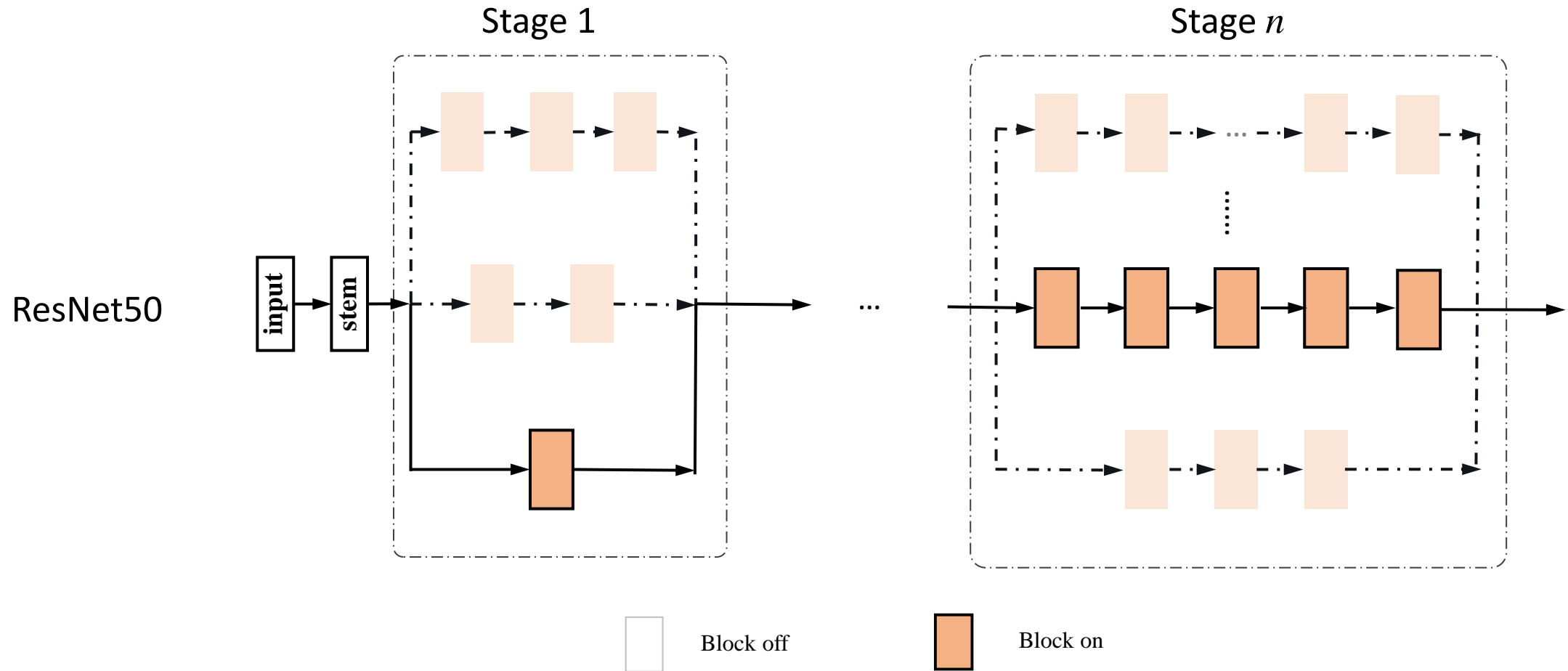
- DetNAS [a] attempted to modify the backbone network automatically by NAS, but it inherits the search space directly from classification.

[a] Chen, Yukang, Tong Yang, Xiangyu Zhang, Gaofeng Meng, Chunhong Pan, and Jian Sun.
"Detnas: Neural architecture search on object detection." *arXiv preprint arXiv:1903.10979* (2019).

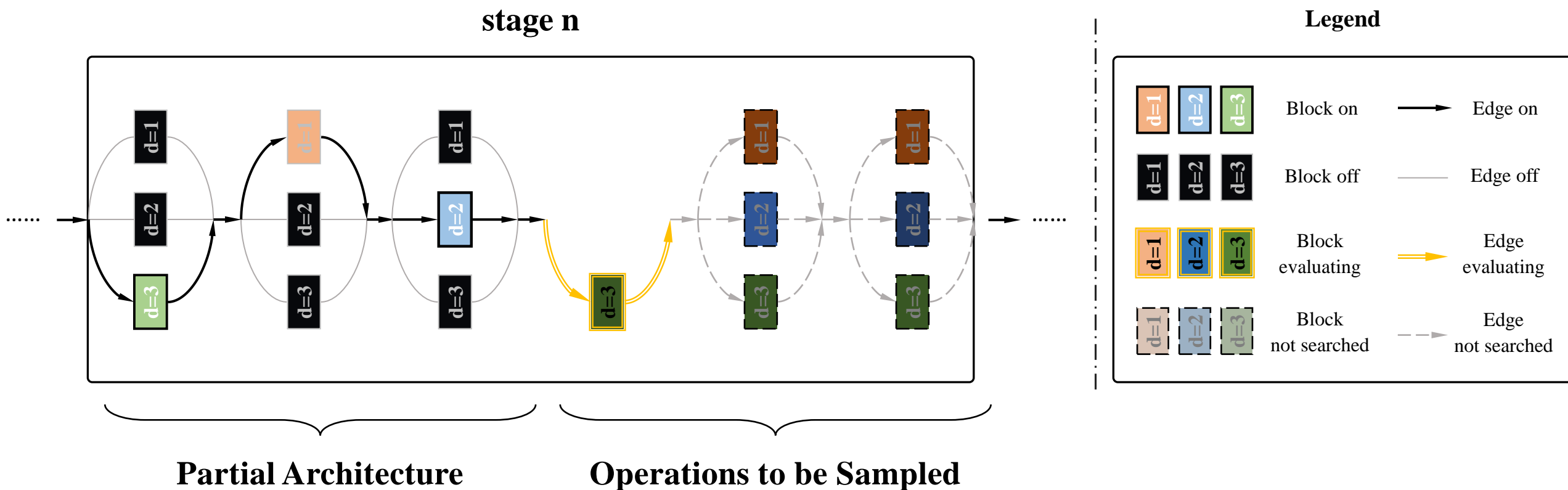
Stage reallocation in different resolution



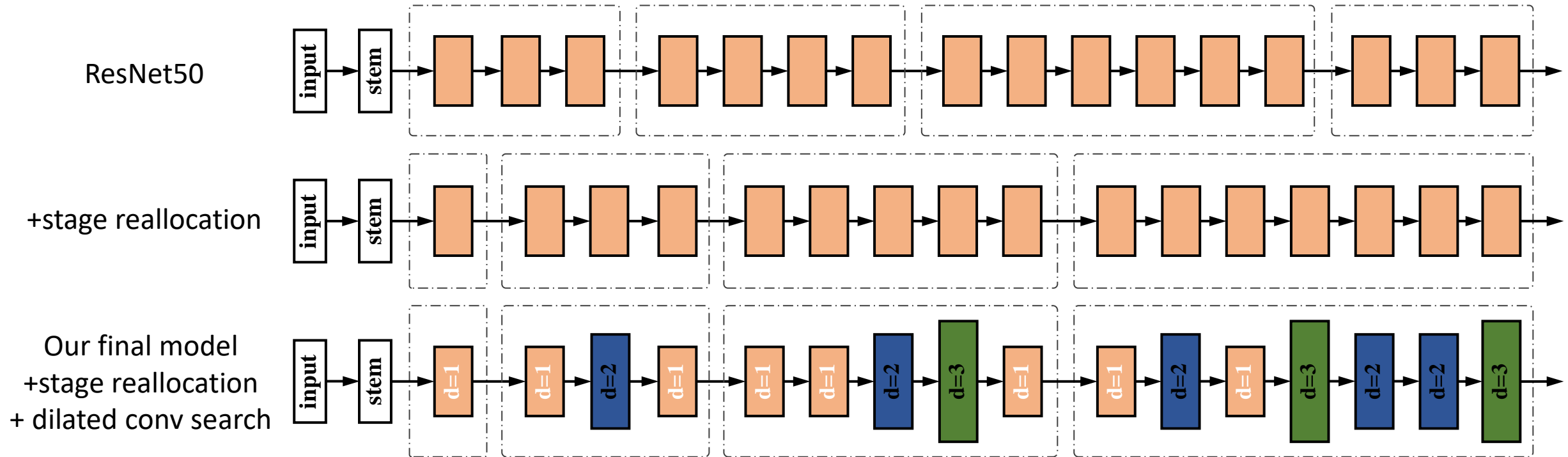
Stage reallocation in different resolution



Dilation reallocation in spatial position



Interpretation of final NAS model



Two trends:

- More blocks in deeper stage: more resources in deeper stage is preferred.
- More dilated conv in deeper stage: Further explore the network potential to detect large objects.

Analysis of ERF in the FPN

ImageNet classification:

- Input size: $224 * 224$
- ERF can be easily satisfied

COCO detection:

- Input size: $800 * 1333$
- sizes of objects vary from 32 to 800
- ERF need more capacities to handle scale variance across the instance

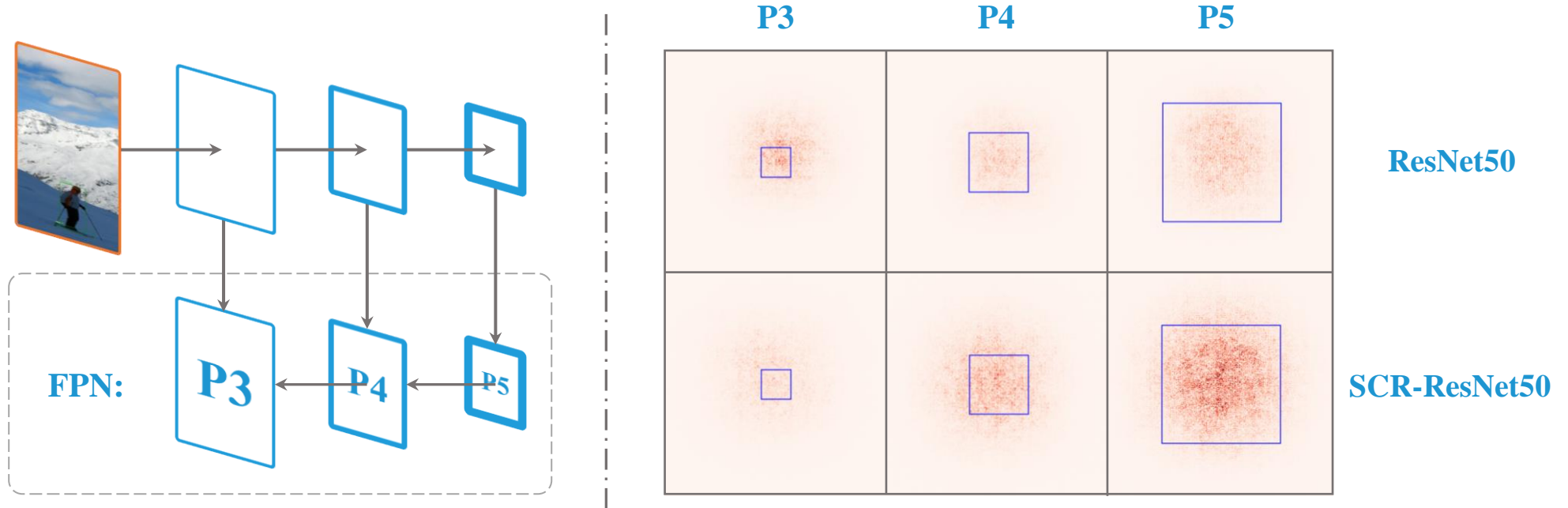
Effective receptive field (ERF) [a]: small Gaussian-like factor of theoretical receptive field (TRF), but it dominates the output.

