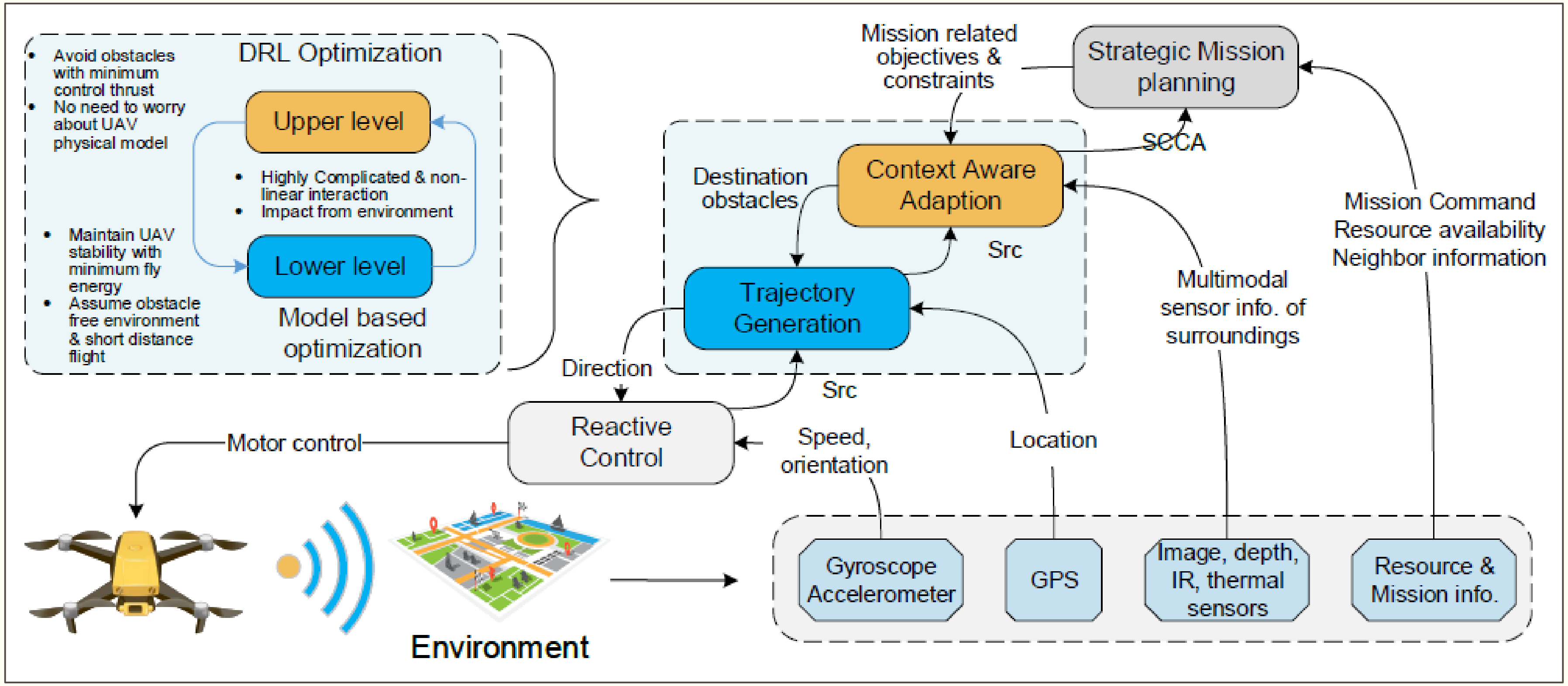
NEURAL NETWORK PRUNING AND FAST TRAINING FOR DRL-BASED UAV TRAJECTORY PLANNING

Yilan Li, Haowen Fang, Mingyang Li, Yue Ma, Qinru Qiu
Department of Electrical Engineering & Computer Science
Syracuse University

Introduction

- Autonomous trajectory planning for unmanned aerial vehicles(UAVs) requires on-board embedded system as well as real-time computation
- This has been considered as optimization problem and Deep Reinforcement Learning(DRL) has been applied to solved it



Motivation

- The payload capacity of small UAVs imposes stringent constraints on the size/weight and energy dissipation of the onboard computing system
    - the computing capabilities vs the real-time computation
    - faster embedded processors
    - more efficient computing models
- Most of the existing pruning works apply pruning on fully trained model
    - three-step process, i.e., training, pruning, and re-training, has high computation and memory complexity.
    - DRL training is more time consuming

