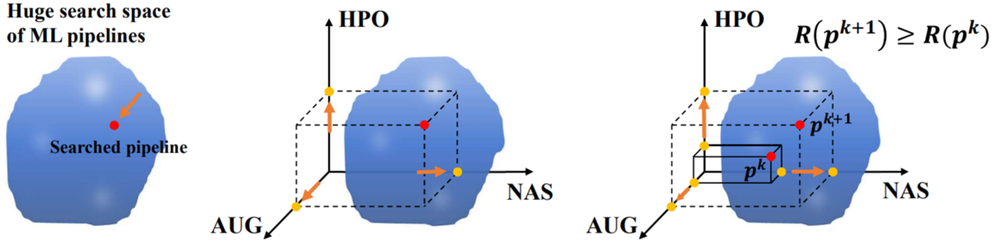
# Multi-Agent Automated Machine Learning

Zhaozhi Wang[123], Kefan Su[1], Jian Zhang[4], Huizhu Jia[1], Qixiang Ye[23], Xiaodong Xie[1], Zongqing Lu[1]

[1]Peking University [2]Peng Cheng Lab [3]University of Chinese Academy of Sciences [4]Huawei

# Introduction

- **Challenge:** hard to solve joint optimization of ML modules (huge search spaces)
- **Motivation:** among AutoML modules there exists a cooperative relationship that facilities the joint optimization of modules
- **Modeling:** MA2ML takes each module as an agent to formulate a multi-agent reinforcement learning problem

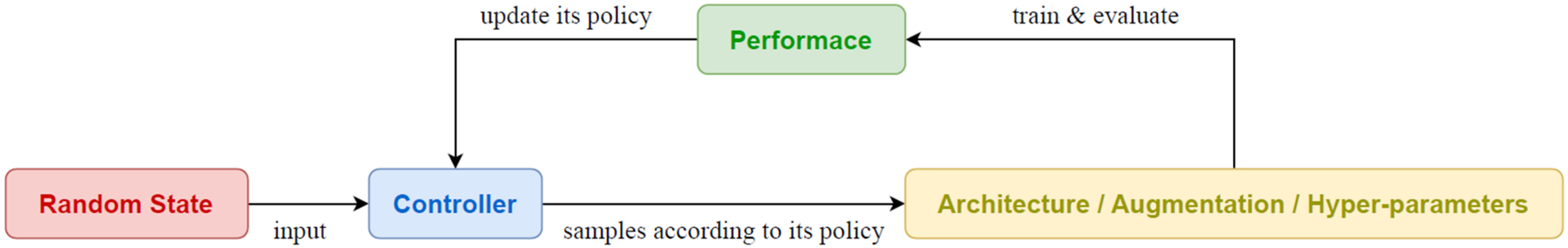



Search spaces of machine learning pipelines. *Left*: single agent controls all modules, and the huge search space makes it ineffective to learn. *Mid*: each agent controls one module, and the learning difficulty is reduced by introducing MA2ML. *Right*: MA2ML guarantees monotonic improvement of the searched pipeline, where $p^k$ and $R(p^k)$ denote the $k$-th searched pipeline and its expected performance, respectively. (NAS: neural architecture search; AUG: data augmentation; HPO: hyper-parameter optimization.)

# Methods

- **Search spaces**
  - **AUG:** AutoAugment
  - **NAS:** NASNet / FBNetV3 (on different datasets)
  - **HPO:** self-designed search spaces (learning rate, weight decay…)

- **Module Agent**




$$A_i \sim \pi_{\theta_i}(\cdot\,|S_i)$$

# Methods

- **Joint optimization**

$$J(\Theta) = \mathbb{E}_{\boldsymbol{\pi_\Theta}(\boldsymbol{A}|\boldsymbol{S})}[R] \quad \text{where } \boldsymbol{\pi_\Theta}(\boldsymbol{A}|\boldsymbol{S}) \triangleq \prod_{i=1}^{n} \pi_{\theta_i}(A_i|S_i)$$

- **MA2ML-Lite**

$$\nabla_{\theta_i} J(\Theta) = \mathbb{E}_{\pi(A|S)}\left[\nabla_{\theta_i} \log \pi_{\theta_i}(A_i|S_i)\,(R - b)\right]$$