Our SCR-ResNet50 has more balanced effective receptive field (ERF) across all resolutions, which leads to higher performance

[a]] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In ICCV, pp. 764–773, 2017.

Feng Liang, Chen Lin, Ronghao Guo, Ming Sun, Wei Wu, Junjie Yan, Wanli Ouyang. "Computation Reallocation for Object Detection." *ICLR 2020*.

Visualization of ERF in the FPN



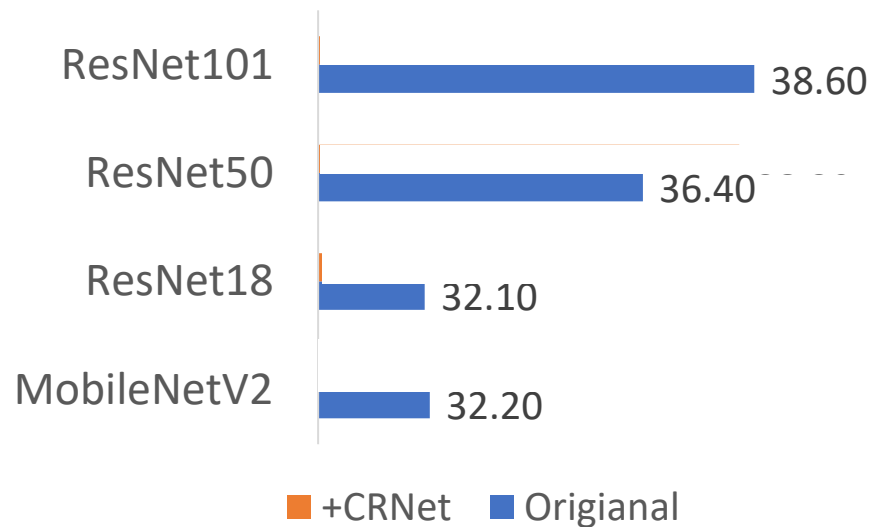
Effective receptive field (ERF) [a]: small Gaussian-like factor of theoretical receptive field (TRF), but it dominates the output.

Our SCR-ResNet50 has more balanced effective receptive field (ERF) across all resolutions, which leads to higher performance

[a]] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In ICCV, pp. 764–773, 2017.

Feng Liang, Chen Lin, Ronghao Guo, Ming Sun, Wei Wu, Junjie Yan, Wanli Ouyang. "Computation Reallocation for Object Detection." *ICLR 2020*.

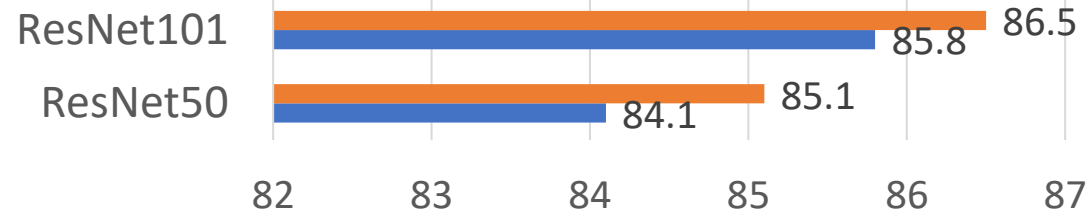
COCO: Consistent gain without additional computation budget.



Backbone	FLOPs(G)
CR-ResNet101	257.5
ResNet101	257.3
CR-ResNet50	192.7
ResNet50	192.5
CR-ResNet18	147.6
ResNet18	147.7
CR-MobileNetV2	121.4
MobileNetV2	121.1

Transfer-ability over other dataset(VOC), other powerful neck/head(NAS-FPN) and other vision task(instance segmentation).

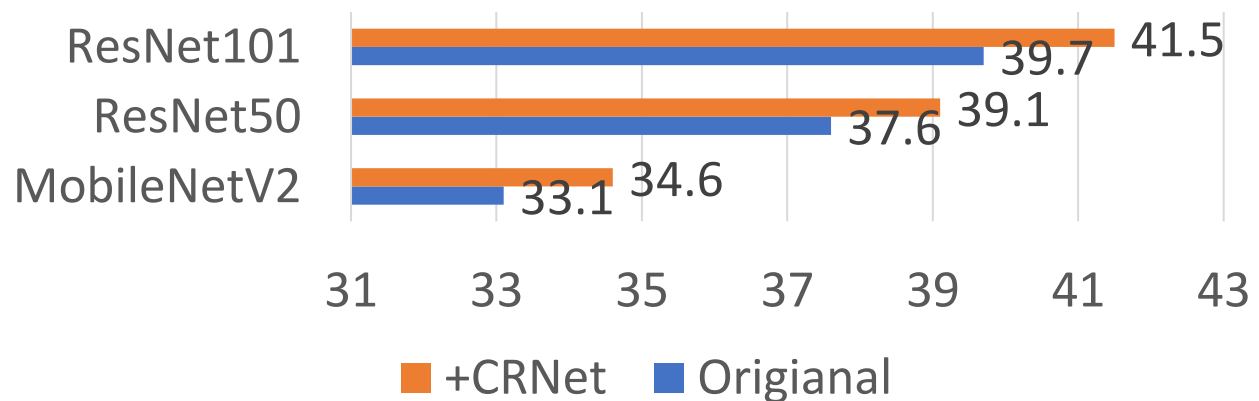
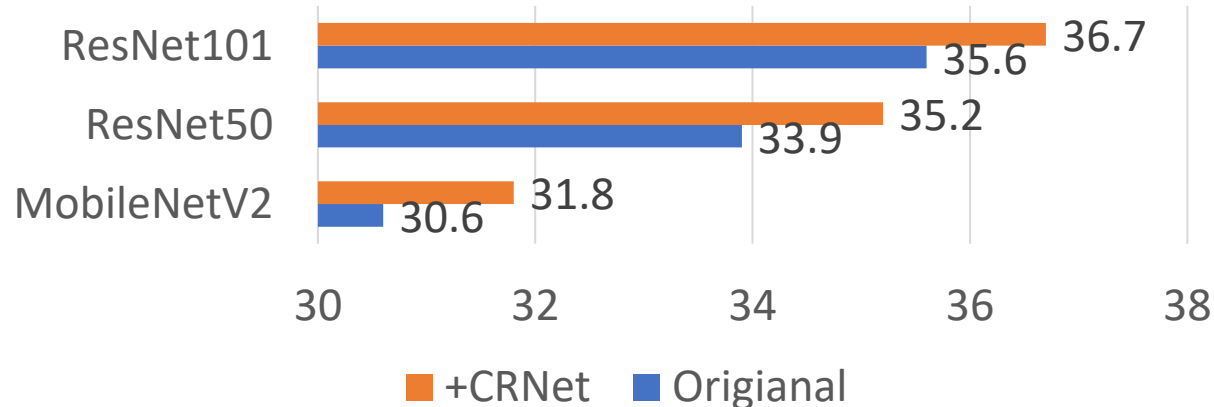
VOC test2007 (Faster RCNN + FPN)



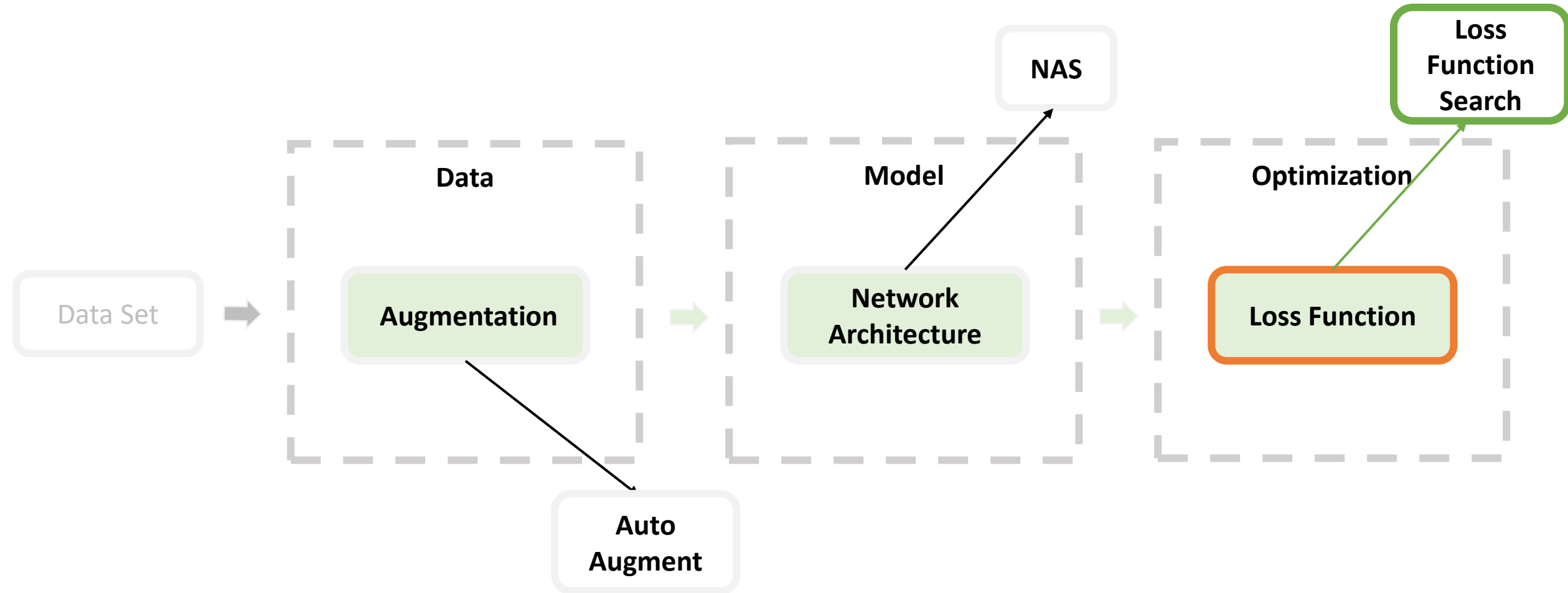
COCO Minival (Mask RCNN)