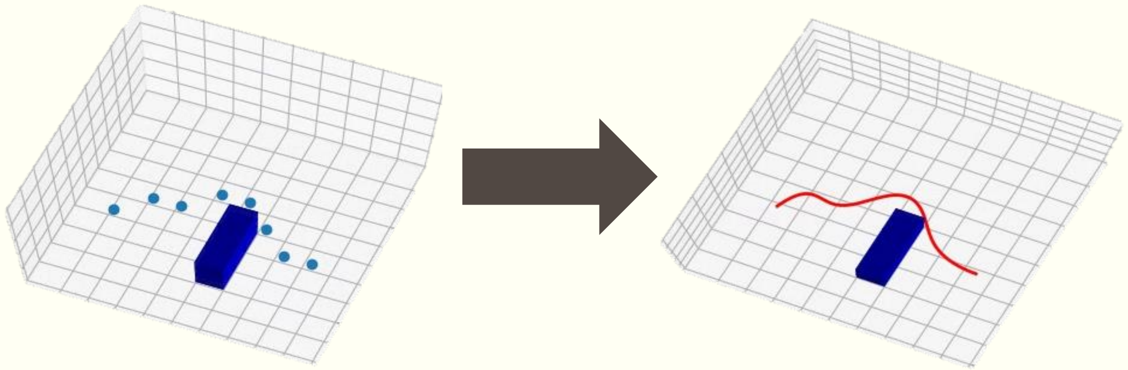
Contribution

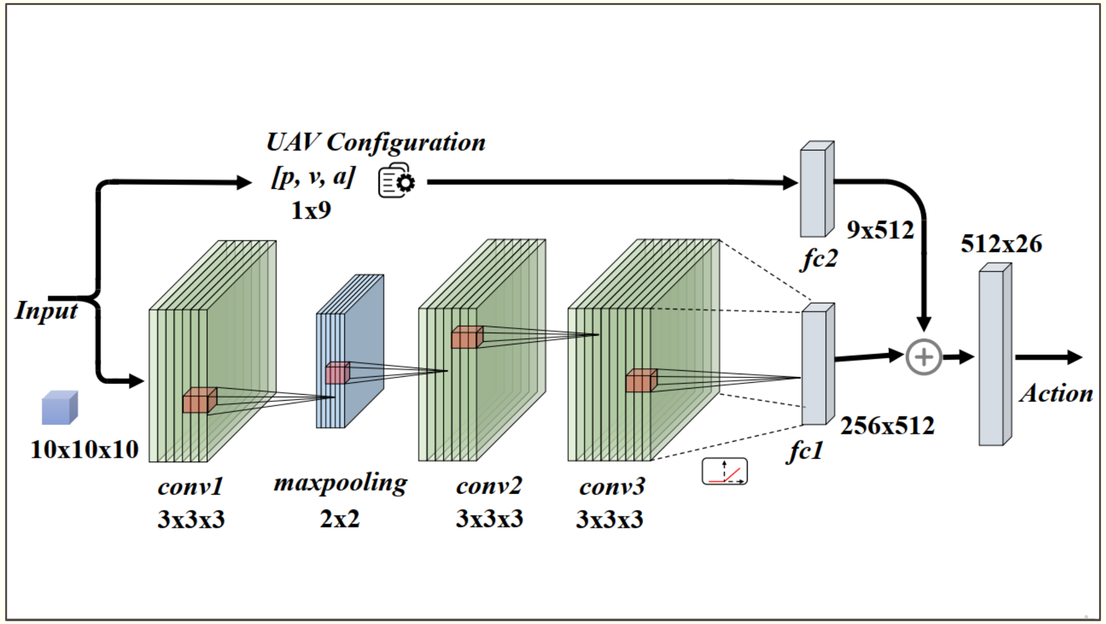
- Improve the DRL training of drone trajectory planning for model compression
  - A DRL model that generates an energy-efficient collision-free trajectory
  - A new reward function and stochastic action selection technology are proposed to improve the DRL training convergence
  - A framework that integrates Alternating Direction Method of Multipliers (ADMM) based structured weight pruning and DRL training.
  - The optimized layer wise compression ratio is studied as a general guideline for structured weight pruning for the deep Q-network.

