AutoML in Full Life Circle of Deep Learning Assembly Line

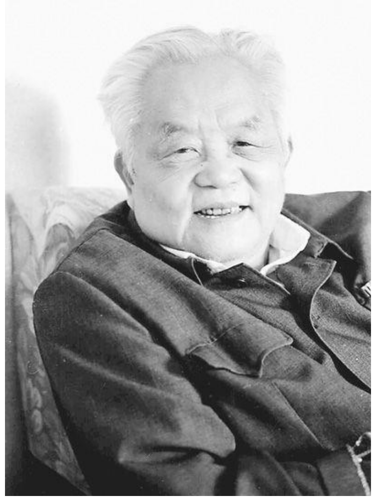
Junjie Yan
SenseTime Group Limited
2019/10/09

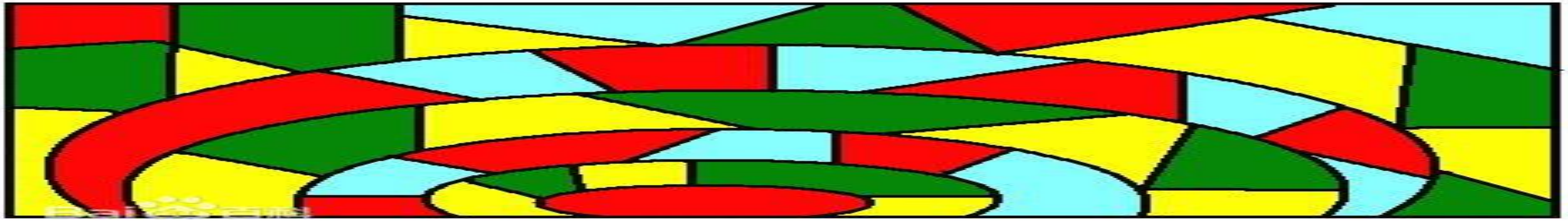
Works by AutoML Group @ SenseTime Research

A Brief History of Axiomatic System

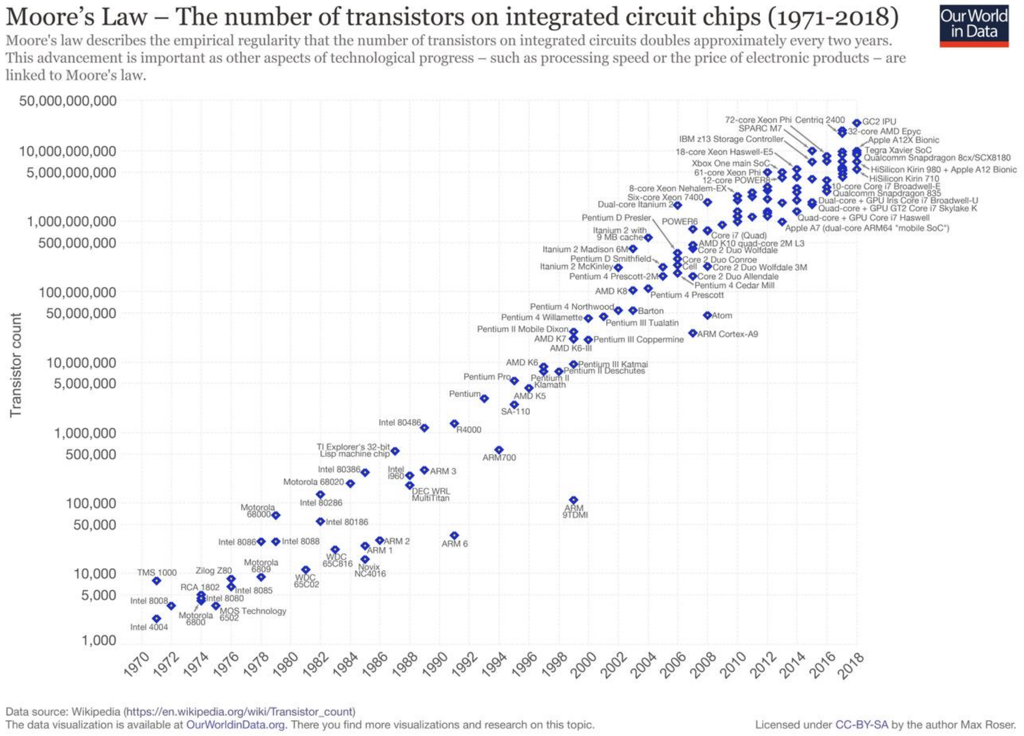


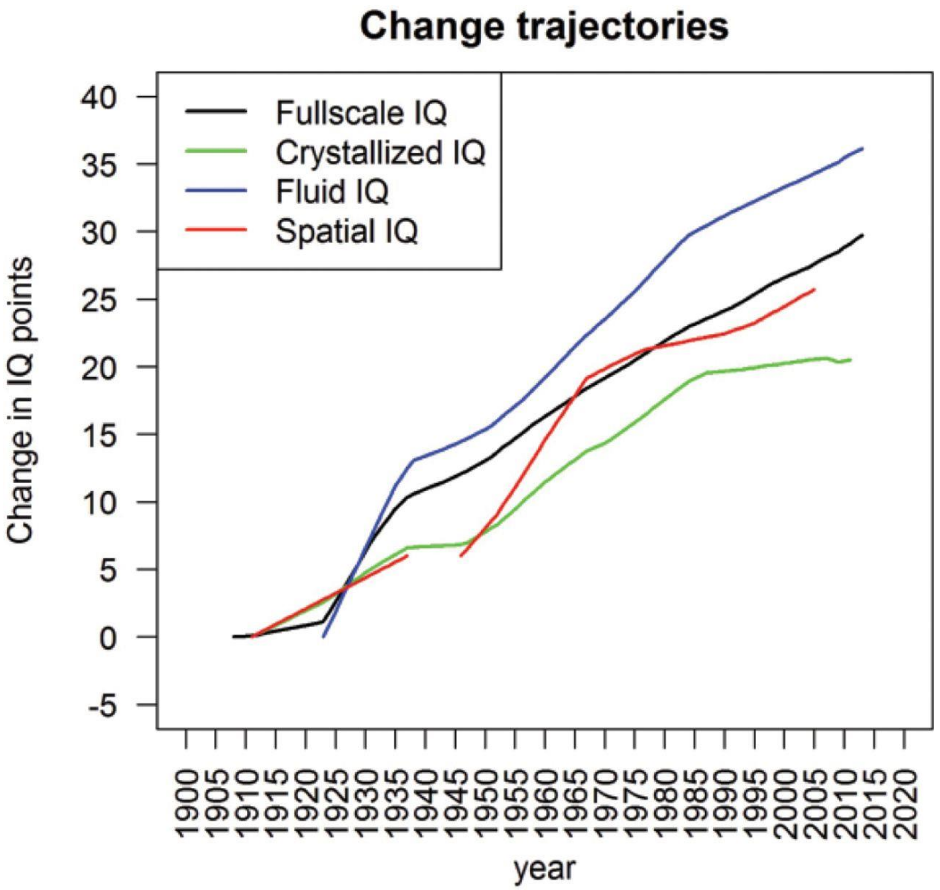






Why AutoML



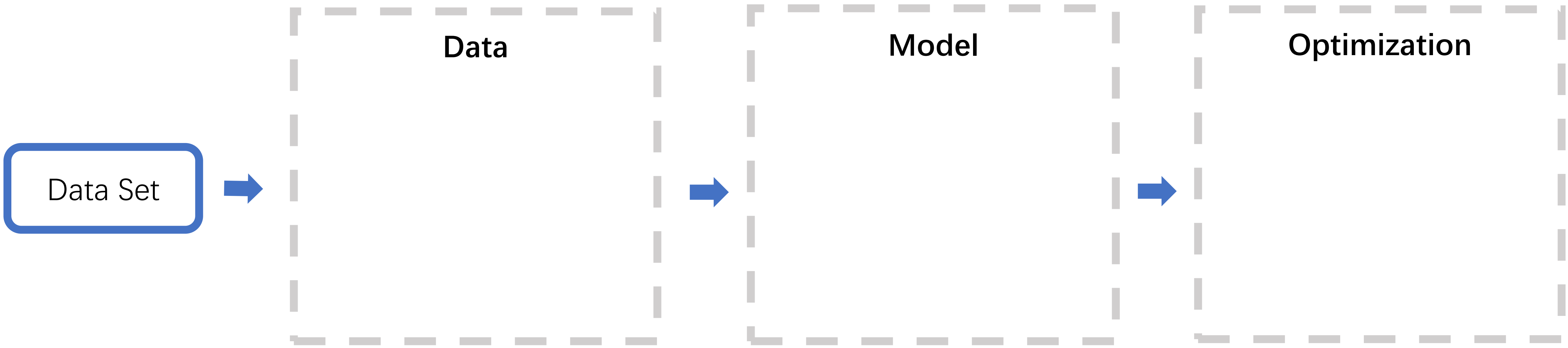


Moore Law

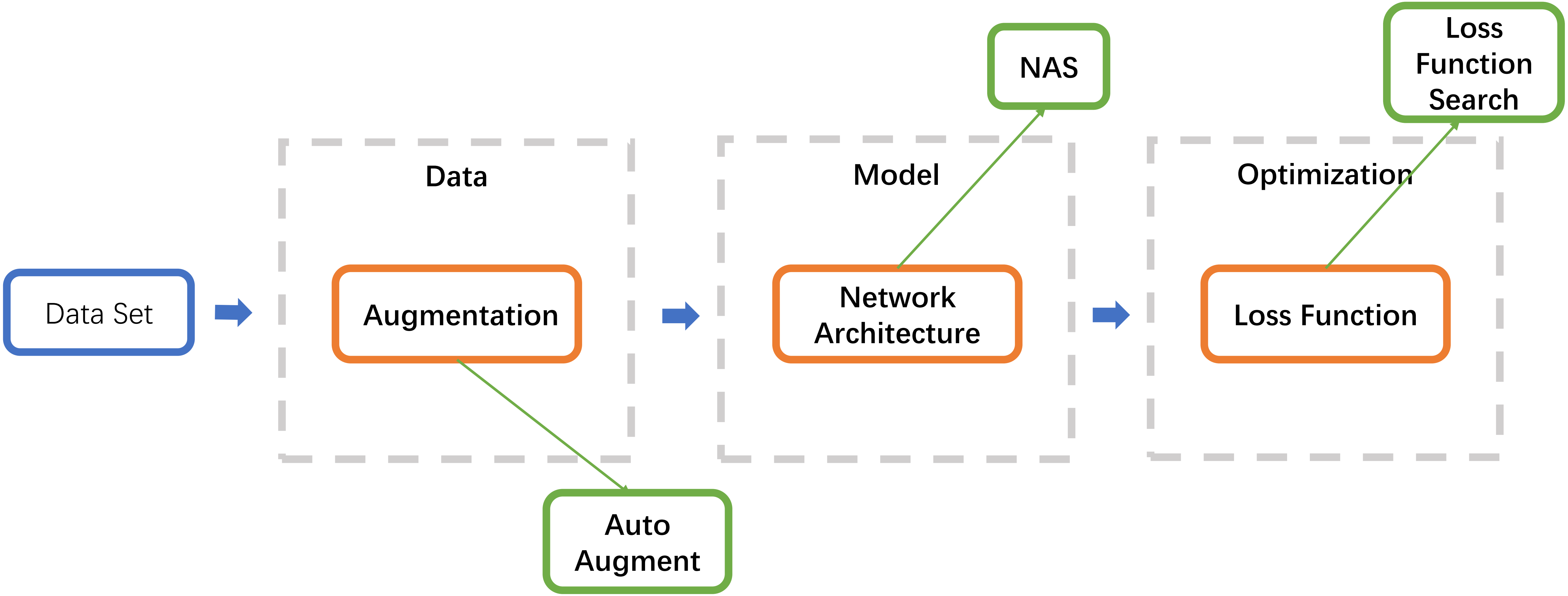
V.S.