- **Credit assignment**

$$b(S, A_{-i}) = \mathbb{E}_{A_i \sim \pi_{\theta_i}}[Q(S, A_i, A_{-i})] \quad \text{(counterfactual baseline)}$$

- **Off-policy learning**

$$J(\Theta) = \mathbb{E}_{\boldsymbol{\pi}(\boldsymbol{A}|\boldsymbol{S})}[Q(\boldsymbol{S}, \boldsymbol{A}) - \lambda D_{KL}(\boldsymbol{\pi}(\cdot\,|\boldsymbol{S})||\boldsymbol{\rho}(\cdot\,|\boldsymbol{S}))] \quad (\boldsymbol{\rho} \triangleq \prod_{i=1}^{n} \rho_i \text{ denotes the joint target policy})$$

# Methods

## Framework

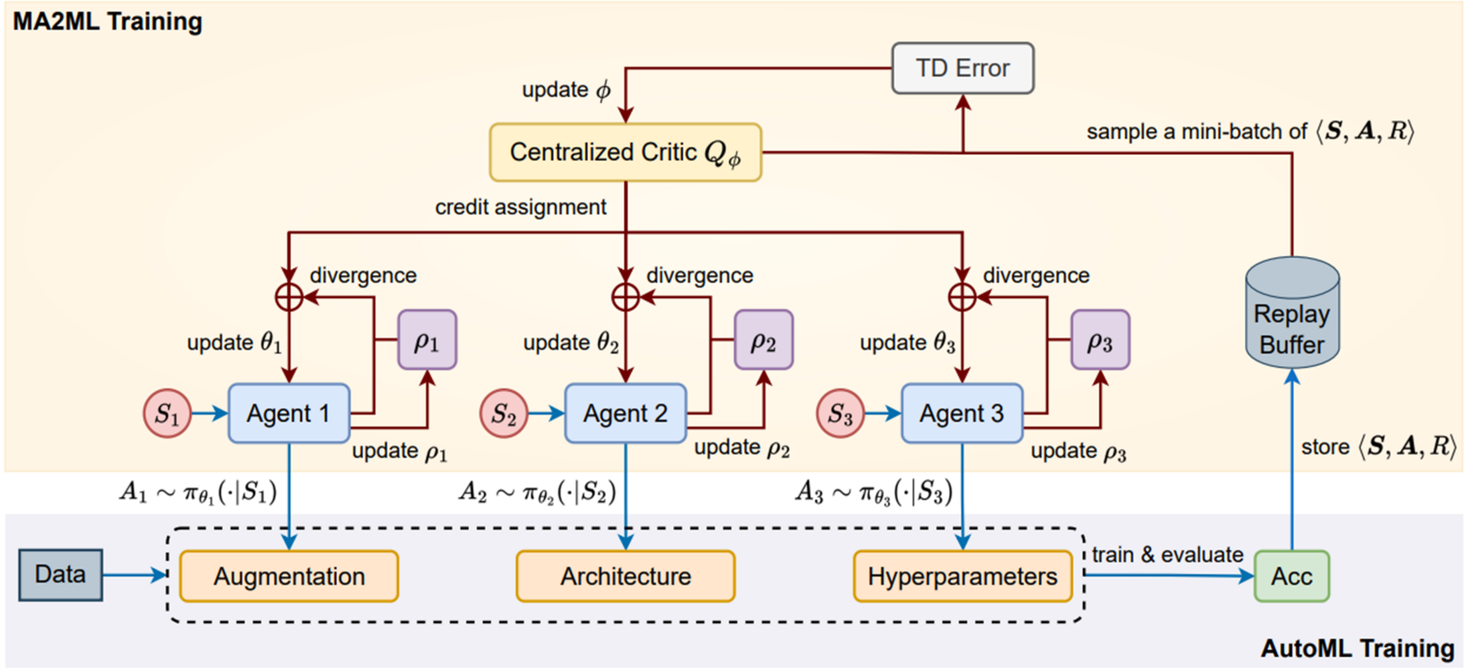



Figure 2. The framework of joint optimization with MA2ML. For AutoML training (lower panel), the ML pipeline is formed by the actions sampled from the policies of agents, then it is deployed for training on the dataset and to obtain the accuracy (reward). After that, the tuple $\langle \boldsymbol{S}, \boldsymbol{A}, R \rangle$ is stored in the replay buffer. For MA2ML training (upper panel), a mini-batch of $\langle \boldsymbol{S}, \boldsymbol{A}, R \rangle$ are sampled from the replay buffer to update the critic, policies, and target policies.