Segmentation

Detection



Towards Auto Training System



AM-LFS: AutoML for Loss Function Search

Li, Chuming, Chen Lin, Minghao Guo, Wei Wu, Wanli Ouyang, and Junjie
Yan

ICCV 2019

Motivation

- Loss function plays an important role in visual analysis.
- Problem of existing loss function designs
 - heavily rely on domain experts to explore the large design space
 - usually time-consuming
 - cannot be adapted to the data.
- Using different loss functions in different training stages had been observed effective under certain conditions, e.g. curriculum learning.

Can machine learn loss
function?

Yes!

Can machine learn free-form
loss function?

Hard!

Existing widely used loss functions

- Loss in identification task
 - Uniform expression:

$$L_i = -\log \left(\frac{e^{\|w_{y_i}\| \|x_i\| t(\cos(\theta_{y_i}))}}{e^{\|w_{y_i}\| \|x_i\| t(\cos(\theta_{y_i}))} + \sum_{j \neq y_i} e^{\|w_j\| \|x_i\| \cos(\theta_j)}} \right)$$

- Loss in classification task
 - Uniform expression:

$$L_i = -\log \left(\tau \left(\frac{e^{\|w_{y_i}\| \|x_i\| \cos(\theta_{y_i})}}{e^{\|w_{y_i}\| \|x_i\| \cos(\theta_{y_i})} + \sum_{j \neq y_i} e^{\|w_j\| \|x_i\| \cos(\theta_j)}} \right) \right)$$

Loss Function	$t(x)$
SphereFace	$\cos(m \cdot \arccos(x))$
CosFace	$x - m$
ArcFace	$\cos(\arccos(x) + m)$

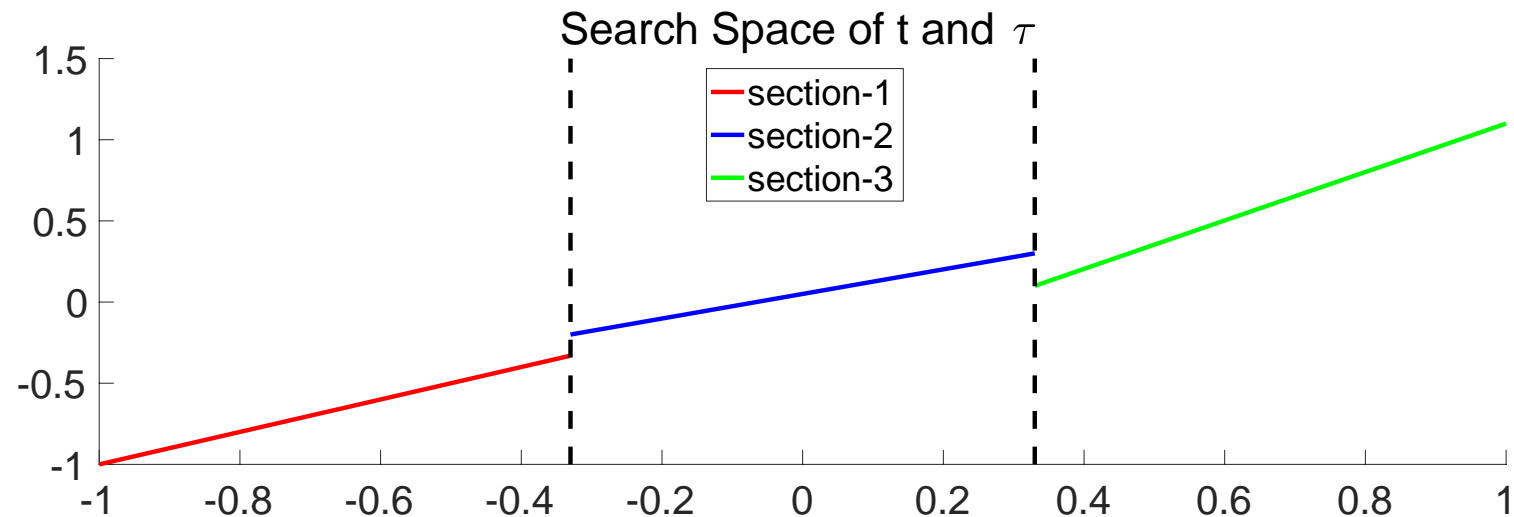
Loss Function	$\tau(x)$
FocalLoss	$x^{(1-x)^m}$

Unified expression of Loss

- A unified expression containing all above losses (Fig. 1)

$$L_i = -\log \left(\tau \left(\frac{e^{\|w_{y_i}\| \|x_i\| t(\cos(\theta_{y_i}))}}{e^{\|w_{y_i}\| \|x_i\| \cos(\theta_{y_i})} + \sum_{j \neq y_i} e^{\|w_j\| \|x_i\| \cos(\theta_j)}} \right) \right)$$

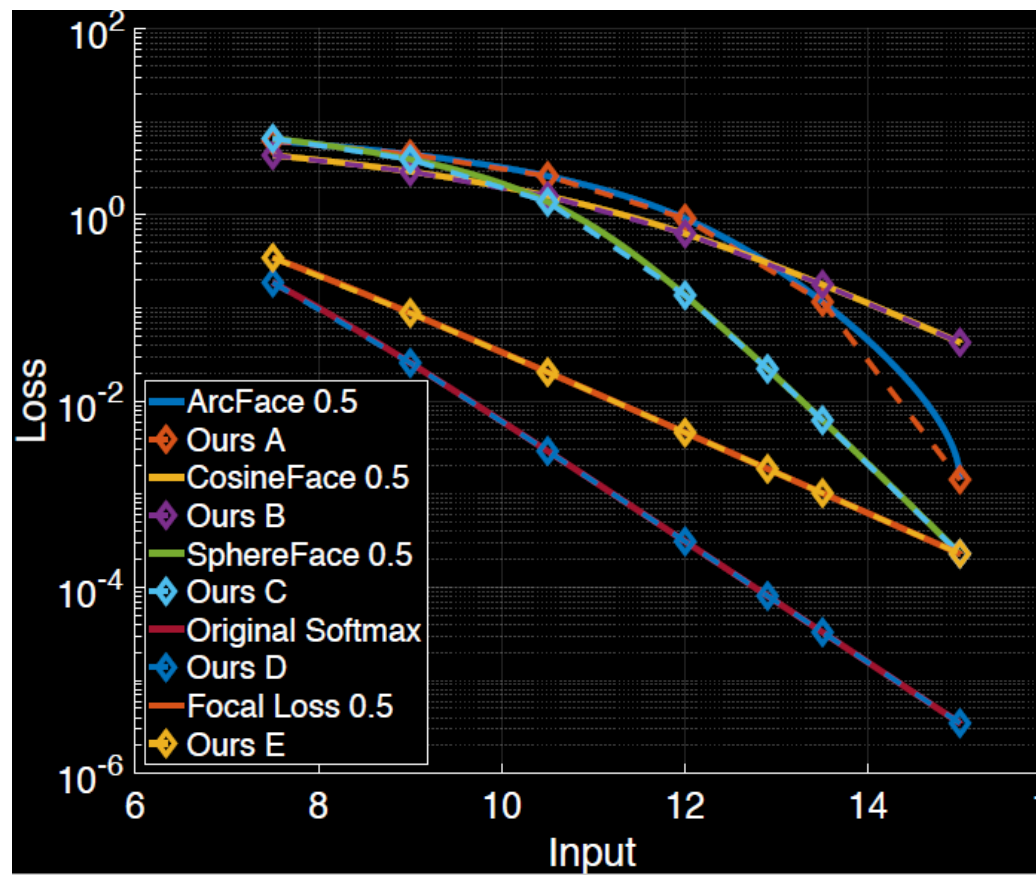
- Model τ and t as piecewise linear function (Fig. 2)



Unified expression of Loss

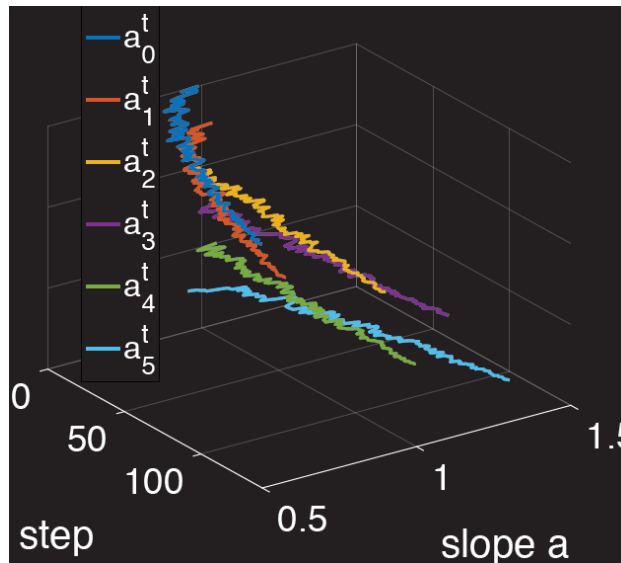
$$L_i = -\log \left(\tau \left(\frac{e^{\|w_{y_i}\| \|x_i\| t (\cos(\theta_{y_i}))}}{e^{\|w_{y_i}\| \|x_i\| \cos(\theta_{y_i})} + \sum_{j \neq y_i} e^{\|w_j\| \|x_i\| \cos(\theta_j)}} \right) \right)$$

- Large portion of hand-crafted loss in different computer vision tasks can be approximated using our unified expression of loss



Unified expression of Loss - continue

- We use independent Gaussian distributions to model τ and t , optimize its mean or even variance.
- The same OHL framework for auto-augmentation works well on optimizing these parameters.