Flynn Effect

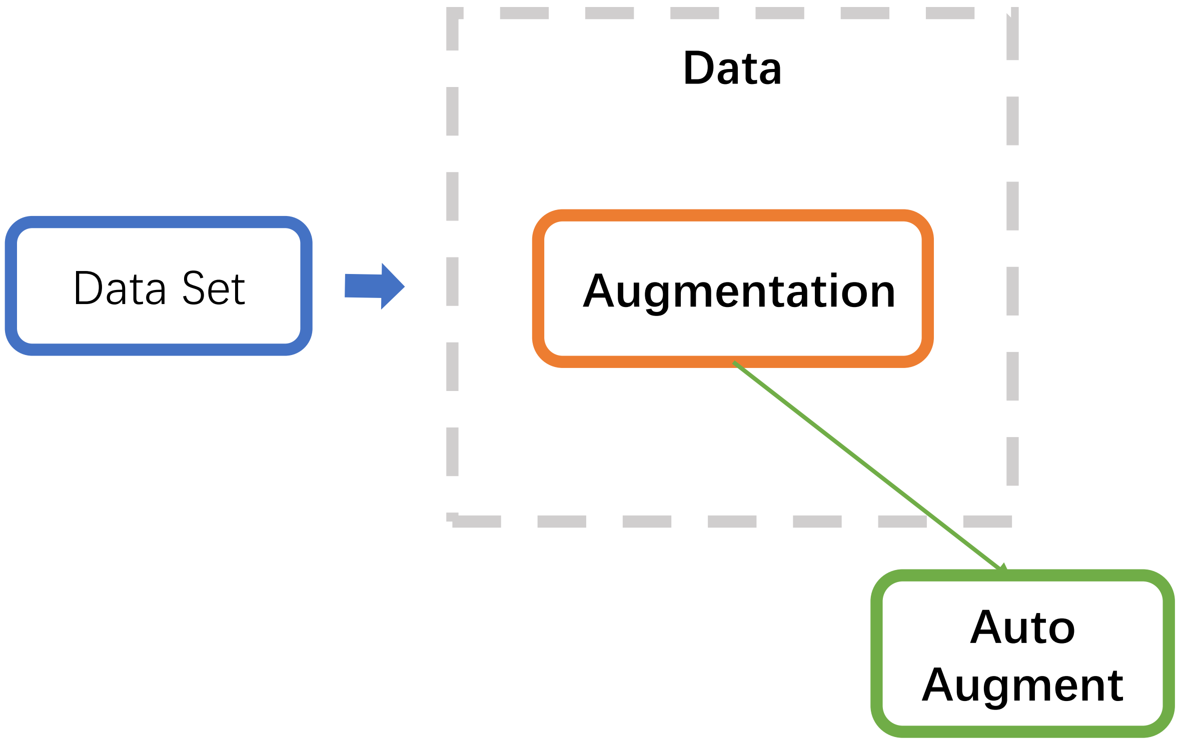
Deep Learning Assembly Line



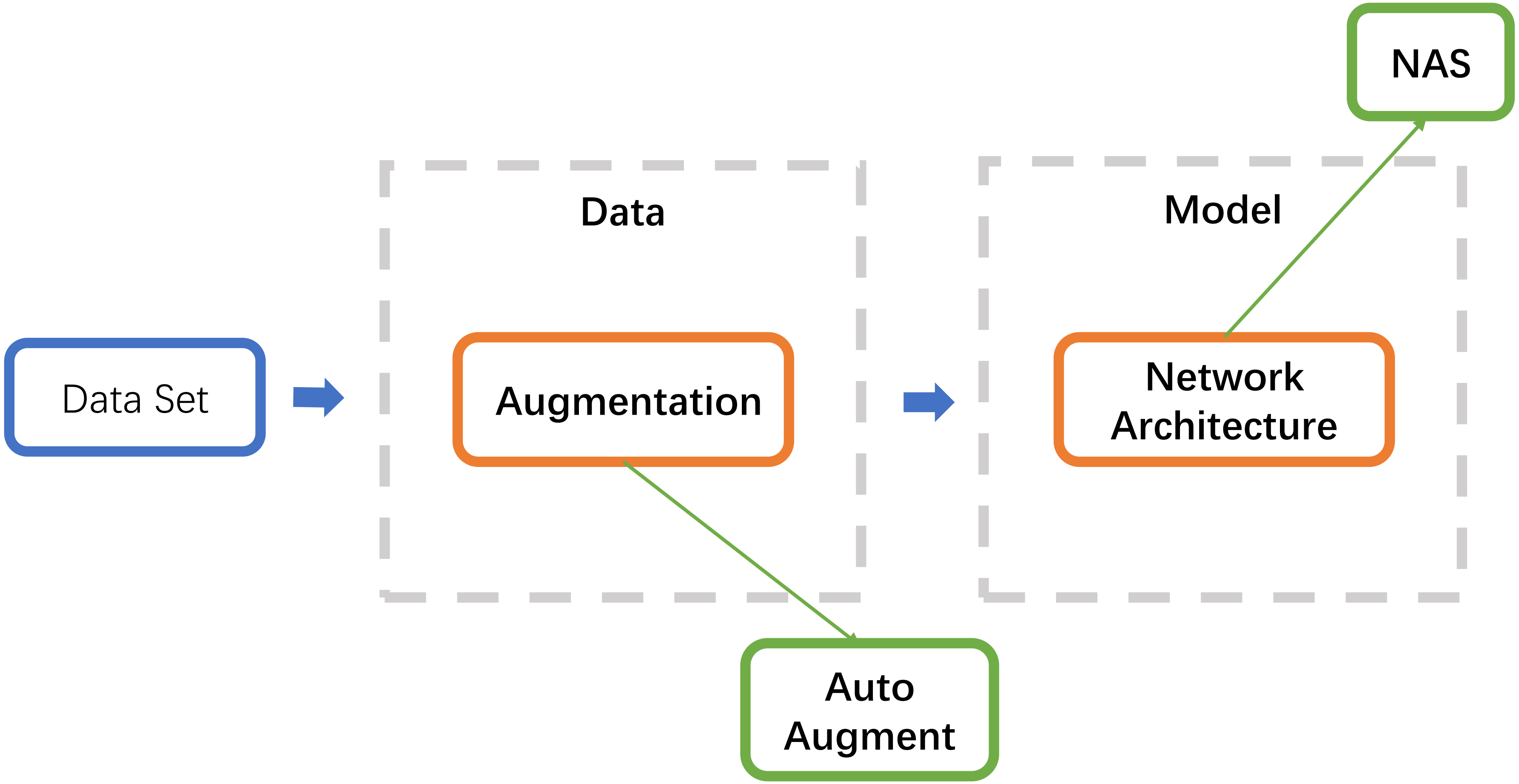
Deep Learning Assembly Line



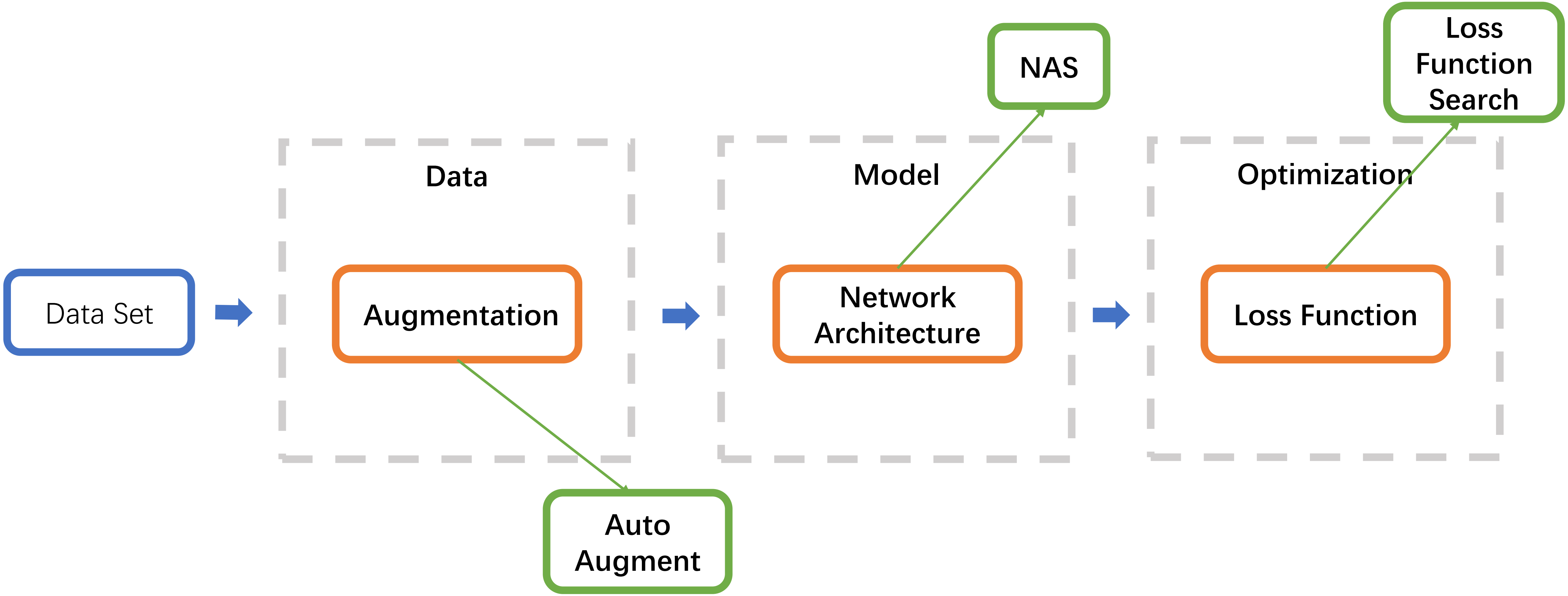
Deep Learning Assembly Line



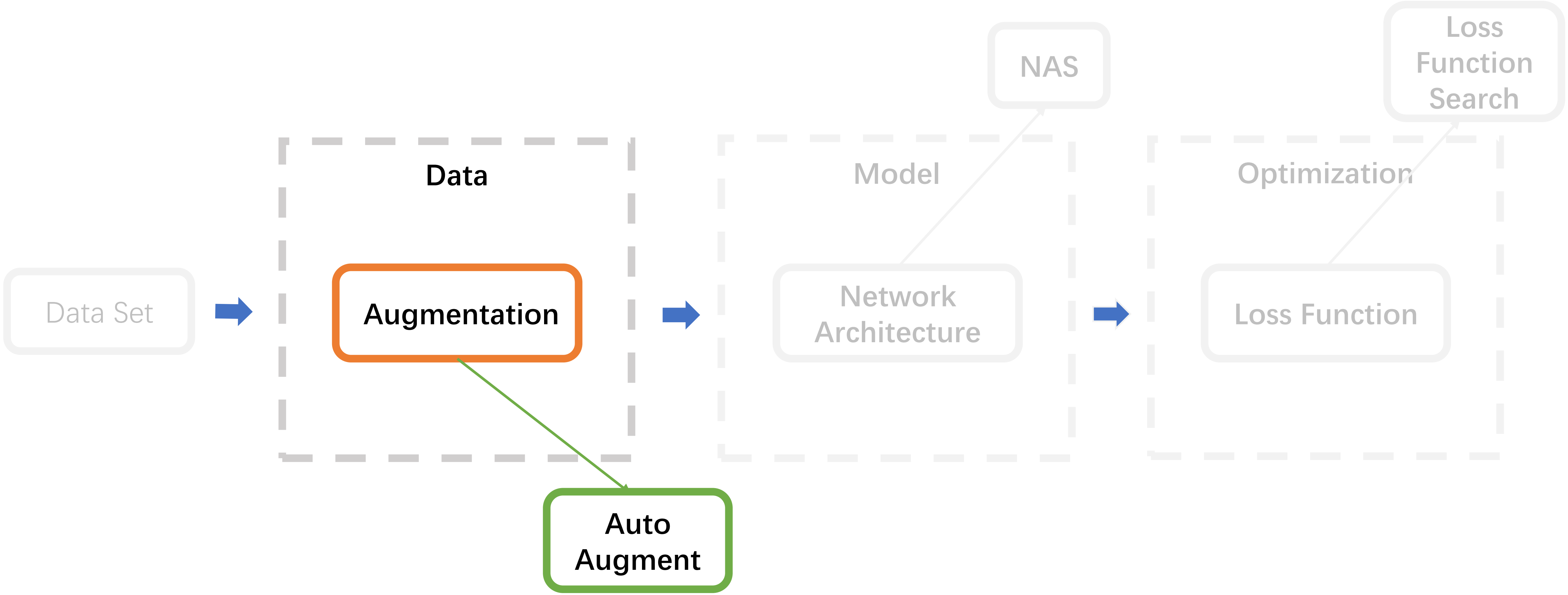
Deep Learning Assembly Line



Deep Learning Assembly Line



Deep Learning Assembly Line



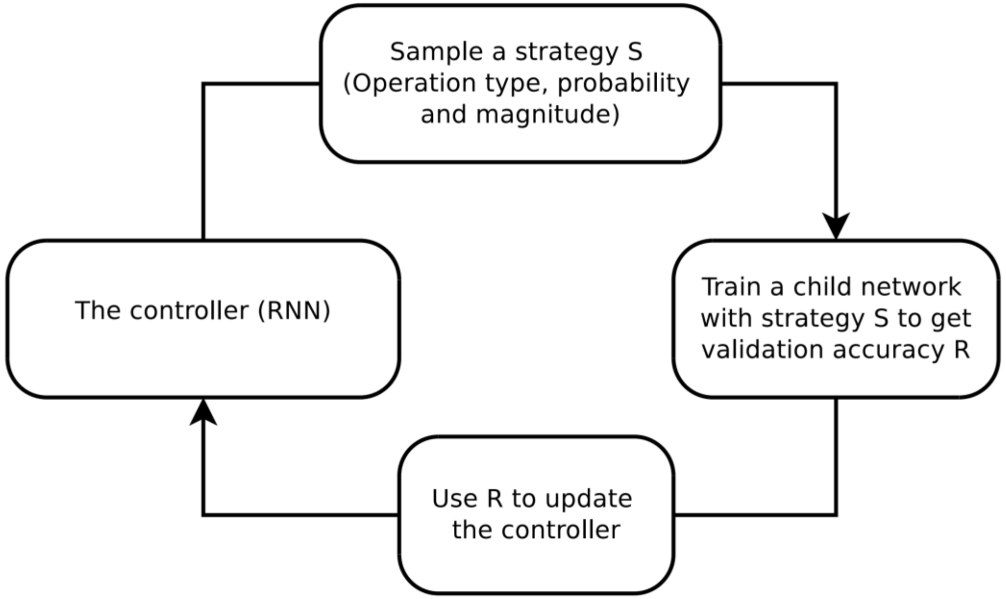
Online Hyper-parameter Learning for Auto-Augmentation Strategy

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

ICCV 2019

Auto-augment search – Existing work

- Previous Auto-augment search policy on a subsampled dataset and a predefined CNN
  - Data:
    - CIFAR-10: 8% subsampled
    - IMAGENET: 0.5% subsampled
  - Network:
    - CIFAR-10: WideResNet-40-2 （small）
    - IMAGENET: Wide-ResNet 40-2
- Suboptimal and not general well



Auto-augment search – Motivation

- Difficulty:
  - Slow evaluation of certain augmentation policy
  - Slow convergence of RL due to the RNN controller
- Solution: Treat augmentation policy search as a hyper-parameter optimization

Lin, Chen, Minghao Guo, Chuming Li, Wei Wu, Dahua Lin, Wanli Ouyang, and Junjie Yan. "Online Hyper-parameter Learning for Auto-Augmentation Strategy." *ICCV*19.

Hyperparameter Learning

- Unlike CNN architecture, which is transferable across different dataset, hyper-parameters in training strategy is KNOWN to be deeply coupled with specific dataset and underlying network architecture.
- Usually the hyper-parameters are not differentiable wrt validation loss.