# Experiments

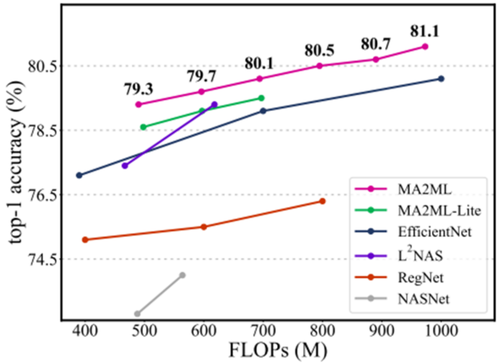
## ImageNet

$$R = Acc \times \left[\frac{FLOPs}{FLOPs\_constraint}\right]^{w}$$ (reward with FLOPs contraint)

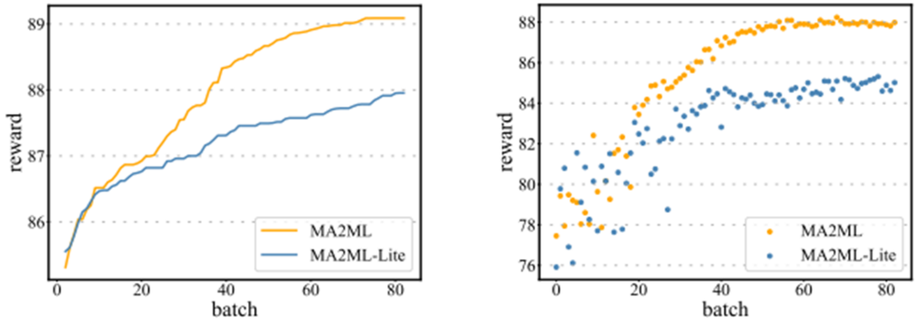
Top-1 accuracy (%) and FLOPs of state-of-the-art AutoML methods on ImageNet, where NARS denotes neural architecture-recipe search. All compared models have computational cost close to 600M FLOPs for a fair comparison.

| Model | Acc (%) | FLOPs (M) | Method | Search Modules |
|---|---|---|---|---|
| DARTS [22] | 73.3 | 574 | gradient | NAS |
| NASNet [54] | 74.0 | 564 | RL | NAS |
| MiLeNAS [11] | 75.3 | 584 | gradient | NAS |
| RMI-NAS [50] | 75.3 | 657 | Random Forest | NAS |
| RegNetY [28] | 75.5 | 600 | pop. param.* | NAS |
| ROME [39] | 75.5 | 556 | gradient | NAS |
| AmoebaNet-C [29] | 75.7 | 570 | evolution | NAS |
| PC-DARTS [43] | 75.8 | 597 | gradient | NAS |
| BaLeNAS [48] | 75.8 | 597 | gradient | NAS |
| ISTA-NAS [44] | 76.0 | 638 | gradient | NAS |
| Shapley-NAS [42] | 76.1 | 582 | gradient | NAS |
| DAAS [40] | 76.6 | 698 | gradient | AUG+NAS |
| DHA [52] | 77.4 | - | gradient | AUG+NAS+HPO |
| MIGO-NAS [51] | 78.3 | 595 | MIGO | NAS |
| OFA† [2] | 79.0 | 595 | gradient | NAS |
| EfficientNet-B1 [37] | 79.1 | 700 | RL | NAS |
| FBNetV3‡ [5] | 79.2 | 550 | NARS | NAS+HPO |
| $L^2$NAS [25] | 79.3 | 618 | RL | NAS |
| **MA2ML-A** | 79.3 | 490 | MARL | AUG+NAS+HPO |
| **MA2ML-B** | **79.7** | 596 | MARL | AUG+NAS+HPO |

*Population parameterization. †Results are given in [25] without distillation.
‡Results are reproduced according to 600M FLOPs constraint without distillation.




Comparison of MA2ML with other AutoML methods on ImageNet.