Trajectory planning for Multi-rotor UAVs

- A two-level optimization framework
  - ❑The upper level DRL model generate a sequences of waypoints
  - ❑The lower level applies non-linear optimization to generate a smooth trajectory
- Input:
  - ❑10x10x10 voxel representation of the environment
  - ❑UAV status consisting of 3-dimensional position, velocity and acceleration
- Output:
  - ❑Probability of 26 possible next waypoint actions

the upper level the lower level





Improving DRL convergence speed – Fast DRL

- Redefine reward function G as a fusion of a navigation reward $\mathcal{N}_r$ and a navigation effort $\mathcal{N}_e$.
- The Nr is proportional to the probability that the current UAV navigation direction θ is the optimal direction $\hat{\theta}$:

$$\mathcal{N}_r = f(\theta; \hat{\theta}, \Sigma, a, b) \\ = \frac{\phi((\theta-\mu),\Sigma)}{\Phi((b-\mu),\Sigma)-\Phi((a-\mu),\Sigma)} \\ a \le \theta \le b$$

$$\mu = \hat{\theta} = (\hat{\theta}_1, \hat{\theta}_2) \\ a = \hat{\theta} - \frac{\pi}{2}, b = \hat{\theta} + \frac{\pi}{2}$$

θ is the current UAV navigation direction

$\phi()$ is the probability density function of normal distribution $\mathcal{N}(\hat{\theta}, \Sigma)$,

$\Phi()$ is the cumulative distribution of $\mathcal{N}(\hat{\theta}, \Sigma)$

θ and $\hat{\theta}$ are both two-dimensional vectors consisting of polar angle $\hat{\theta}_1$ and azimuthal angle $\hat{\theta}_2$

a and b are lower and upper bounds of random variables

Improving DRL convergence speed

Then the reward G is defined as:

$$G(\theta) = [1-g \quad g]\begin{bmatrix}\mathcal{N}_r(\theta)\\ \mathcal{N}_e(\theta)\end{bmatrix}\ s.t.\ g \in [0, \quad 1]$$

where g as the gain of fusion, and a navigation effort $\mathcal{N}_e$.

The variance can be obtained as:

$$Var\big(G(\theta)\big) = (1-g^2)Var\big(\mathcal{N}_e(\theta)\big) + g^2 Var(\mathcal{N}_r(\theta))$$

Finally, we can get g as:

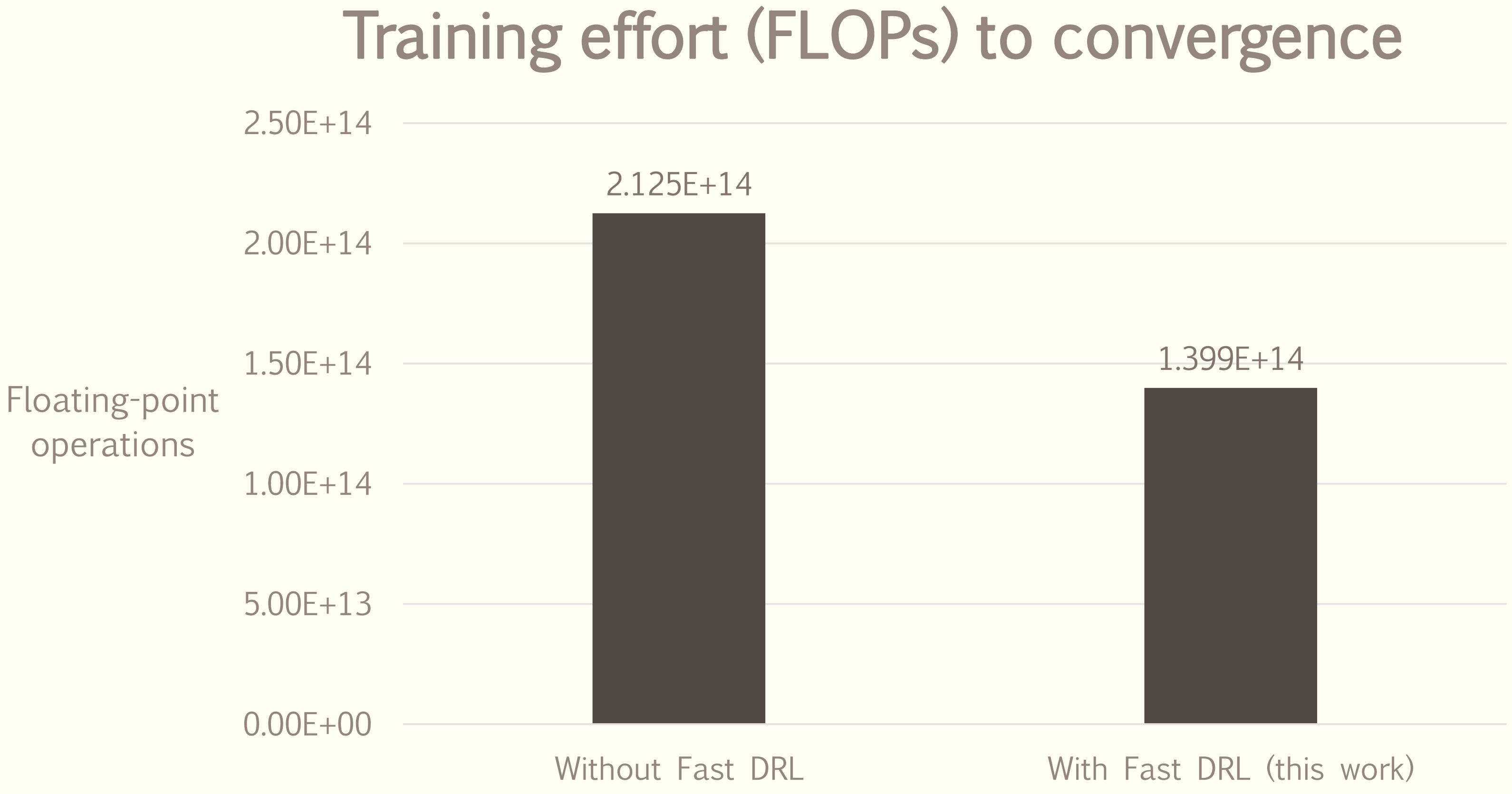
$$g = \frac{tr[Cov(\mathcal{N}_e(\theta))]}{tr\big[Cov\big(\mathcal{N}_r(\theta)\big)\big] + tr[Cov(\mathcal{N}_e(\theta))]}$$