- Full evaluation based method using reinforcement learning, evolution, Bayesian optimization is computational expensive and implausible to apply on industrial-scaled dataset

Online Hyperparameter Learning (OHL)

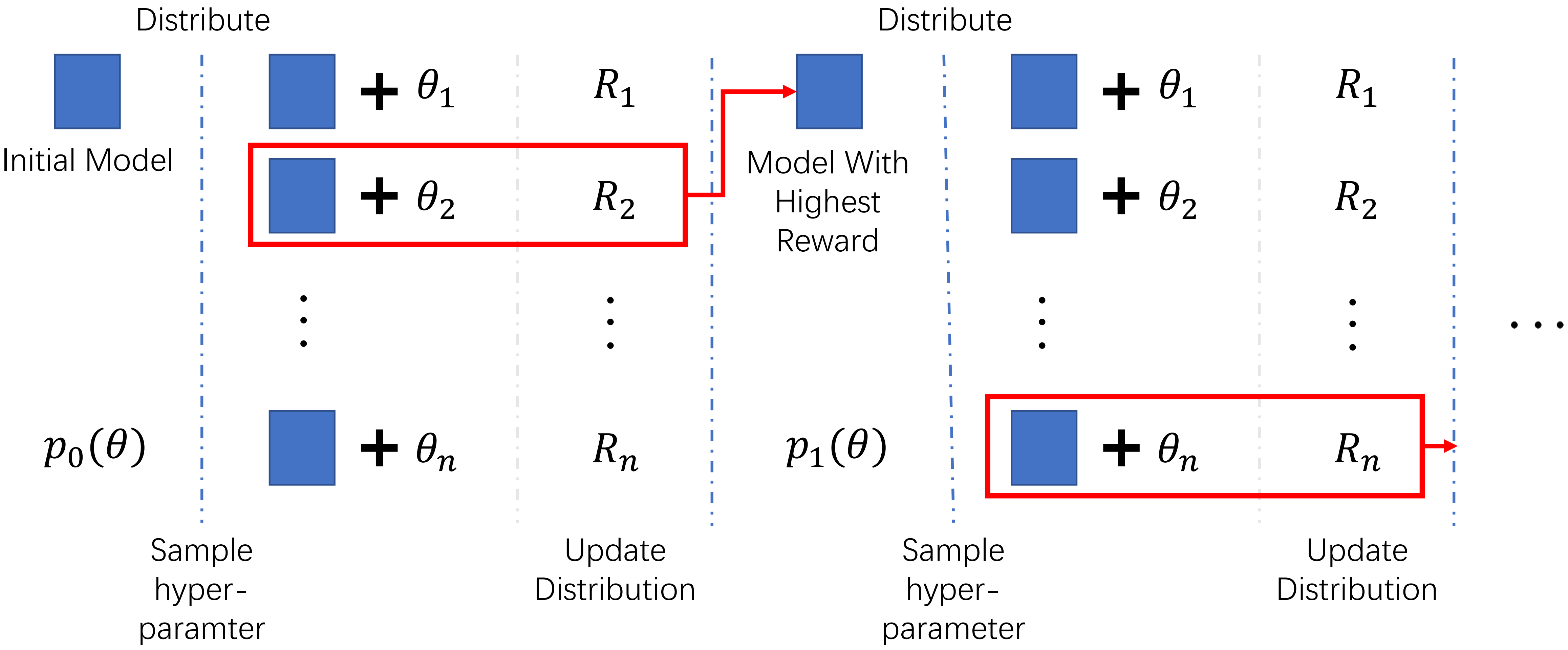
- What is OHL
  - Online Hyper-parameter Learning aims to learning the best hyper-parameter within only a **single** run.
  - While learning the hyper-parameters, it improves the performance of the model at mean time.

Online Hyperparameter Learning (OHL)

- How does OHL work:
  - Hyper-parameter is modeled as stochastic variables.
  - Split the training stage into trunks
  - Run multiple copy 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

$p_0(\theta)$: Initial Distribution



Augmentation as hyperparameter

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

Table 1. List of Candidate Augmentation Elements

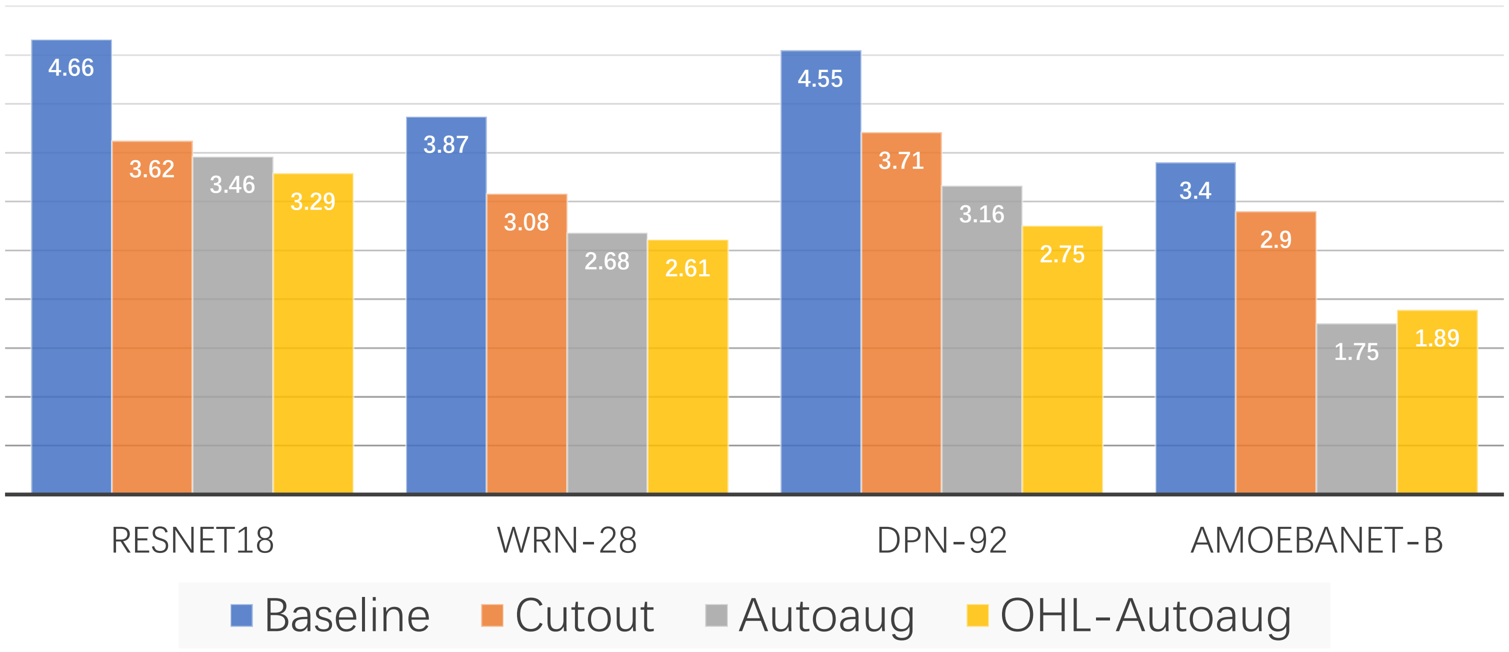
Elements Name	range of magnitude
$HorizontalShear$	$\{0.1, 0.2, 0.3\}$
$VerticalShear$	$\{0.1, 0.2, 0.3\}$
$HorizontalTranslate$	$\{0.15, 0.3, 0.45\}$
$VerticalTranslate$	$\{0.15, 0.3, 0.45\}$
$Rotate$	$\{10, 20, 30\}$
$ColorAdjust$	$\{0.3, 0.6, 0.9\}$
$Posterize$	$\{4.4, 5.6, 6.8\}$
$Solarize$	$\{26, 102, 179\}$
$Contrast$	$\{1.3, 1.6, 1.9\}$
$Sharpness$	$\{1.3, 1.6, 1.9\}$
$Brightness$	$\{1.3, 1.6, 1.9\}$
$AutoContrast$	None
$Equalize$	None
$Invert$	None

- 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

Lin, Chen, Minghao Guo, Chuming Li, Wei Wu, Dahua Lin, Wanli Ouyang, and Junjie Yan. "Online Hyper-parameter Learning for Auto-Augmentation Strategy." *ICCV*19.