The convergence of these parameters.

Experimental results

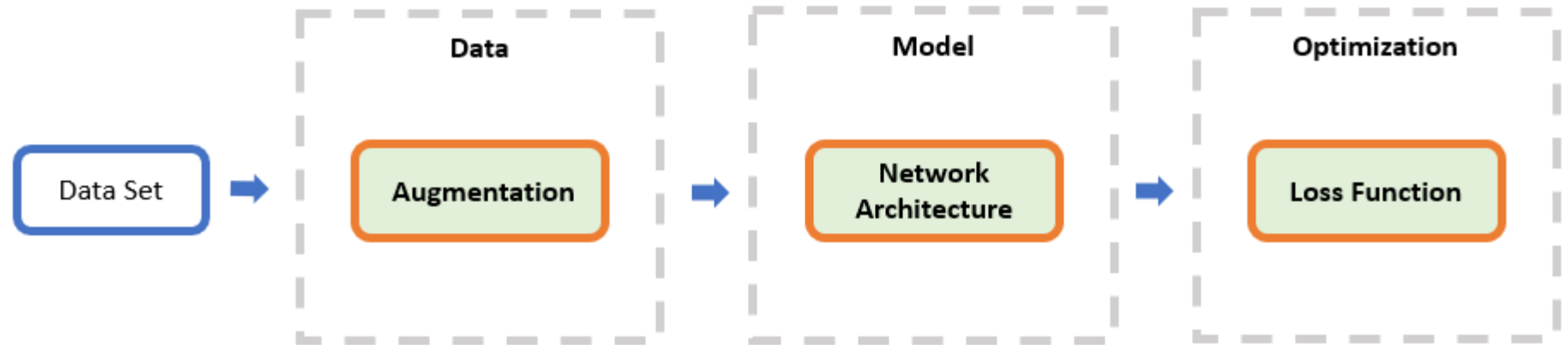
- Results on classification
- Dataset: Cifar10+noise

Noise ratio	Baseline	Ours
0%	91.2	93.1
10%	87.9	89.9
20%	84.9	87.3

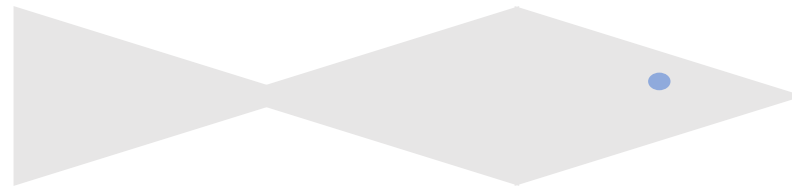
Outline

- Introduction

- Auto-ML



- Manual design



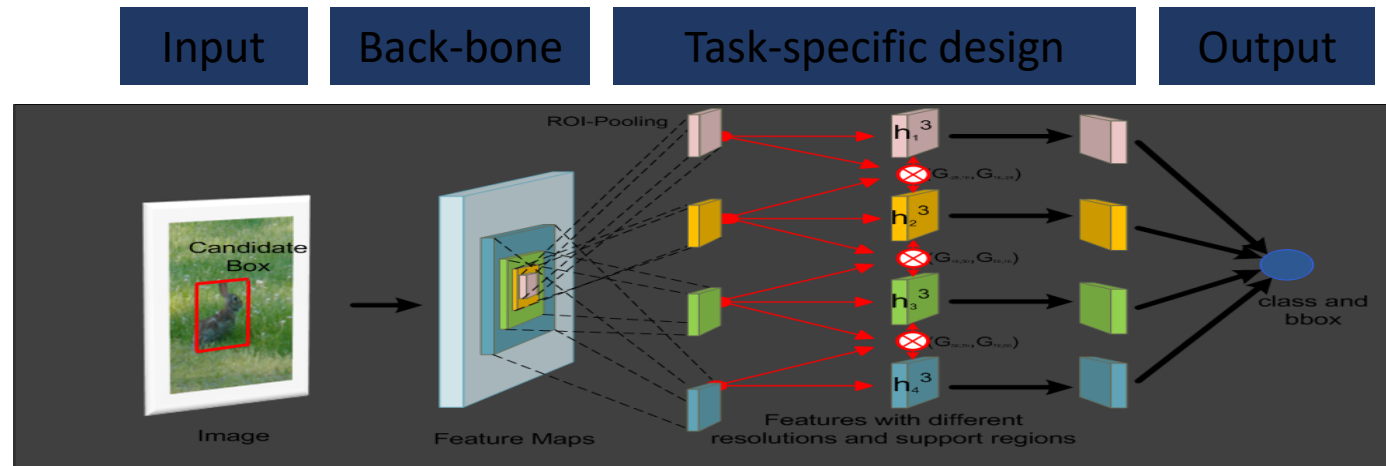
Manual design is not important?

No!

Provides new possible structure and operations as the Search Space for NAS

Back-bone deep model design

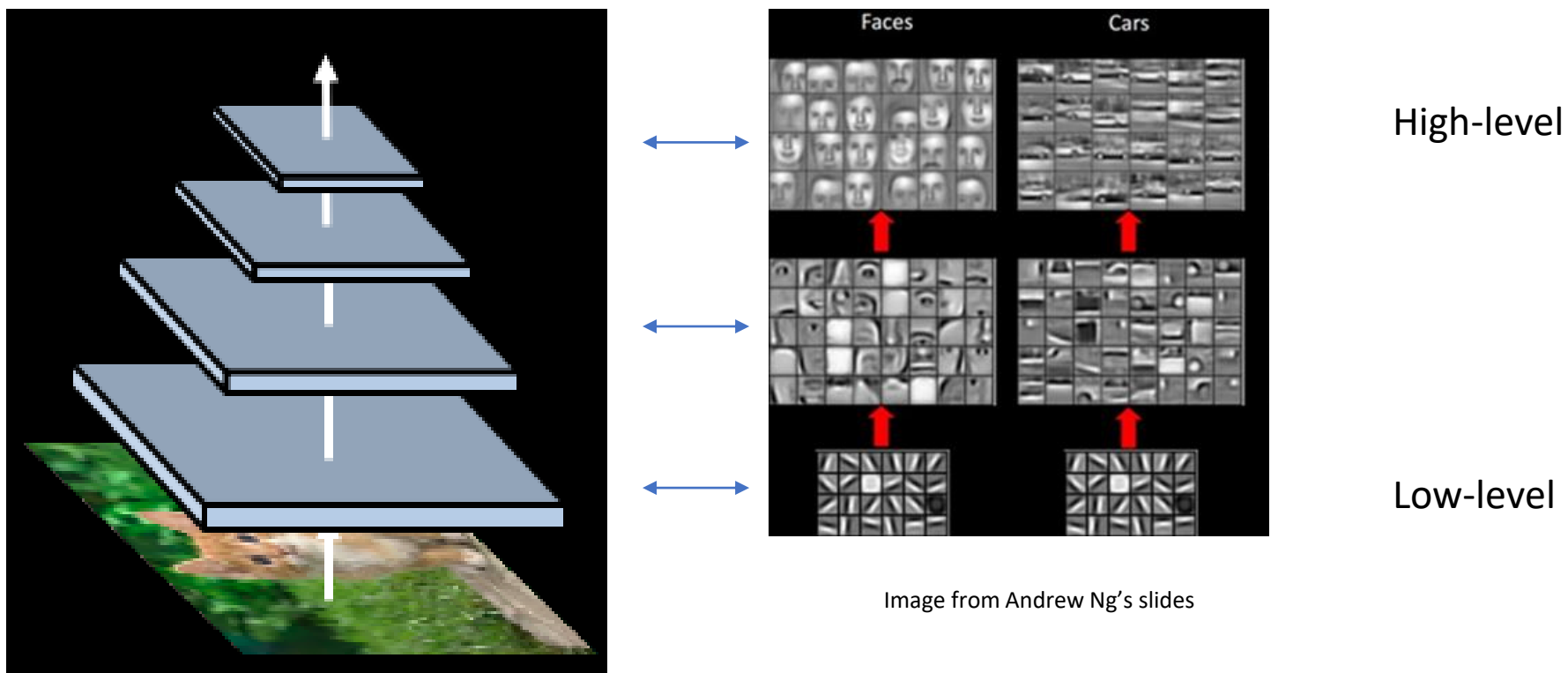
- Basis structure of deep model
 - AlexNet, VGG, GoogleNet, ResNet, DenseNet
 - Validated on large-scale classification tasks such as ImageNet
 - Models pretrained on ImageNet are found to be the effective initial model for other tasks



FishNet: A Versatile Backbone for Image, Region, and Pixel Level Prediction

Shuyang Sun, Jiangmiao Pang, Jianping Shi, Shuai
Yi, **Wanli Ouyang**, *NurIPS*. 2018

Low-level and high-level features



Current CNN Structures

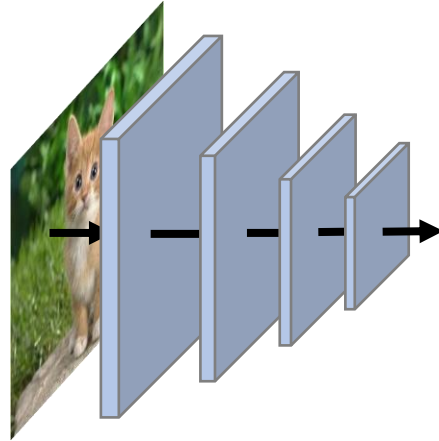
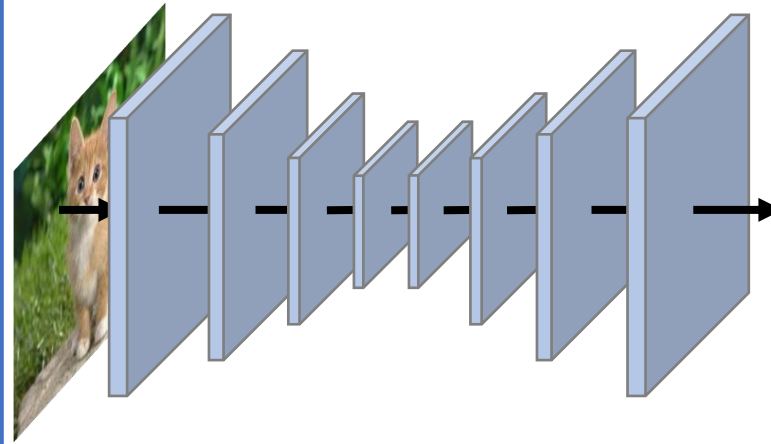


Image Classification:
Summarize high-level semantic
information of the whole image.