where tr represents trace of the covariance matrix

- Stochastic action exploration during DRL training
  - When UAV fails it mission, the model rollback to last step, stochastically select an action
  - Furthermore, the model will randomly choose the action of the top M actions with the highest value

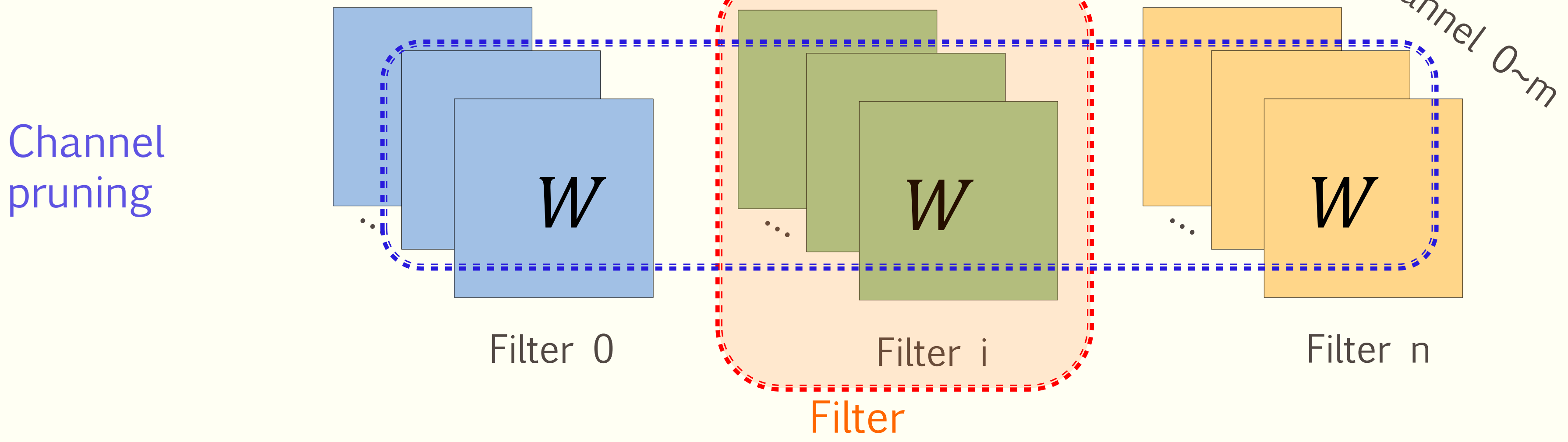
Experimental Results I

- Impact of convergence speed by applying proposed fast DRL.
  - The training effort needed of a fully trained model reduces 34.14%