Experimental Results - CIFAR10

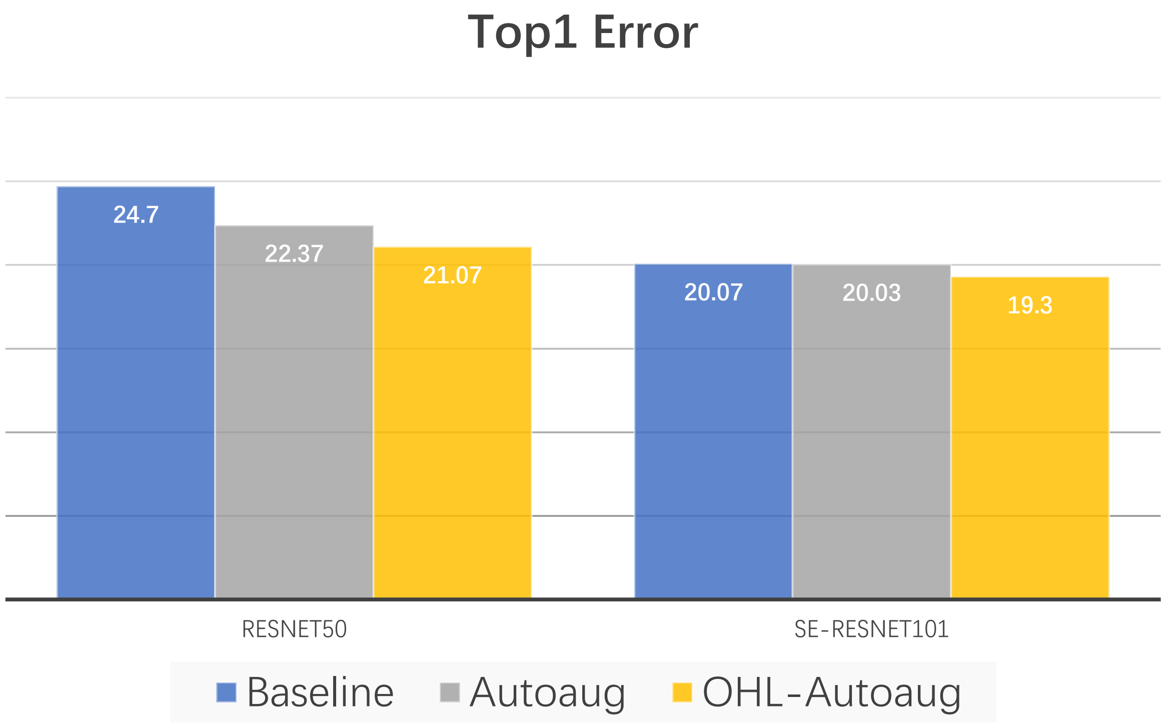
- Using OHL, we train our performance model while learning alpha at the same time.
  - On CIFAR10 (Top1 Error)



Lin, Chen, Minghao Guo, Chuming Li, Wei Wu, Dahua Lin, Wanli Ouyang, and Junjie Yan. "Online Hyper-parameter Learning for Auto-Augmentation Strategy." *ICCV*19.

Experimental Results - ImageNet

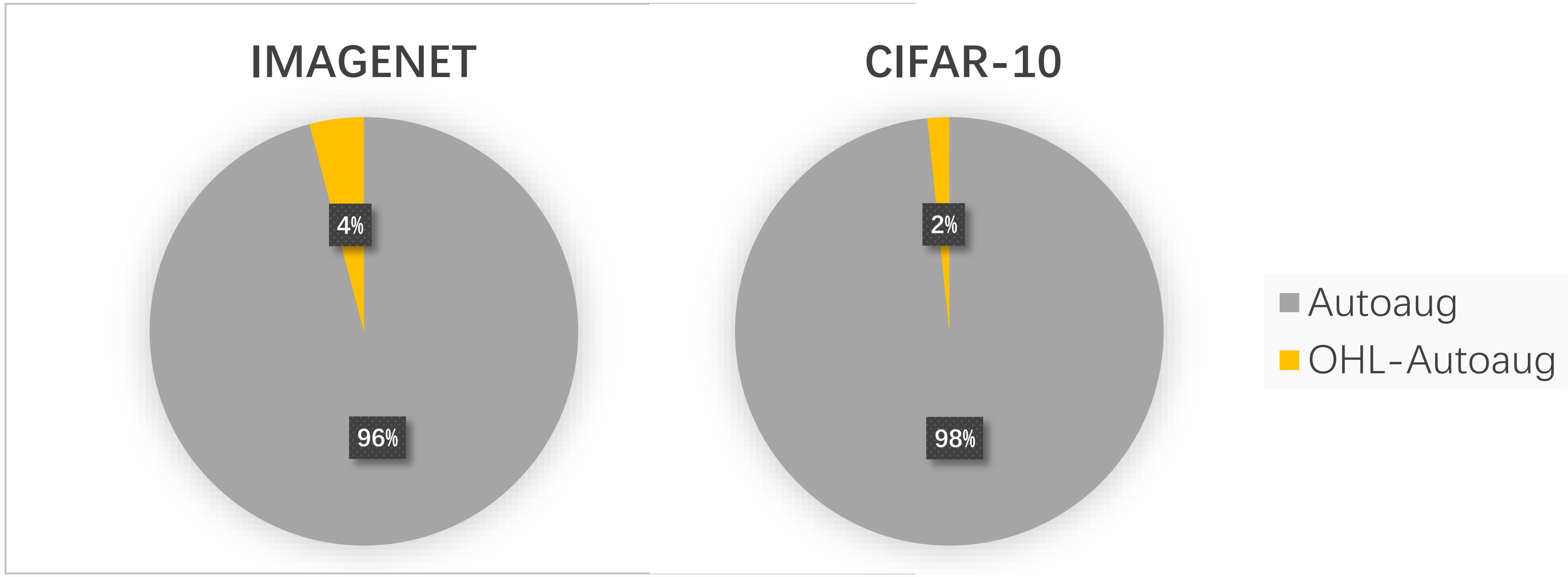
- On ImageNet (Top1/Top5 Error)



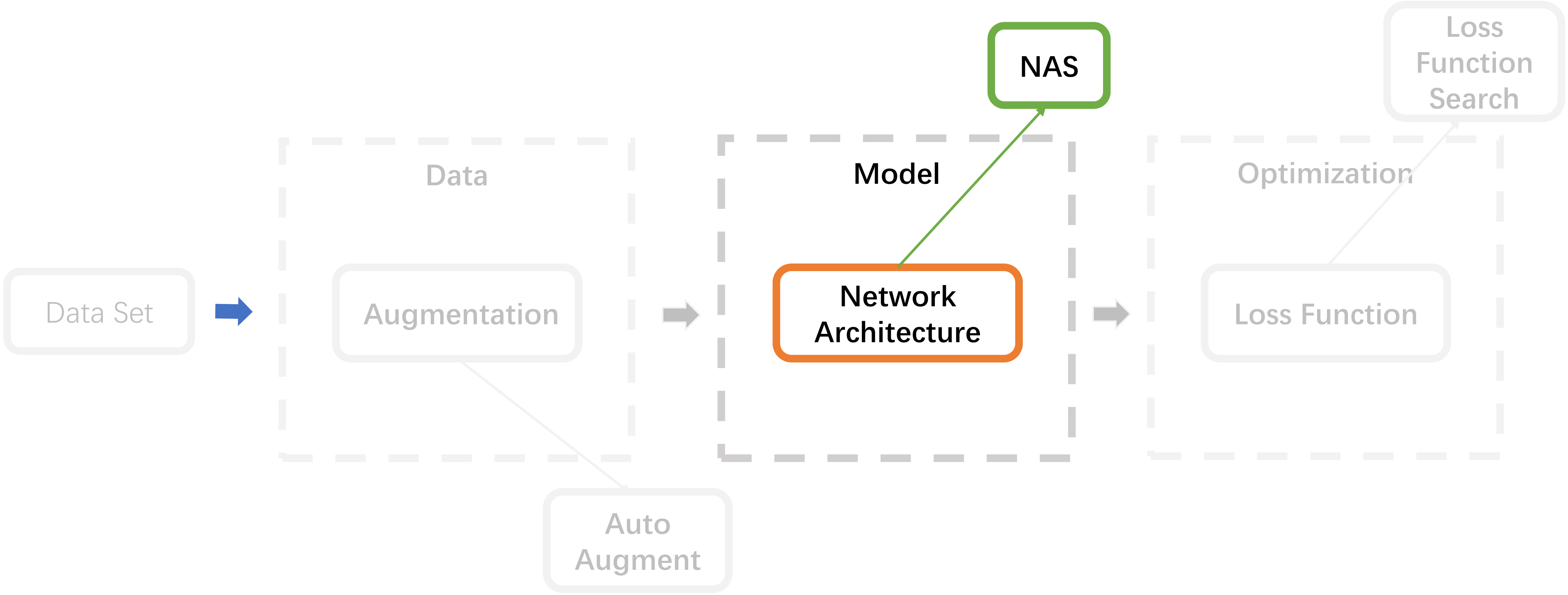
Lin, Chen, Minghao Guo, Chuming Li, Wei Wu, Dahua Lin, Wanli Ouyang, and Junjie Yan. "Online Hyper-parameter Learning for Auto-Augmentation Strategy." *ICCV*19.