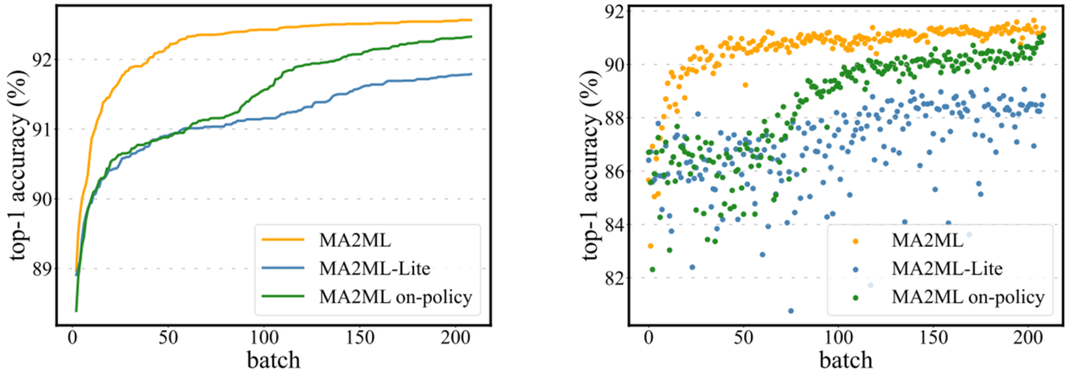

Learning patterns of MA2ML and MA2ML-Lite on ImageNet-200 (FLOPs_constraint = 600M). *Left:* the average reward curves of top-20 pipelines in terms of patch numbers. *Right:* the scatter plot for average rewards of pipelines in each batch.

# Experiments

## ● CIFAR-10/100

Top-1 accuracy (%) and parameter size (MB) of compared AutoML methods on CIFAR-10 and CIFAR-100. The results of MA2ML and MA2ML-Lite are mean and standard deviation of 5 trials of the best ML pipeline.

| Model | CIFAR-10 | | CIFAR-100 | | Method |
|---|---|---|---|---|---|
| | Acc (%) | Param (M) | Acc (%) | Param (M) | |
| NASNet-A [54] | 97.60 | 27.6 | - | - | RL |
| ENAS [27] | 97.11 | 4.6 | 80.57 | 4.6 | RL |
| $L^2$NAS [25] | 97.51±0.12 | 3.8 | 82.24±0.19 | 3.5 | RL |
| AmoebaNet [29] | 97.45±0.05 | 2.8 | 81.07 | 3.1 | evolution |
| DARTS [22] | 97.24±0.09 | 3.3 | 82.64±0.44 | 3.3 | gradient |
| DARTS- [3] | 97.41±0.08 | 3.5 | 82.49±0.25 | 3.3 | gradient |
| MiLeNAS [11] | 97.49±0.11 | 3.9 | - | - | gradient |
| ISTA-NAS [44] | 97.64±0.06 | 3.4 | 83.10±0.11 | - | gradient |
| AutoHAS [6] | 95.0 | - | 78.4 | - | RL |
| Joint Search [17] | 97.46±0.09 | - | 83.81±0.49 | - | gradient |
| DAAS [40] | 97.76±0.10 | 4.0 | 84.63±0.31 | 3.8 | gradient |
| DHA [52] | 98.11±0.26 | - | 83.93±0.23 | - | gradient |
| MA2ML-Lite | 97.70±0.10 | 7.8 | 84.80±0.12 | 9.0 | MARL |
| **MA2ML** | 97.77±0.07 | 9.0 | 85.08±0.14 | 7.7 | MARL |




Learning patterns of MA2ML, MA2ML on-policy, and MA2ML-Lite on CIFAR-10. *Left*: average accuracy curve of top-30 pipelines in terms of the number of batches. *Right*: the scatter plot for the average accuracy of different pipelines in each batch. MA2ML outperforms MA2ML on-policy and MA2ML-Lite consistently in terms of accuracy and sample efficiency.

# Conclusion



- **MA2ML**
  - a general framework for the joint optimization of ML modules
  - transforms the joint optimization of modules as an MARL problem
  - yields state-of-the-art top-1 accuracy on ImageNet under several constraints of computational cost

- **Advantages**
  - can be extended to more modules
  - agnostic to search spaces
  - can be generalized to different tasks
  - directly optimizes the reward