Called U-Net, Hourglass, or Conv-deconv



Detection/Segmentation:
High-level semantic meaning with
high spatial resolution

Architectures designed for tasks of different granularities are
DIVERGING

Unify the advantages of networks for pixel-level,
region-level, and image-level tasks

Observation and design

- Our observation

1. Diverged structures for tasks requiring different resolutions.

- Design

1. Unify the advantages of networks for pixel-level, region-level, and image-level tasks.

Hourglass for Classification

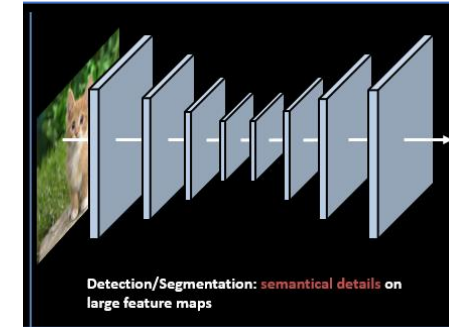
Features with high-level semantics and high resolution is good

Directly applying hourglass for classification?

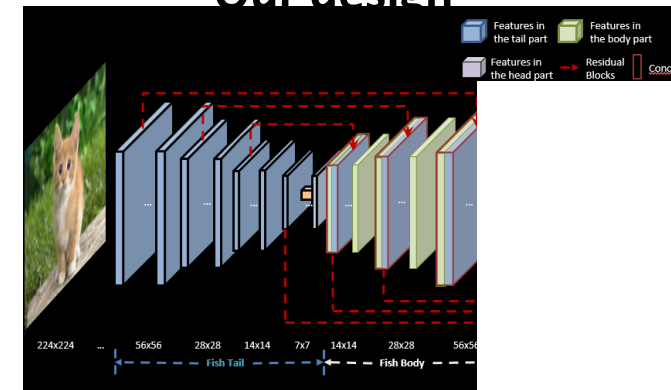
Poor performance.

So what is the **problem**?

- Different tasks require different resolutions of feature
- Down sample high-level features with high resolution



Our design



Observation and design

- Observation

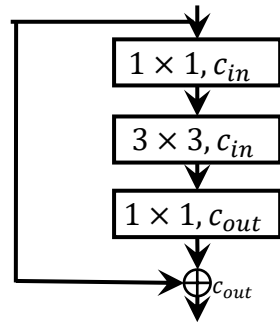
1. Diverged structures for tasks requiring different resolutions.
2. Isolated Conv blocks the direct back-propagation

- Design

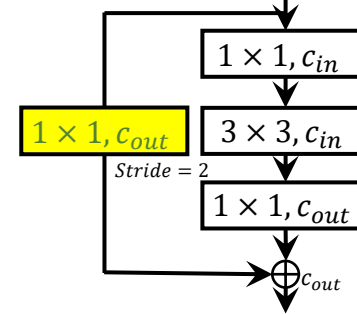
1. Unify the advantages of networks for pixel-level, region-level, and image-level tasks.

Hourglass for Classification

Normal Res-Block



Res-Block for
down/up sampling



The 1×1 convolution layer in yellow indicates the **isolated convolution**.

- Hourglass may bring more **isolated convolutions** than ResNet

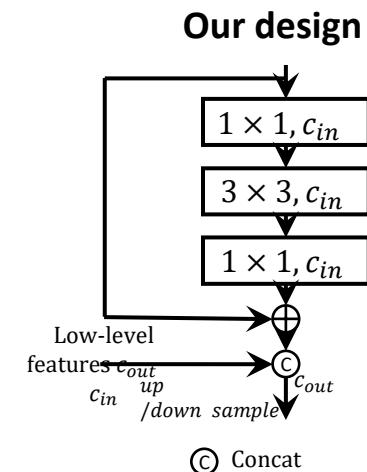
Observation and design

- Observation

1. Diverged structures for tasks requiring different resolutions.
2. Isolated Conv blocks the direct back-propagation

- Design

1. Unify the advantages of networks for pixel-level, region-level, and image-level tasks.
2. Design a network that does not need isolated convolution



Observation and design

- Observation

1. Diverged structures for tasks requiring different resolutions.
2. Isolated Conv blocks the direct back-propagation
3. Features with different depths are not fully explored, or **mixed** but not preserved

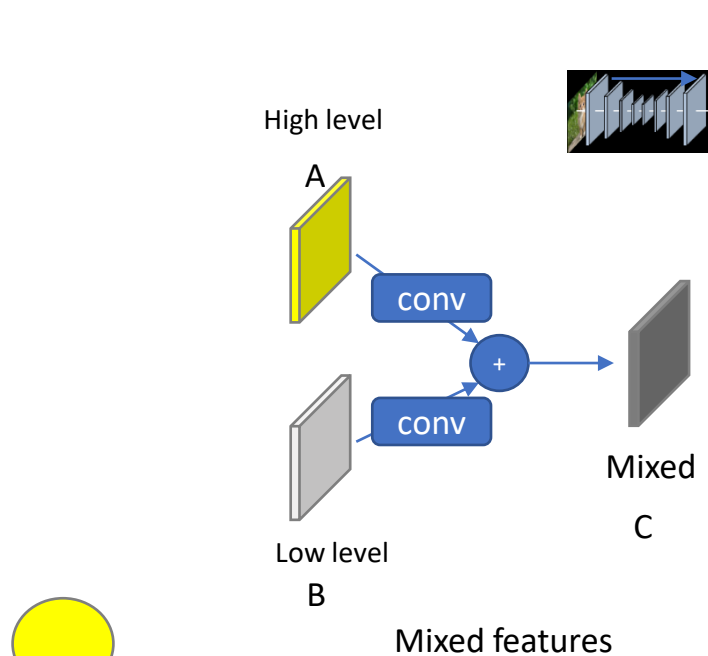
- Design

1. Unify the advantages of networks for pixel-level, region-level, and image-level tasks.
2. Design a network that does not need isolated convolution
3. Features from varying depths are **preserved and refined** from each other.

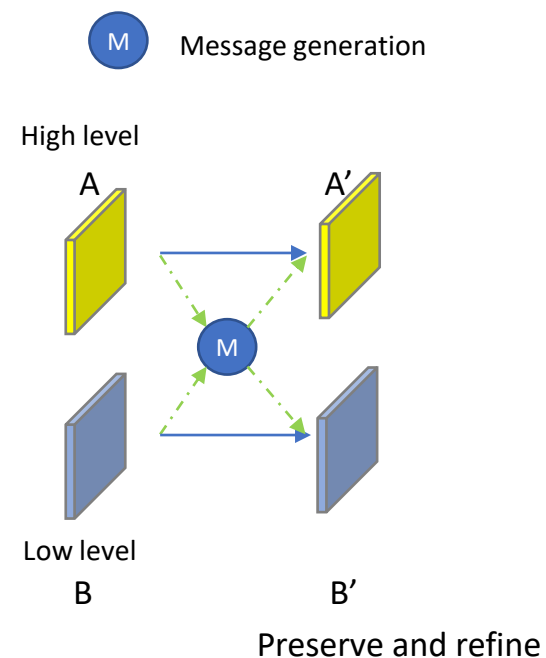
Bharath Hariharan, et al. "Hypercolumns for object segmentation and fine-grained localization." *CVPR*'15.

Newell, Alejandro, Kaiyu Yang, and Jia Deng. "Stacked hourglass networks for human pose estimation." *ECCV*'16.

Difference between **mix** and **preserve and refine**

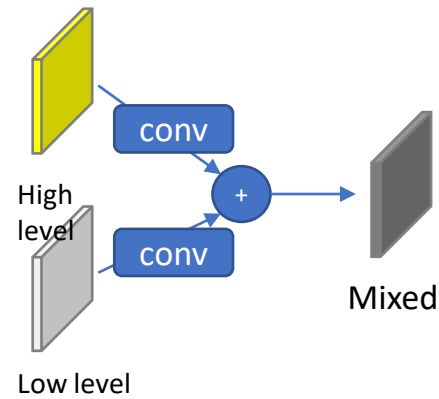


$$C = f_A(A) + f_B(B)$$



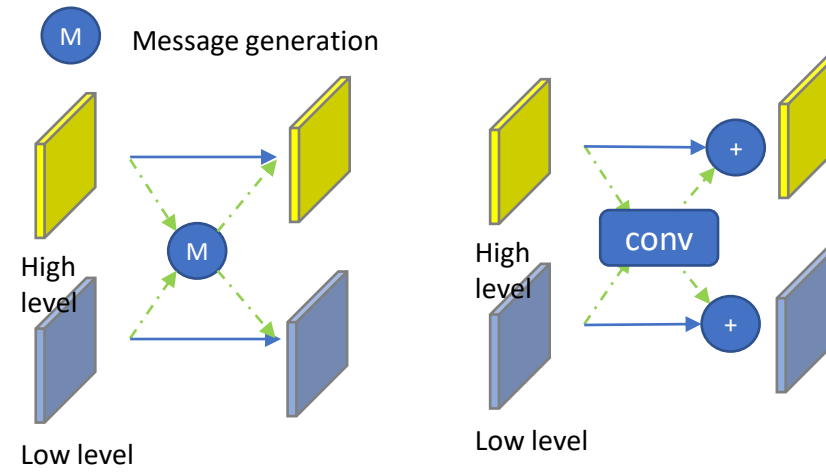
$$A' = A + M_1(A, B)$$
$$B' = B + M_2(A, B)$$

Difference between mix and preserve and refine



Mixed features

$$C = f_A(A) + f_B(B)$$



Preserve and refine

$$A' = A + M_1(A, B)$$
$$B' = B + M_2(A, B)$$

Observation and design

Solution

1. Diverged structures for tasks requiring different resolutions.
2. Isolated Conv blocks the direct back-propagation
3. Features with different depths are not fully explored, or **mixed** but not preserved