Computation Required vs Offline Learning

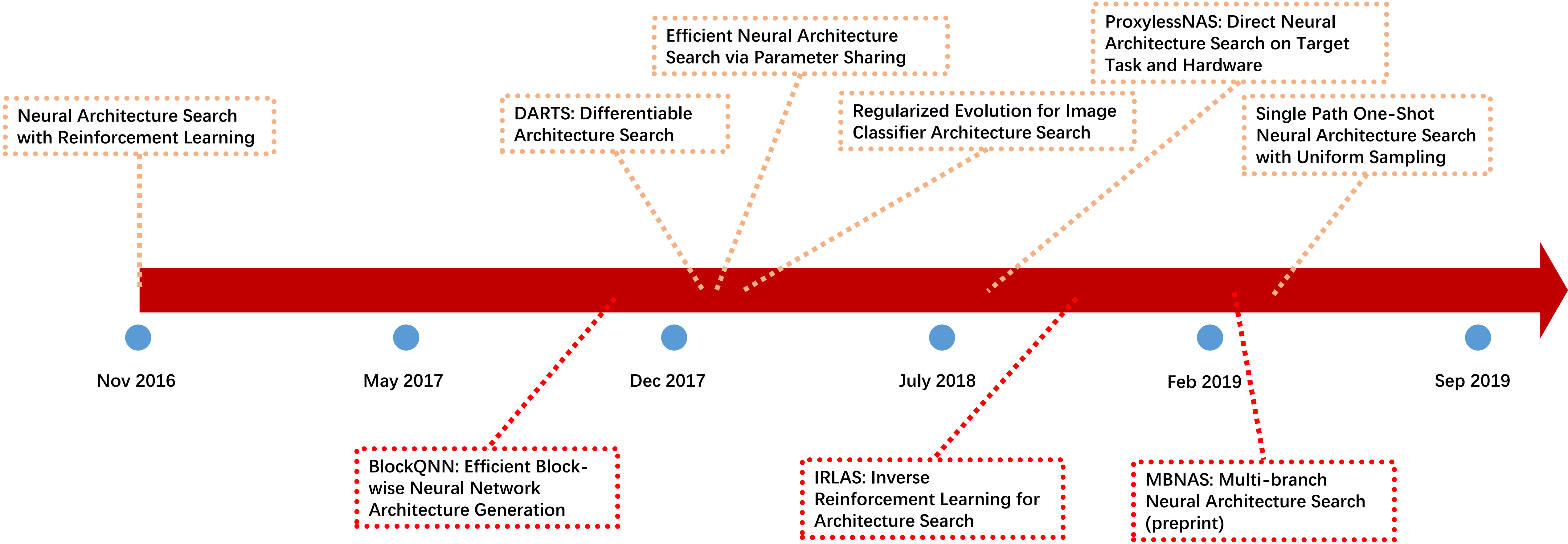
Dataset	Auto-Augment [6]		OHL-Auto-Aug	
	$\#Iterations$	Usage of Dataset (%)	$\#Iterations$	Usage of Dataset (%)
CIFAR-10	7.03×10^6	8%	1.17×10^5	100%
ImageNet	1.76×10^7	0.5%	7.5×10^5	100%
No Need to Retrain	×		✓	



Deep Learning Assembly Line



Time Line of SenseTime NAS



Improving One-Shot NAS By Suppressing The Posterior Fading

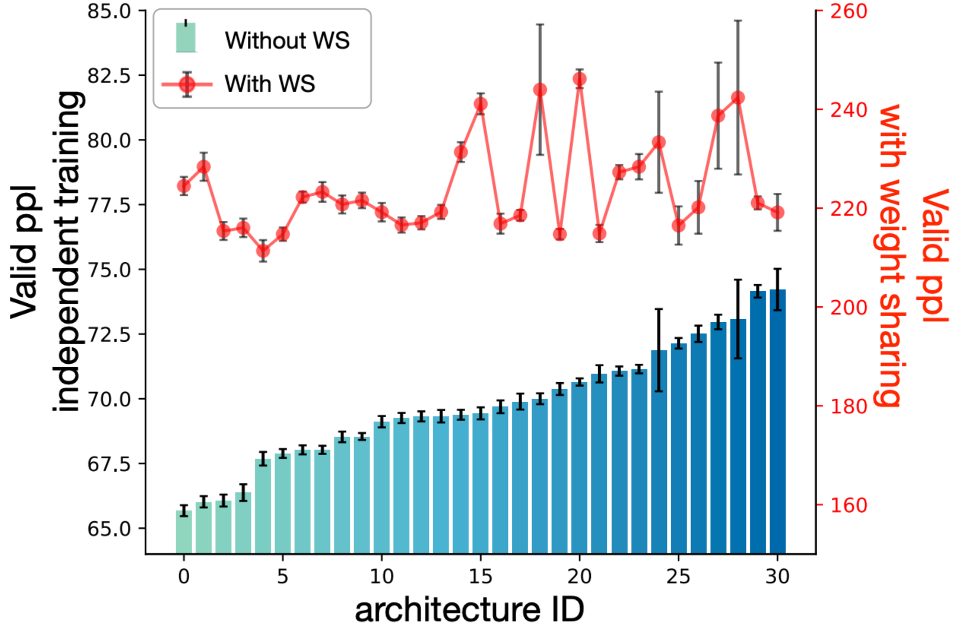
Xiang Li*, Chen Lin*, Chuming Li, Ming Sun, Wei Wu,

Junjie Yan, Wanli Ouyang

Preprint.

Posterior Convergent NAS

- What wrong with the parameter sharing approach:
  - All candidate models share the same set of parameters during training.
  - Such parameters performs poor in ranking models.



(a) Architecture performance with and without WS

*Christian Sciuto, Swisscom Kaicheng Yu, Martin Jaggi and Mathieu Salzmann. "Evaluating the Search Phase of Neural Architecture Search" https://arxiv.org/pdf/1902.08142.pdf.

Posterior Convergent NAS

- Compute the KL-divergence of the parameter distribution of a single operator (operator o at l-th layer) trained alone or share weights under certain independence assumption:

$$D_{\mathcal{KL}}\left(p_{\text{alone}}(\theta_{l,o}|\mathbf{m}_k,\mathcal{D})\;\middle|\middle|\;p_{\text{share}}(\theta_{l,o}|\mathcal{D})\right)$$

$$=\sum_{i\neq k}-\int p_{\text{alone}}(\theta_{l,o}|\mathbf{m}_k,\mathcal{D})\log p_{\text{alone}}(\theta_{l,o}|\mathbf{m}_i,\mathcal{D})\mathrm{d}\theta.$$

Xiang Li*, Chen Lin*, Chuming Li, Ming Sun, Wei Wu, Junjie Yan, Wanli Ouyang. "Improving One-Shot NAS By Suppressing The Posterior Fading"Preprint

Posterior Convergent NAS

- The KL of share weights posterior and train alone posterior is just the sum of cross-entropy (Posterior Fading).
- It is suggested that having less possible models in the share weights could reduce the dis-alignment.

Xiang Li*, Chen Lin*, Chuming Li, Ming Sun, Wei Wu, Junjie Yan, Wanli Ouyang. “Improving One-Shot NAS By Suppressing The Posterior Fading”Preprint

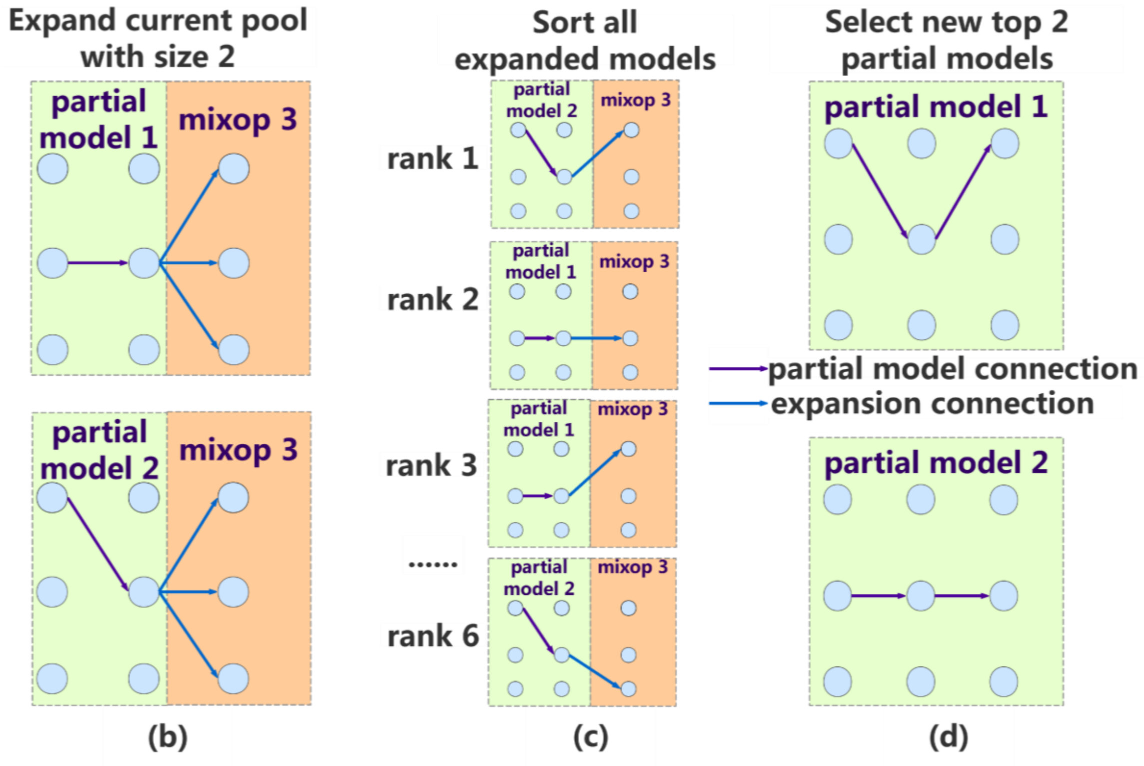
Posterior Convergent NAS

- Implementation:
  - Guide the posterior to converge to its true distribution!
  - Progressively shrink the search space to mitigate the divergence.
  - For a layer-by-layer search space, the combinations of operators in early layers are reduced to a fixed set when models are sampled for training.
  - The depth of fixed layers grows from 0 to full length during training.
  - At last, the fixed set of combinations are the resulted models.

Xiang Li*, Chen Lin*, Chuming Li, Ming Sun, Wei Wu, Junjie Yan, Wanli Ouyang. "Improving One-Shot NAS By Suppressing The Posterior Fading"Preprint

Posterior Convergent NAS

- Implemented using Multiple Training Stage & Partial Model Pool
  - The training is divided into multiple stages.
  - During the i-th stage, models are uniformly sampled, with the earlier i layers sampled from the partial model pool.
  - After the i-th stage, the pool updated by expanding its partial models by one layer and selecting the top-K partial model.



Xiang Li*, Chen Lin*, Chuming Li, Ming Sun, Wei Wu, Junjie Yan, Wanli Ouyang. "Improving One-Shot NAS By Suppressing The Posterior Fading" Preprint

Posterior Convergent NAS

- Evaluation of the partial models
  - We estimate the average validation accuracy of partial models by uniform sampling the unspecified layers.
  - The latency cost is computed for each architecture sample. The architecture with unsatisfied latency would be removed from the average computation.

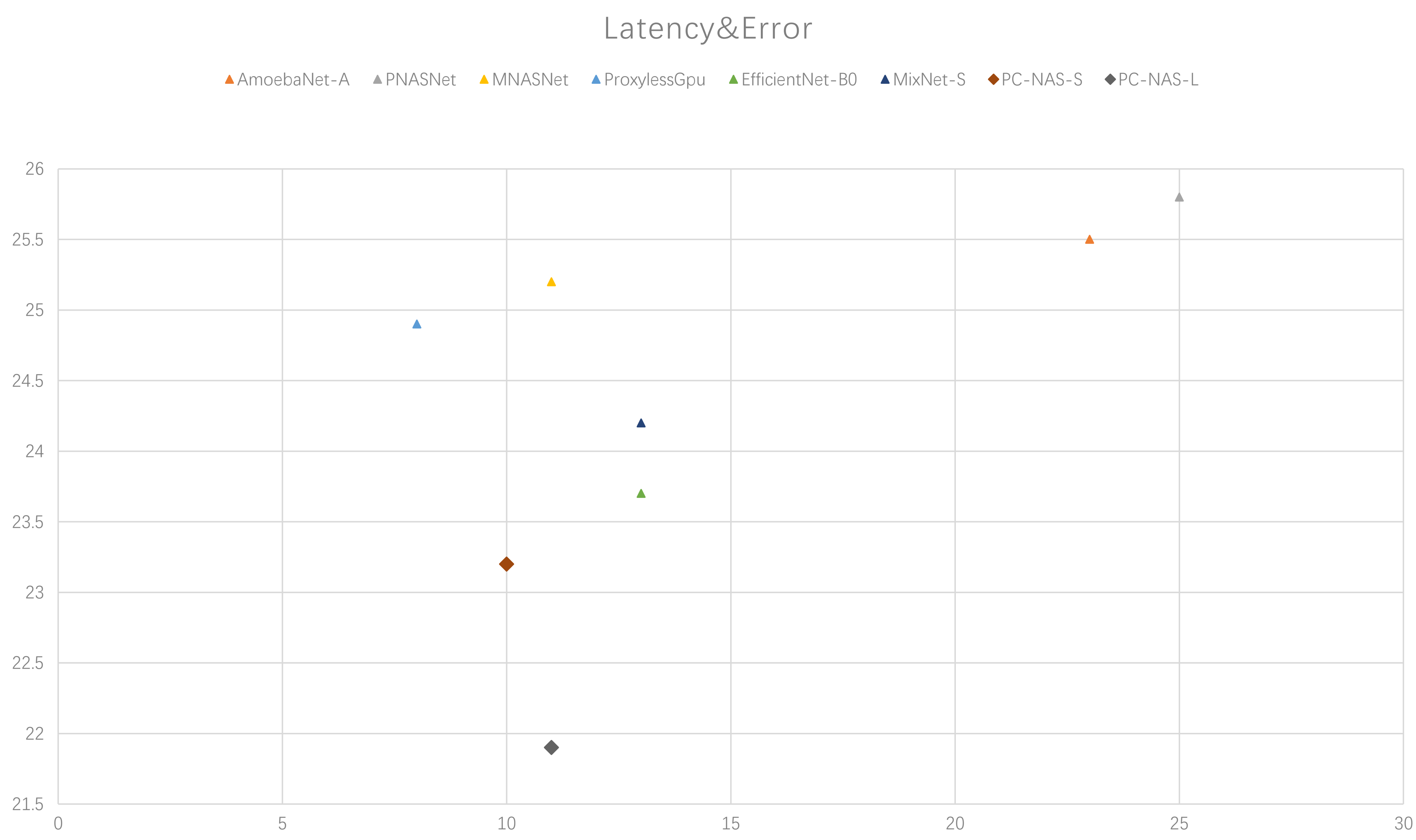
$$\text{Potential}(o_1, o_2, ..., o_l) = E_{\mathbf{m} \in \{\mathbf{m} | m_i = o_i, \forall i \leq l\}}(Acc(\mathbf{m})).$$

Xiang Li*, Chen Lin*, Chuming Li, Ming Sun, Wei Wu, Junjie Yan, Wanli Ouyang. “Improving One-Shot NAS By Suppressing The Posterior Fading”Preprint

Posterior Convergent NAS

- It benefits the later stage of search to have fewer possible models.
- The method has been applied to search for imagenet small gpu models with 10 ms latency constraint.
- Two search space had been tested.
  - PC-NAS-S: search result of “small search space”
  - PC-NAS-L: search result of “big search space”

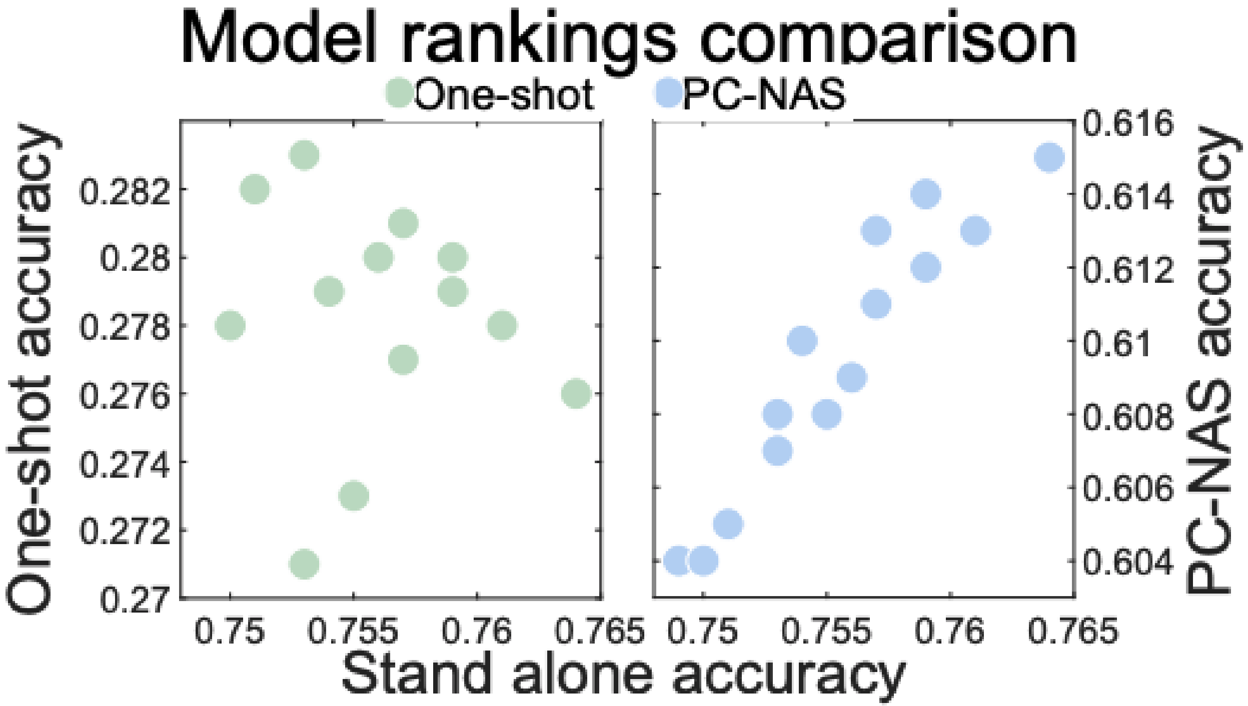
Xiang Li*, Chen Lin*, Chuming Li, Ming Sun, Wei Wu, Junjie Yan, Wanli Ouyang. “Improving One-Shot NAS By Suppressing The Posterior Fading”Preprint



Xiang Li*, Chen Lin*, Chuming Li, Ming Sun, Wei Wu, Junjie Yan, Wanli Ouyang. “Improving One-Shot NAS By Suppressing The Posterior Fading”Preprint

Posterior Convergent NAS

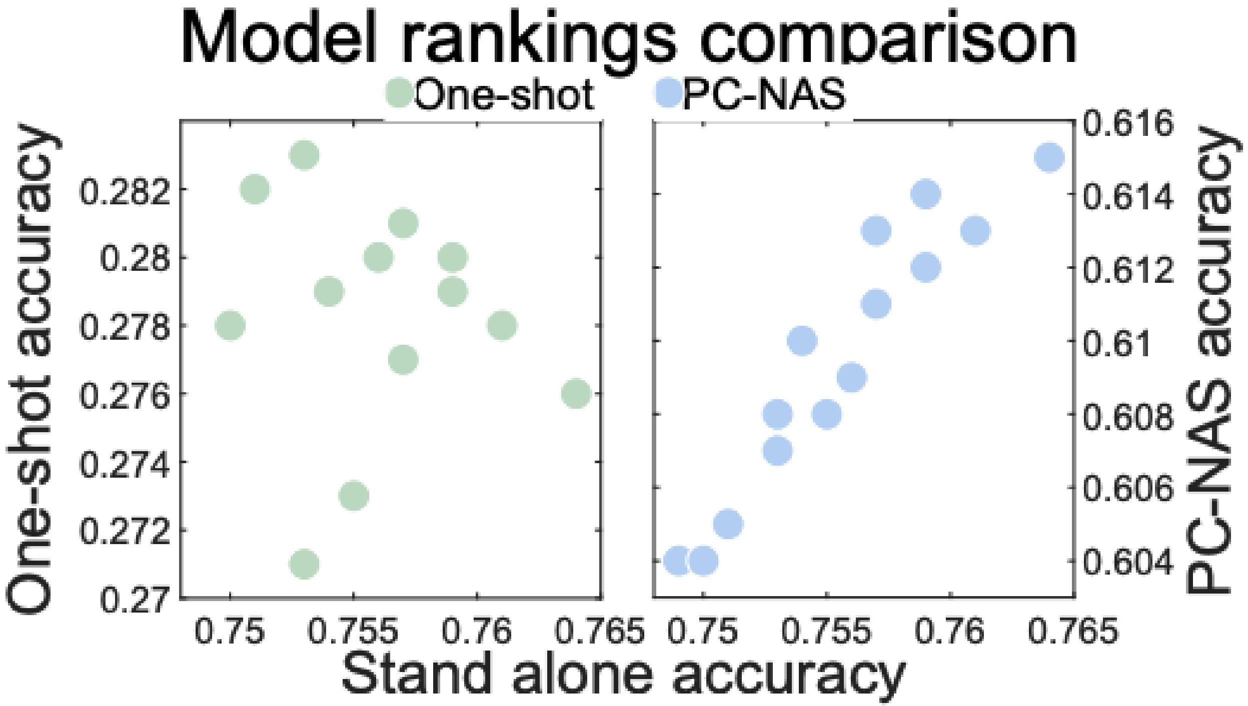
- posterior convergence with/without
  - Left(without):
    - Progressively updating a partial model pool
    - No space shrinking and finetuning
  - Right(with):
    - The proposed method



Top models among final candidate is selected

Posterior Convergent NAS

- posterior convergence with/without
  - Left(without):
    - Progressively updating a partial model pool
    - No space shrinking and finetuning
  - Right(with):
    - The proposed method



Top models among final candidate is selected

Computation Reallocation for Object Detection

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

Preprint

Computation Reallocation for Object Detection

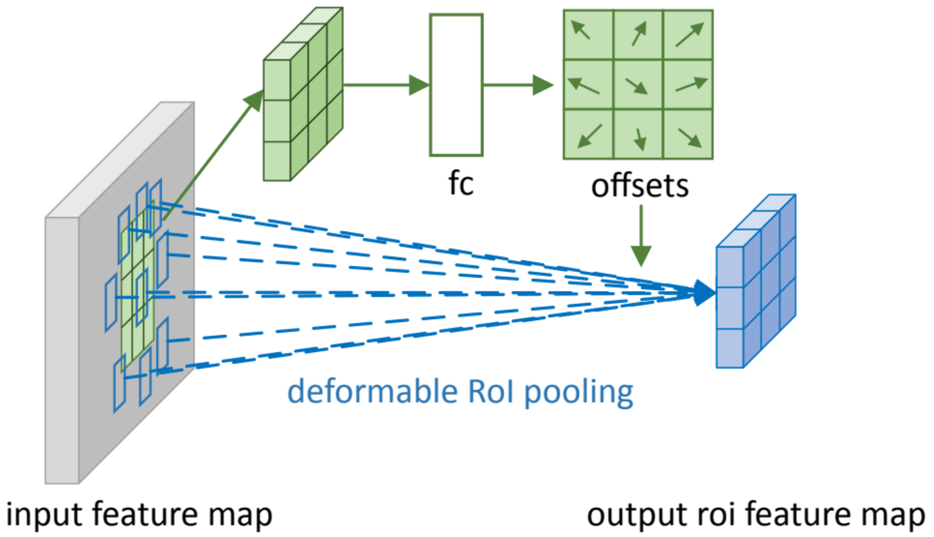
- Many blocks (computation) each stage is predefined in early work on searching detection backbone.

Stage	Block	Large (40 blocks)		Small (20 blocks)	
		c_1	n_1	c_2	n_2
0	Conv3×3-BN-ReLU	48	1	16	1
1	ShuffleNetv2 block (search)	96	8	64	4
2	ShuffleNetv2 block (search)	240	8	160	4
3	ShuffleNetv2 block (search)	480	16	320	8
4	ShuffleNetv2 block (search)	960	8	640	4

Previous work “DetNAS: Backbone Search for Object Detection”use a fixed allocation with is common in NAS for classification.