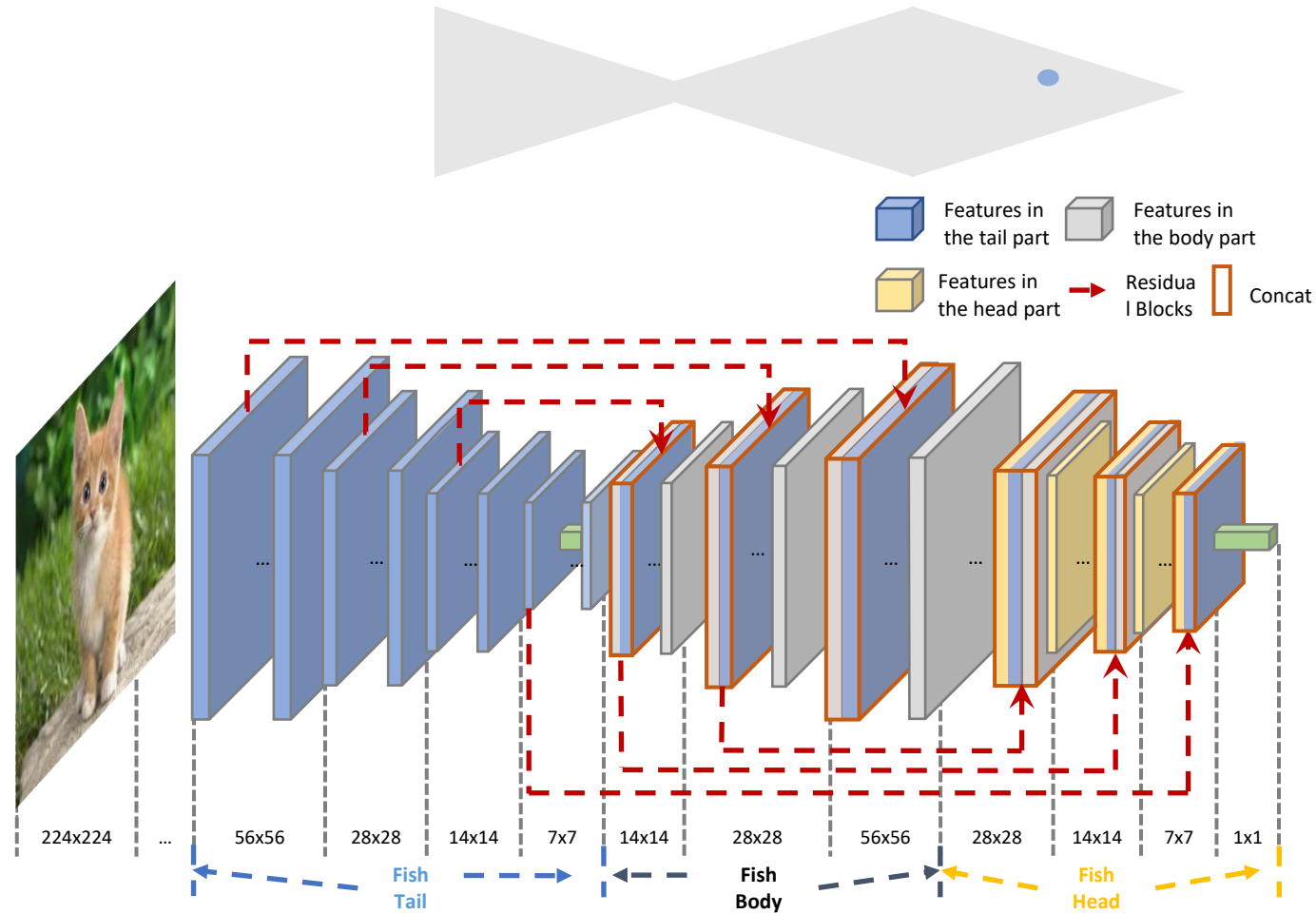
Our observation

1. Unify the advantages of networks for pixel-level, region-level, and image-level tasks.
2. Design a network that does not need isolated convolution
3. Features from varying depths are **preserved and refined** from each other.

Bharath Hariharan, et al. "Hypercolumns for object segmentation and fine-grained localization." *CVPR*'15.

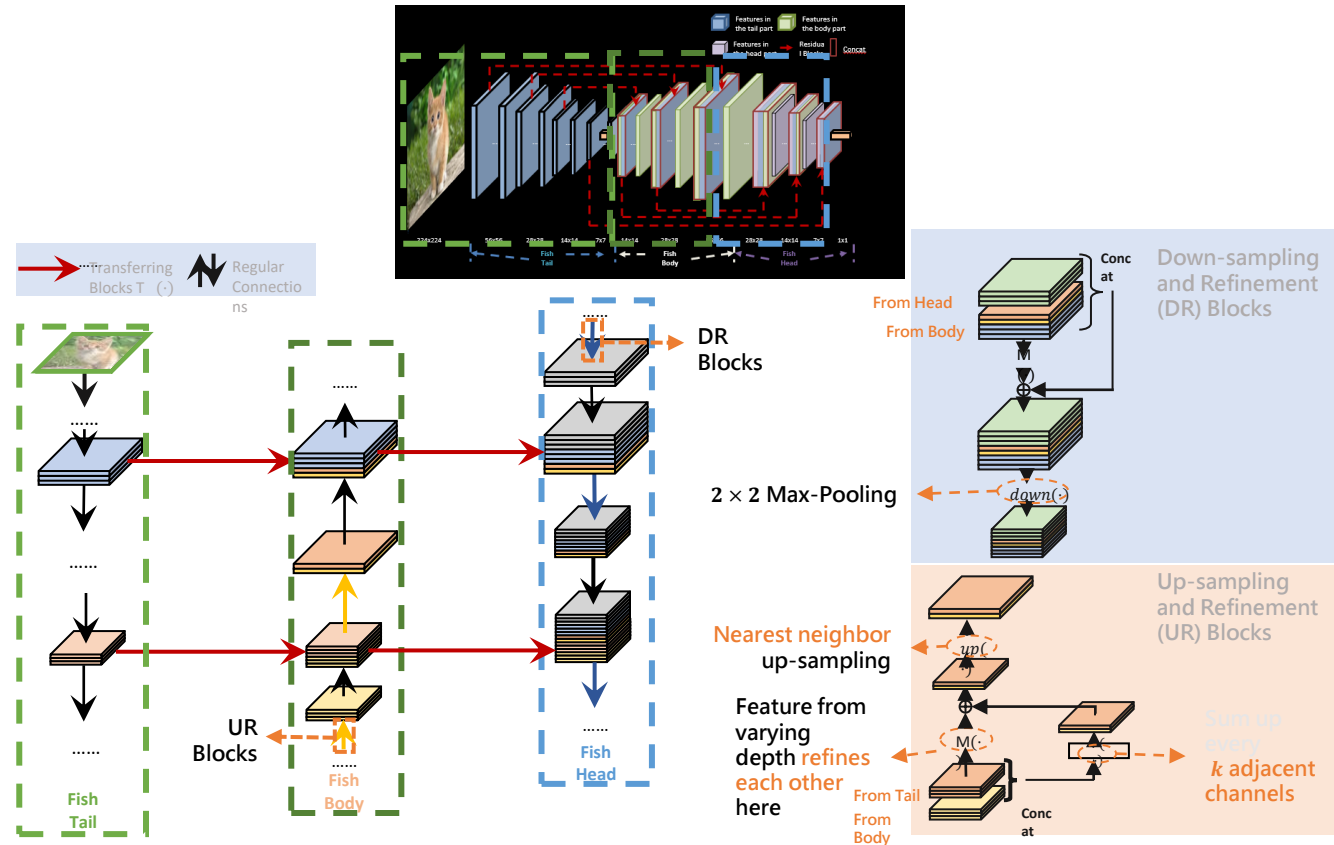
Newell, Alejandro, Kaiyu Yang, and Jia Deng. "Stacked hourglass networks for human pose estimation." *ECCV*'16.

FishNet: Overview



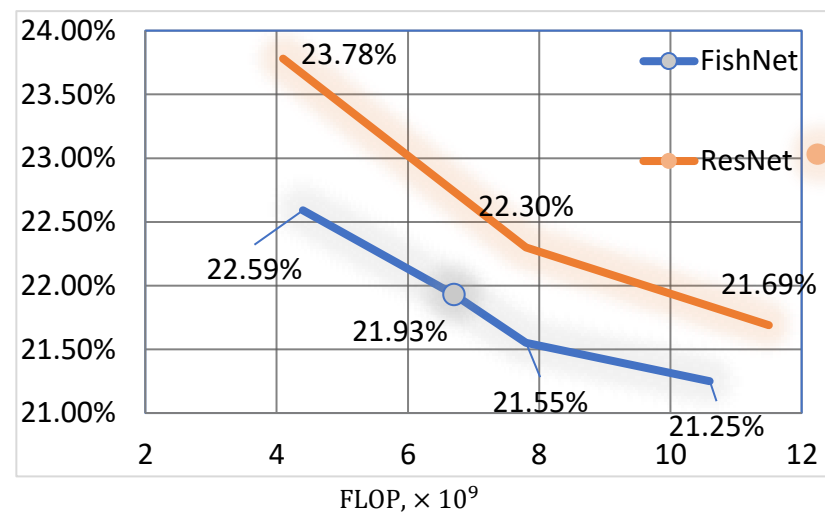
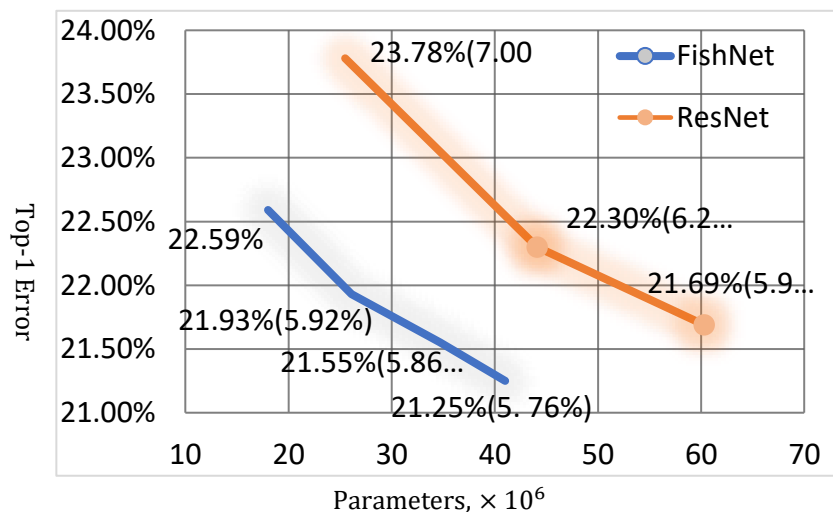
Shuyang Sun, Jiangmiao Pang, Jianping Shi, Shuai Yi, **Wanli Ouyang**, "FishNet: A Versatile Backbone for Image, Region, and Pixel Level Prediction," *NurIPS*. (Previously called *NIPS*), 2018.