Computation Reallocation for Object Detection

- The spatial computation allocation strategy has been explored as in Dai et al. 2017, Zhu et al. 2019.



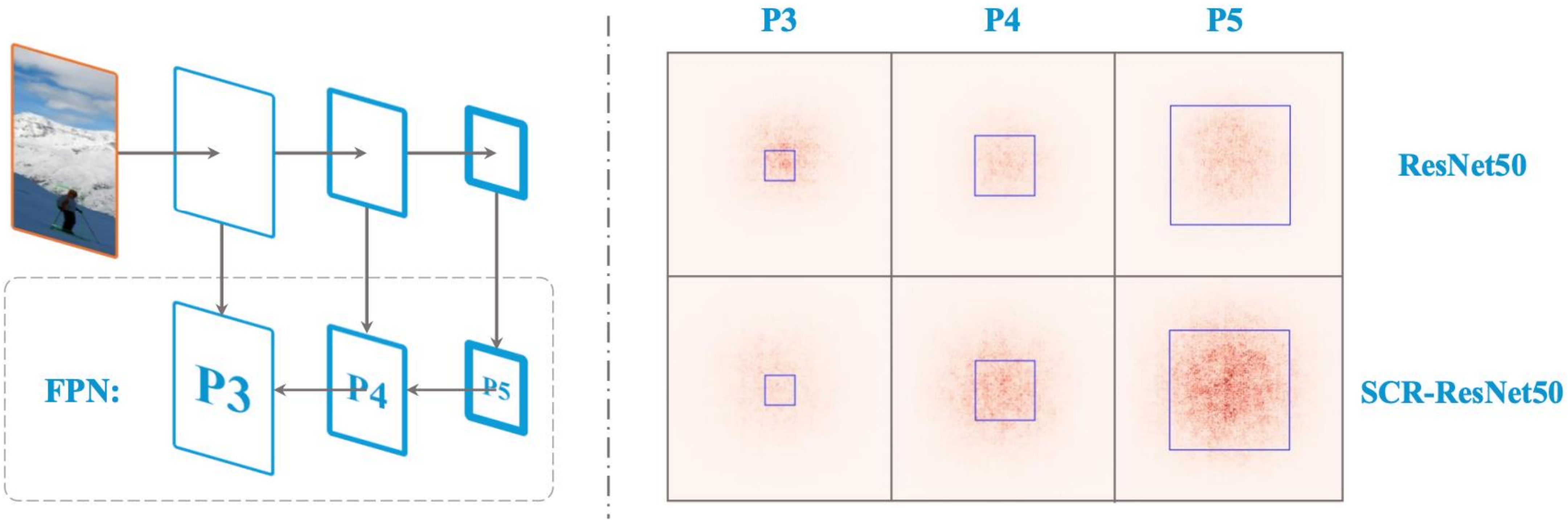
Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. "Deformable convolutional networks". ICCV17

Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. "Deformable convnets v2: More deformable, better results". CVPR19

Computation Reallocation for Object Detection

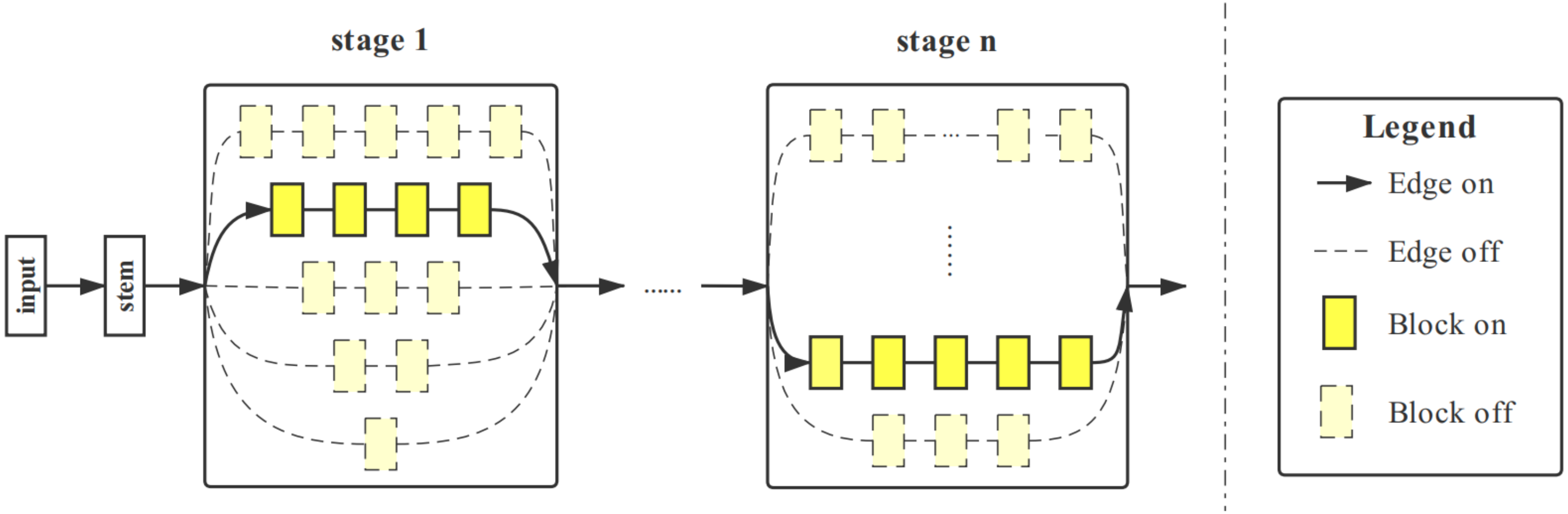
- We argue that these two type of computation allocation is the determining factor of Effective Receptive Fields thus crucial to object detector.
- We propose to search the computation allocation directly on detection tasks to improve the backbone.
- Our Computation Reallocation NAS could be adopted as a plugin to improve the performance of various networks

Computation Reallocation for Object Detection



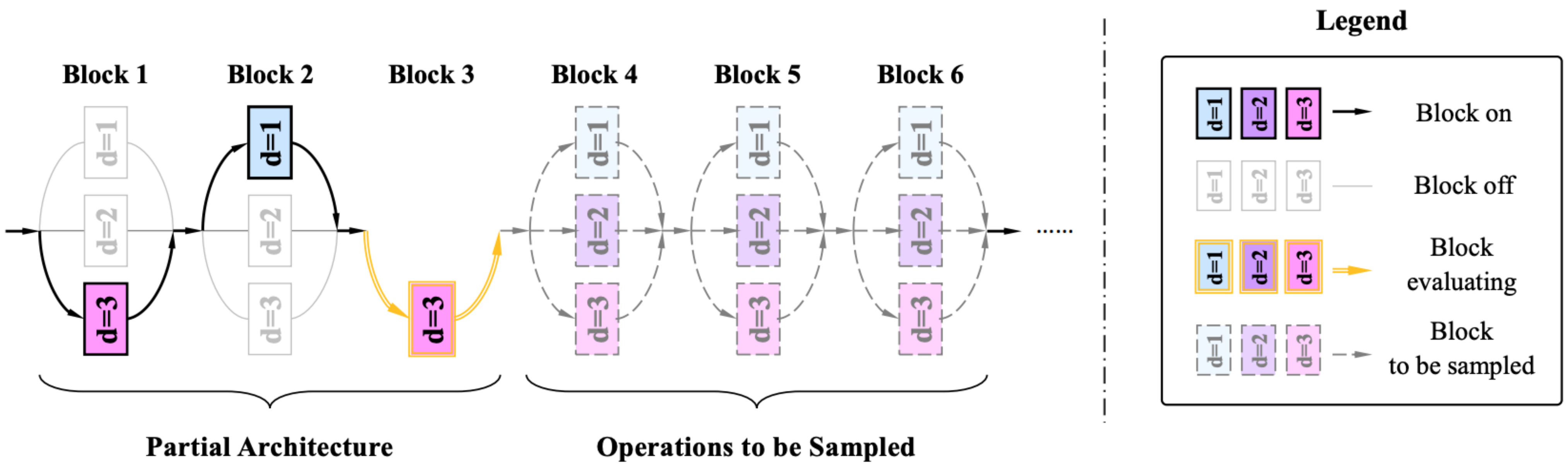
Computation Reallocation for Object Detection

- The Stage Reallocation Space:
  - Different path has different number of block.
  - Looking for the right amount of computation in a stage.
  - For reallocation, we require the total number of blocks remain the same.



Computation Reallocation for Object Detection

- The Spatial Reallocation Space
  - We conduct spatial reallocation by choosing the right dilation.



Computation Reallocation for Object Detection

- Hierarchical Search
  - Stage reallocation space
    - One-shot share parameter
    - Full validation set evaluation
  - Spatial reallocation space
    - One-shot share parameter
    - Greedy search strategy

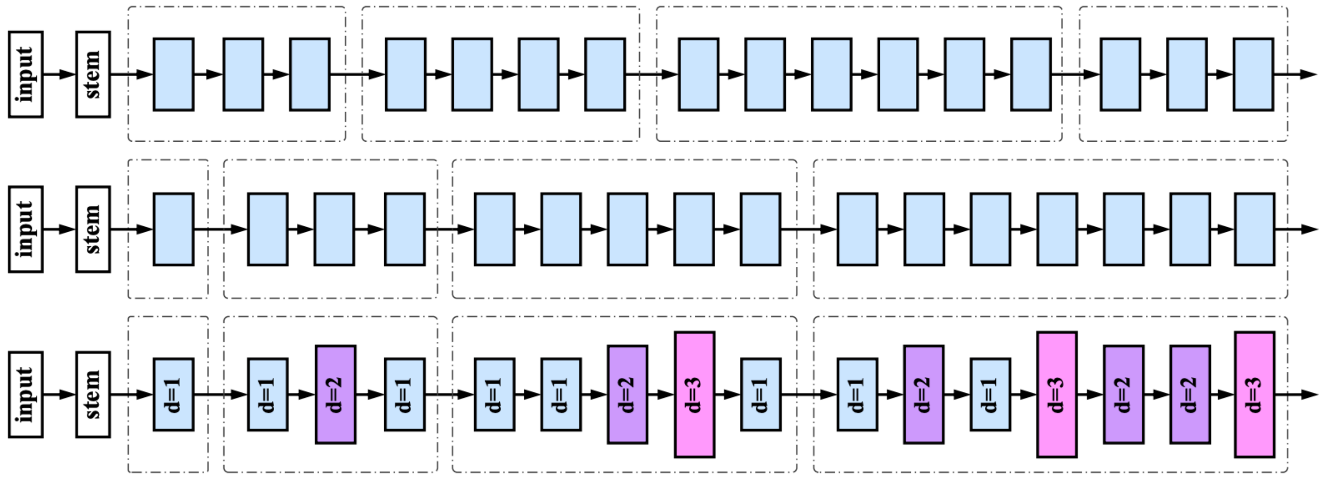
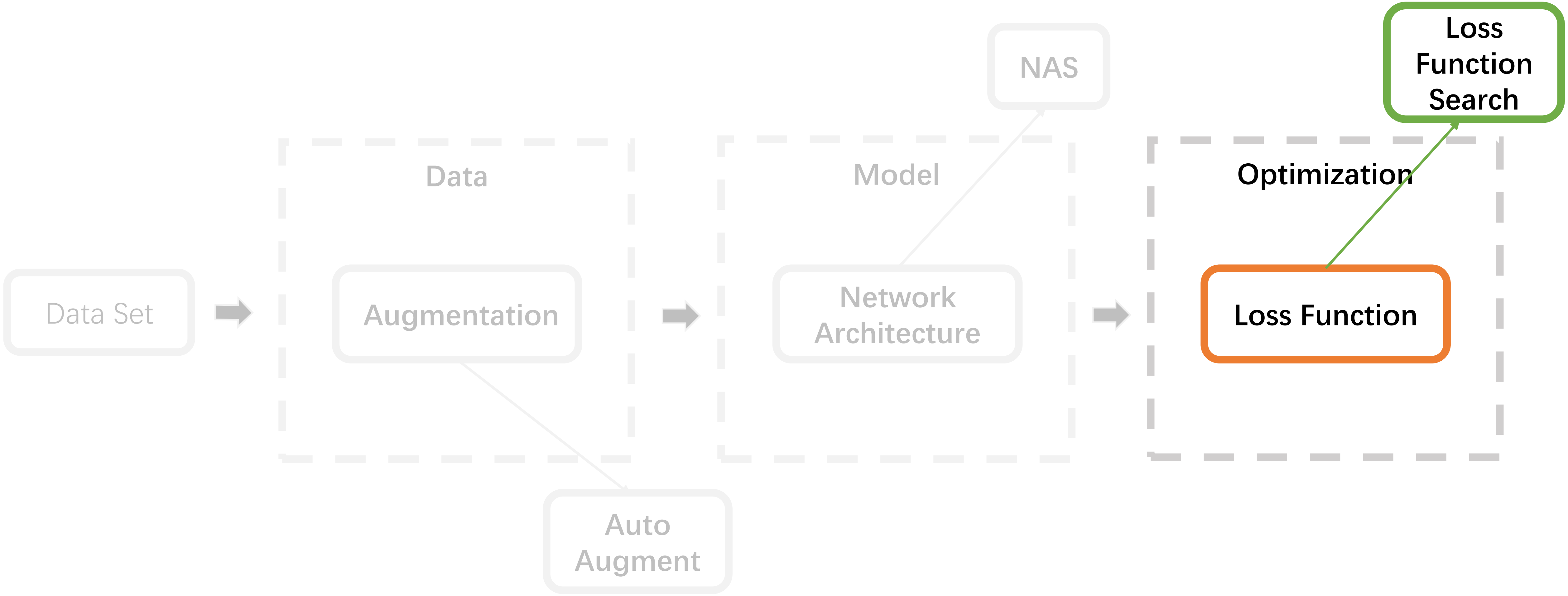


Figure 4: Architecture sketches. From top to bottom, they are baseline ResNet50, stage reallocation SCR-ResNet50 and final CR-ResNet50.

Computation Reallocation for Object Detection

Backbone	FLOPs (G)	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
MobileNetV2	121.1	32.2	54.0	33.6	18.1	34.9	42.1
CR-MobileNetV2	121.4	**33.9**	**56.2**	**35.6**	**19.7**	**36.8**	**44.8**
ResNet18	147.7	32.1	53.5	33.7	17.4	34.6	41.9
CR-ResNet18	147.6	**33.8**	**55.8**	**35.4**	**18.2**	**36.2**	**45.8**
ResNet50	192.5	36.4	58.6	38.7	21.8	39.7	47.2
CR-ResNet50	192.7	**38.3**	**61.1**	**40.9**	**21.8**	**41.6**	**50.7**
ResNet101	257.3	38.6	60.7	41.7	**22.8**	42.8	49.6
CR-ResNet101	257.5	**40.2**	**62.7**	**43.0**	22.7	**43.9**	**54.2**

Deep Learning Assembly Line



AM-LFS: AutoML for Loss Function Search

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

ICCV 2019

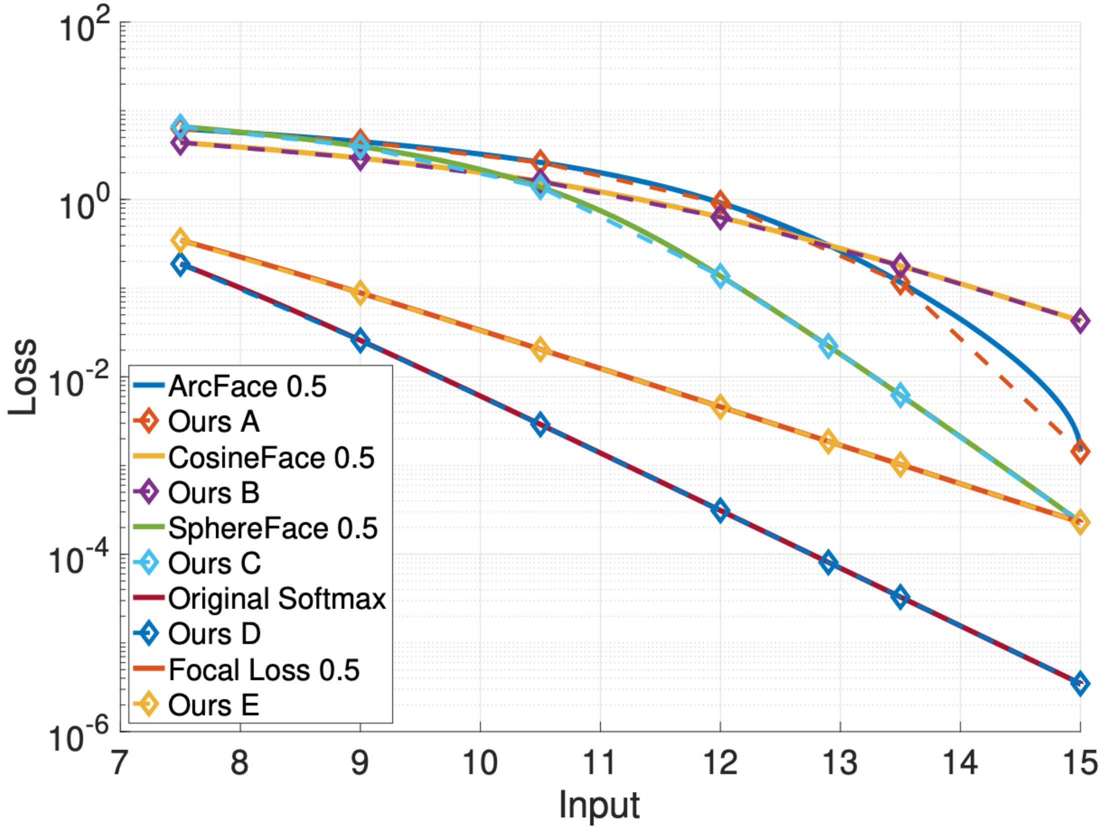
Motivation

- Designing an effective loss function plays an important role in visual analysis.
- Most existing loss function designs rely on hand-crafted heuristics that require domain experts to explore the large design space, which is usually suboptimal and time-consuming.
- Using different loss function in the training stage had been observed effective under certain condition e.g. Curriculum learning

Li, Chuming, Chen Lin, Minghao Guo, Wei Wu, Wanli Ouyang, and Junjie Yan. "AM-LFS: AutoML for Loss Function Search." *ICCV 2019*.

AM-LFS: AutoML for Loss Function Search

- Large portion of hand-crafted loss in different computer vision tasks could be approximated in simple function space



Motivation

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

- Loss in classification task
  - Uniform expression:

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

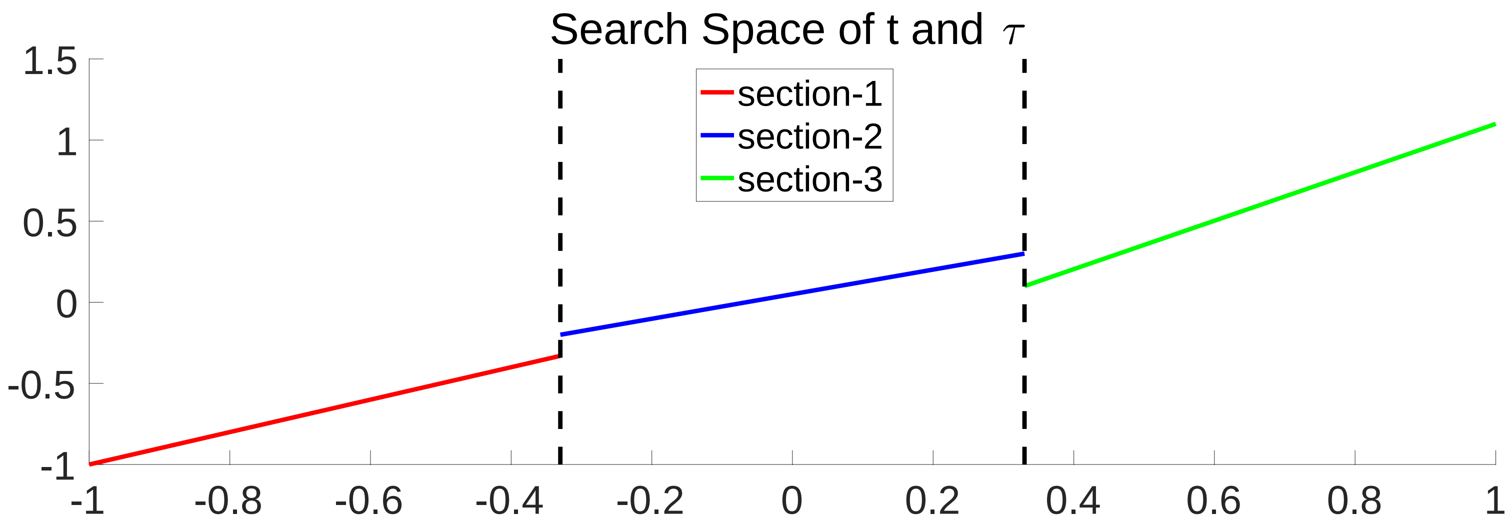
Loss Function	$\tau(x)$
FocalLoss	$x^{(1-x)^m}$

Unified expression of Loss

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

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

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



Unified expression of Loss - continue

- We use independent Gaussian distributions to model $\boldsymbol{\tau}$ and $\boldsymbol{t}$, optimize its mean or even variance.
- We discovered that the same OHL framework works well on optimizing these parameters.
- Here is the convergence of these parameters.

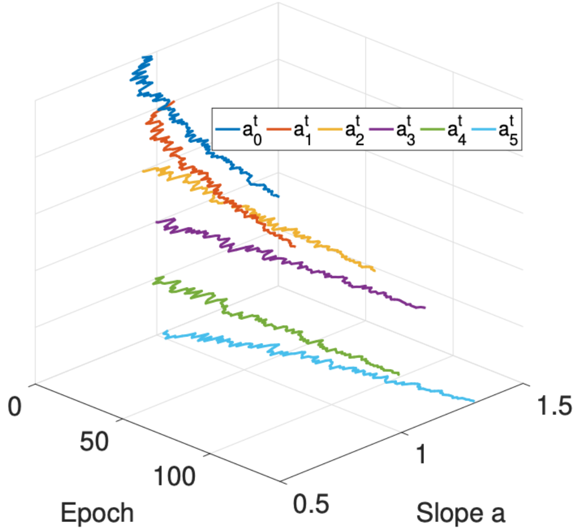


Figure 3. Convergency analysis of AM-LFS.

Experimental results

- Results on person-reID
- Dataset: DukeMTMC-reID

Methods	mAP	Top 1 Acc
SFT	73.2	86.9
MGN	78.4	88.7
MGN(RK)	88.6	90.9
SFT+ours	73.8(+0.6)	87.0
MGN+ours	80.0(+1.6)	89.9
MGN(RK)+ours	90.1(+1.5)	92.4

- 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

Future Work

AutoML +

Data

RunTime

System

AutoML V.S. Arts