FishNet: Preservation & Refinement



Shuyang Sun, Jiangmiao Pang, Jianping Shi, Shuai Yi, **Wanli Ouyang**, "FishNet: A Versatile Backbone for Image, Region, and Pixel Level Prediction," *NurIPS*. (Previously called *NIPS*), 2018.

FishNet: Performance-ImageNet

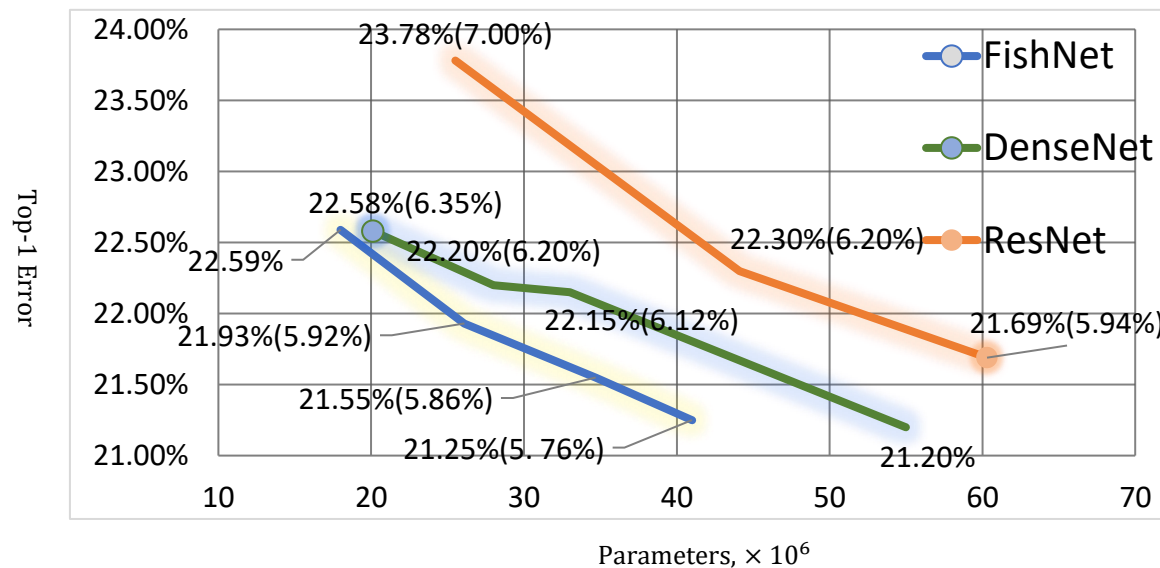


Code

<https://github.com/kevin-ssy/FishNet>

Shuyang Sun, Jiangmiao Pang, Jianping Shi, Shuai Yi, **Wanli Ouyang**, "FishNet: A Versatile Backbone for Image, Region, and Pixel Level Prediction," *NurIPS*. (Previously called *NIPS*), 2018.

FishNet: Performance-ImageNet

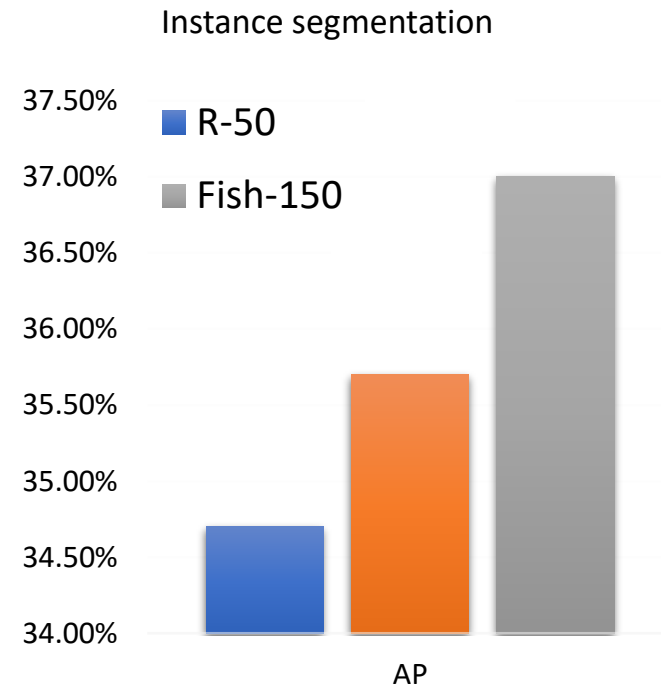
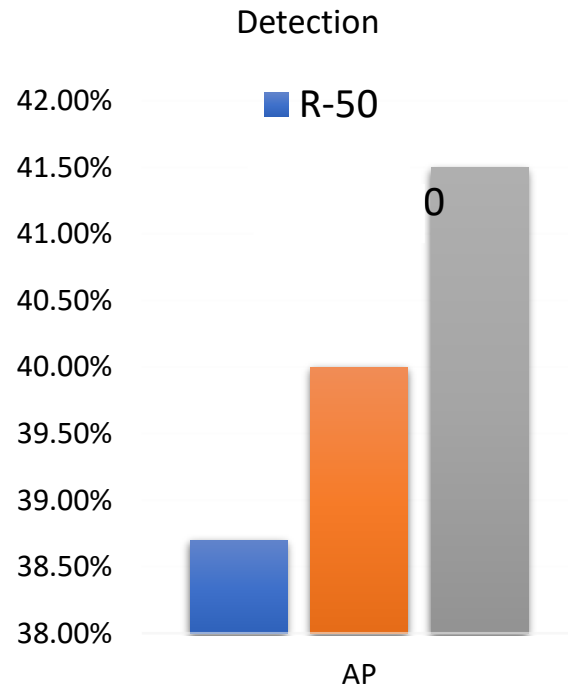


Code

<https://github.com/kevin-ssy/FishNet>

Shuyang Sun, Jiangmiao Pang, Jianping Shi, Shuai Yi, **Wanli Ouyang**, "FishNet: A Versatile Backbone for Image, Region, and Pixel Level Prediction," *NurIPS*. (Previously called *NIPS*), 2018.

FishNet: Performance on COCO Detection and Segmentation



Code

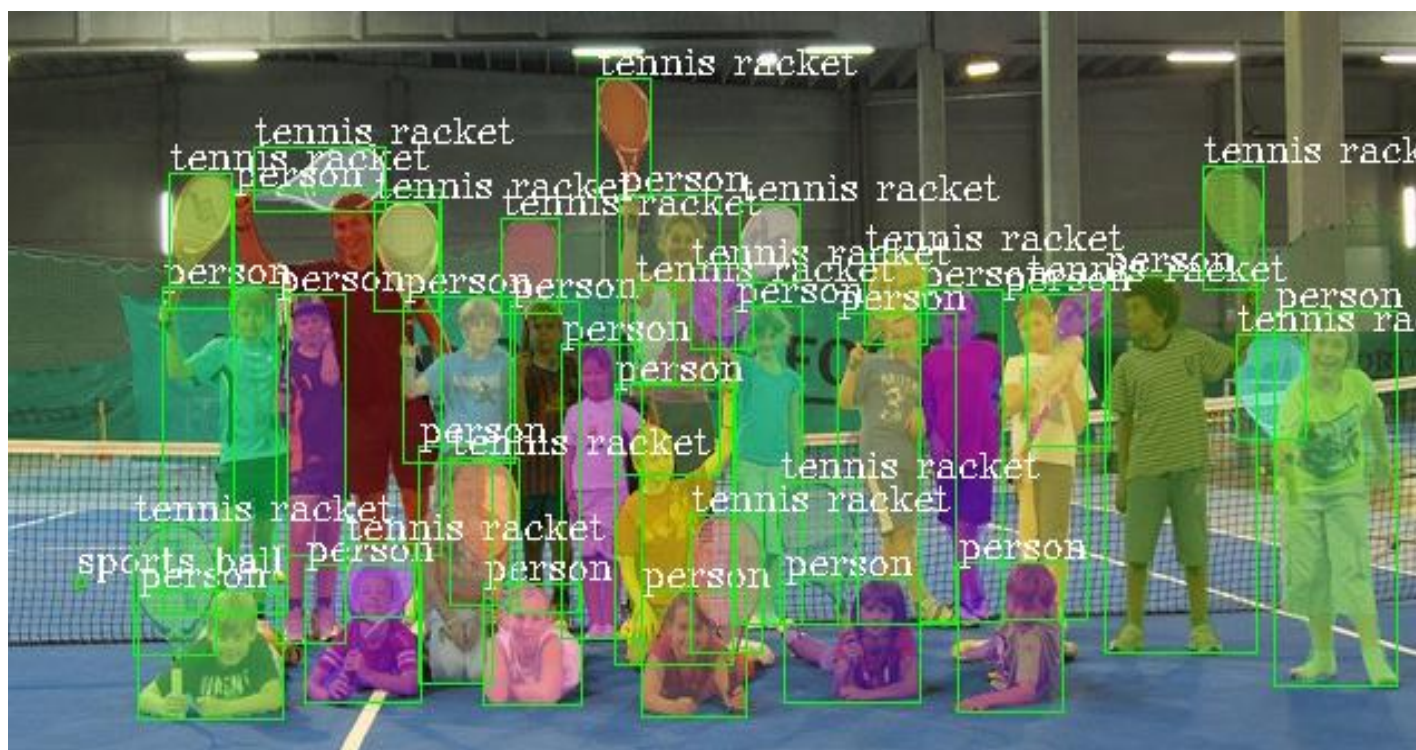
<https://github.com/kevin-ssy/FishNet>

Shuyang Sun, Jiangmiao Pang, Jianping Shi, Shuai Yi, **Wanli Ouyang**, "FishNet: A Versatile Backbone for Image, Region, and Pixel Level Prediction," *NurIPS. (Previously called NIPS)*, 2018.

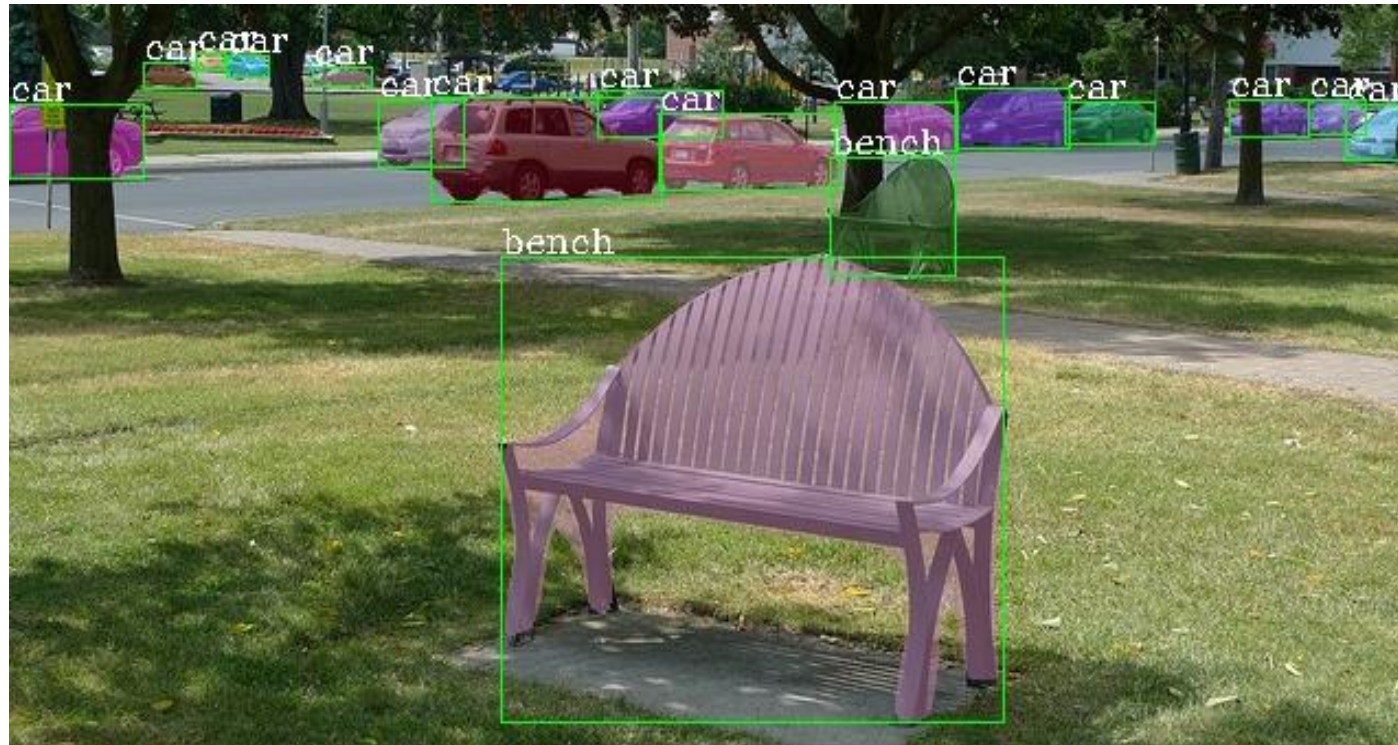
Winning COCO 2018 Instance Segmentation Task

	AP	AP ⁵⁰	AP ⁷⁵	AP ^S	AP ^M	AP ^L	AR ¹	AR ¹⁰	AR ¹⁰⁰	AR ^S	AR ^M	AR ^L	date
MMDet	0.486	0.730	0.530	0.339	0.520	0.602	0.368	0.593	0.632	0.464	0.665	0.777	2018-08-18
Megvii (Face++)	0.485	0.737	0.532	0.298	0.507	0.641	0.369	0.594	0.630	0.474	0.659	0.767	2018-08-18
FirstShot	0.463	0.681	0.508	0.258	0.483	0.636	0.359	0.580	0.622	0.445	0.655	0.776	2018-08-17

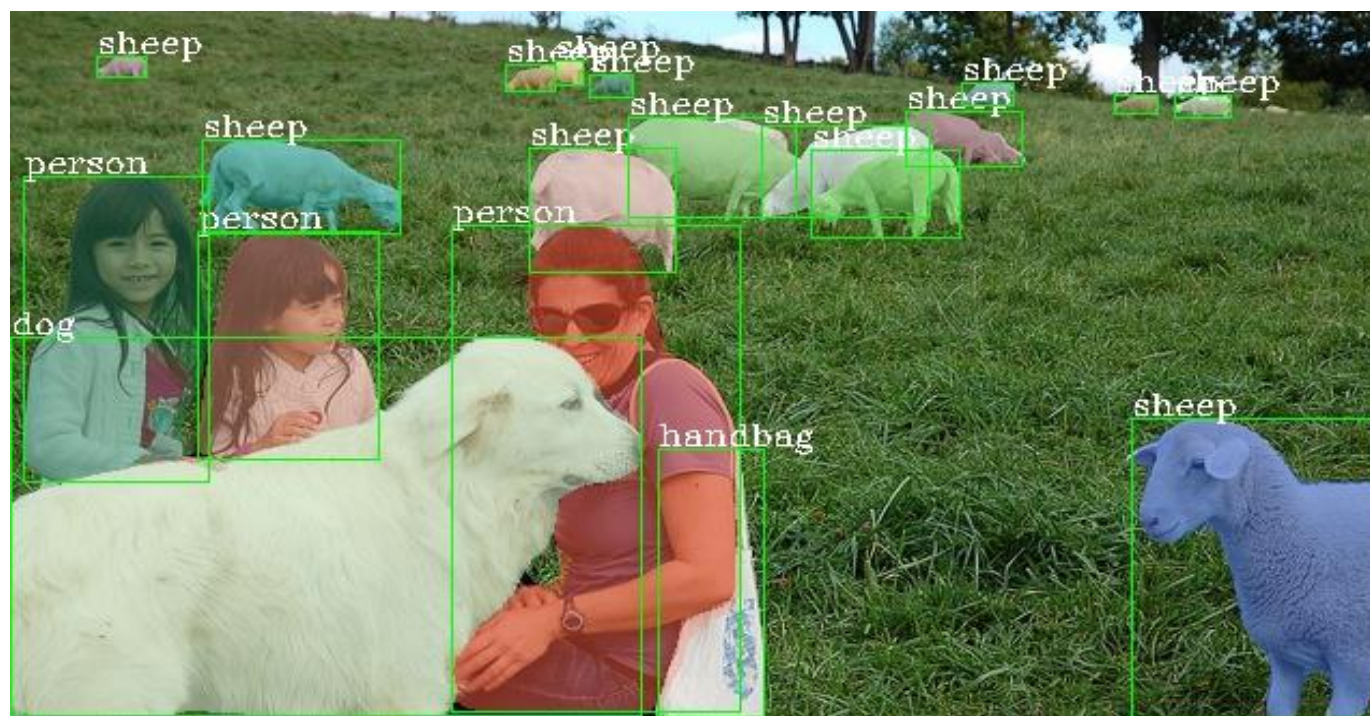
Visualization



Visualization



Visualization



Visualization



Visualization



Codebase



- **Comprehensive**

- ☒ FPN Fast/Faster ☒ R-CNN
- ☒ Mask R-CNN FPN ☒
- ☒ Cascade R-CNN RetinaNet ☒
- ☒ More

- **High performance**

- ☒ Better performance
- ☒ Optimized memory consumption
- ☒ Faster speed

- **Handy to develop**

- ☒ Written with PyTorch
- ☒ Modular design



[GitHub: mmdet](https://github.com/mmdetection/mmdet)

FishNet: Advantages

1. Better gradient flow to shallow layers

2. Features

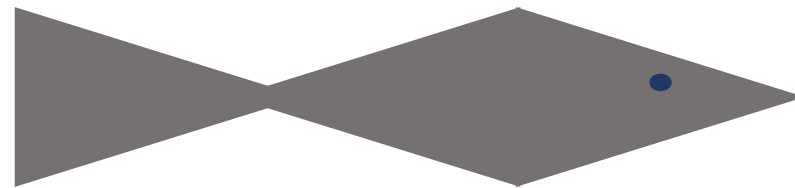
- contain rich low-level and high-level semantics
- are preserved and refined from each other



Code

<https://github.com/kevin-ssy/FishNet>

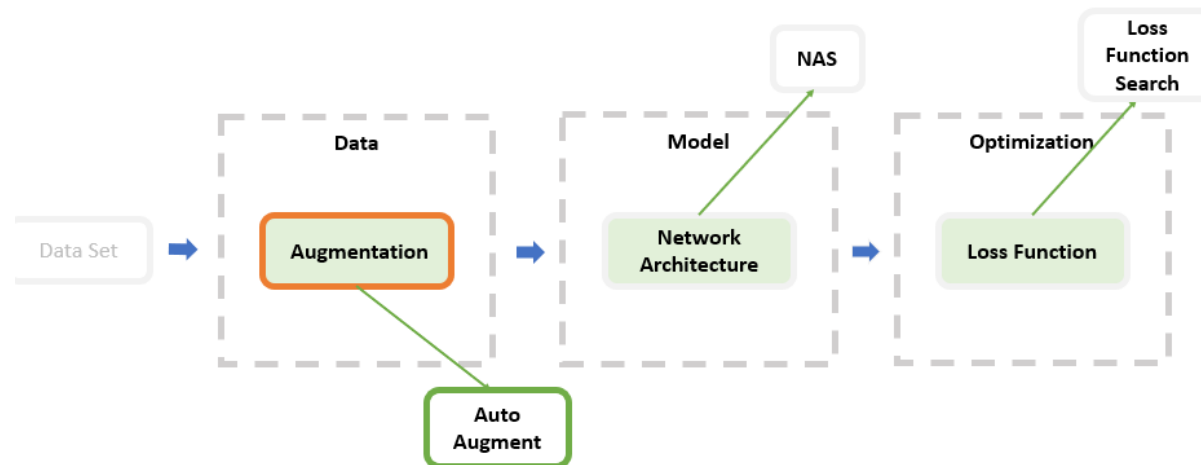
Outline



- Conclusion

Summary

- Deep learning is effective by automatically learning features
- For a specific task, Auto-ML can automatically find the proper network structure, data usage strategy, loss function, and more ...
- To tackle the problem from huge search space, online hyper-parameter learning (OHL) and proxy are good choices
- Manual design from observation is still important



Q&A