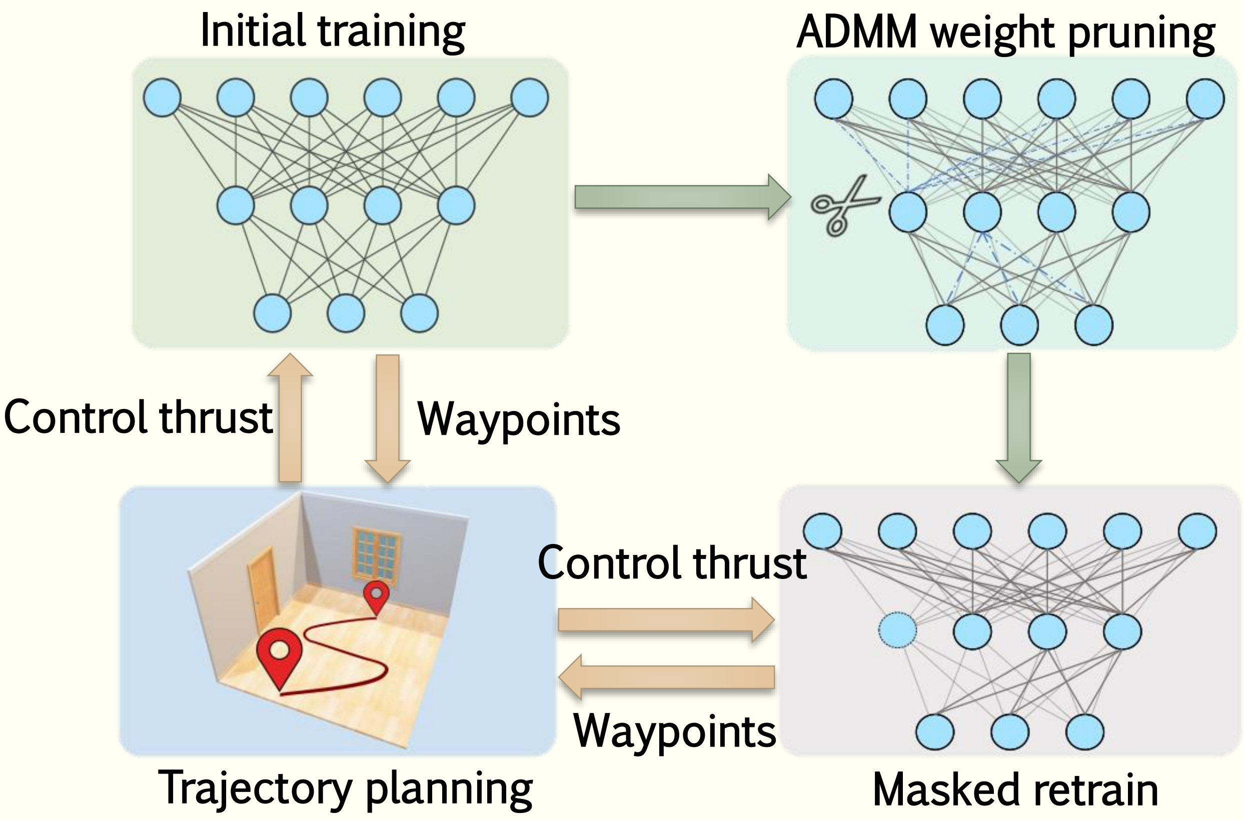
Neural network pruning in deep learning

- ADMM is widely used in structured weight compression for deep neural networks(DNNs)
- We adopt ADMM to prune DRL model to reduce time complexity
- Structured weight pruning strategies:
  - Filter pruning
  - Channel pruning
  - Column pruning
- Filter pruning and channel pruning are more accessible and hardware friendly



Overall flow of pruning framework

- Apply structured weight compression at early iterations of DRL training of the original model



Early phase integrated weight compression

- The goal of weight pruning can be defined as:

$$\min_{(W,b)} f(W,b)$$
$$s.t.\ W = \{w_i\}_{i=0}^{L-1},\ \ b = \{b_i\}_{i=0}^{L-1}$$
$$w_i \in s_i, \qquad i = 0, \dots, L-1$$

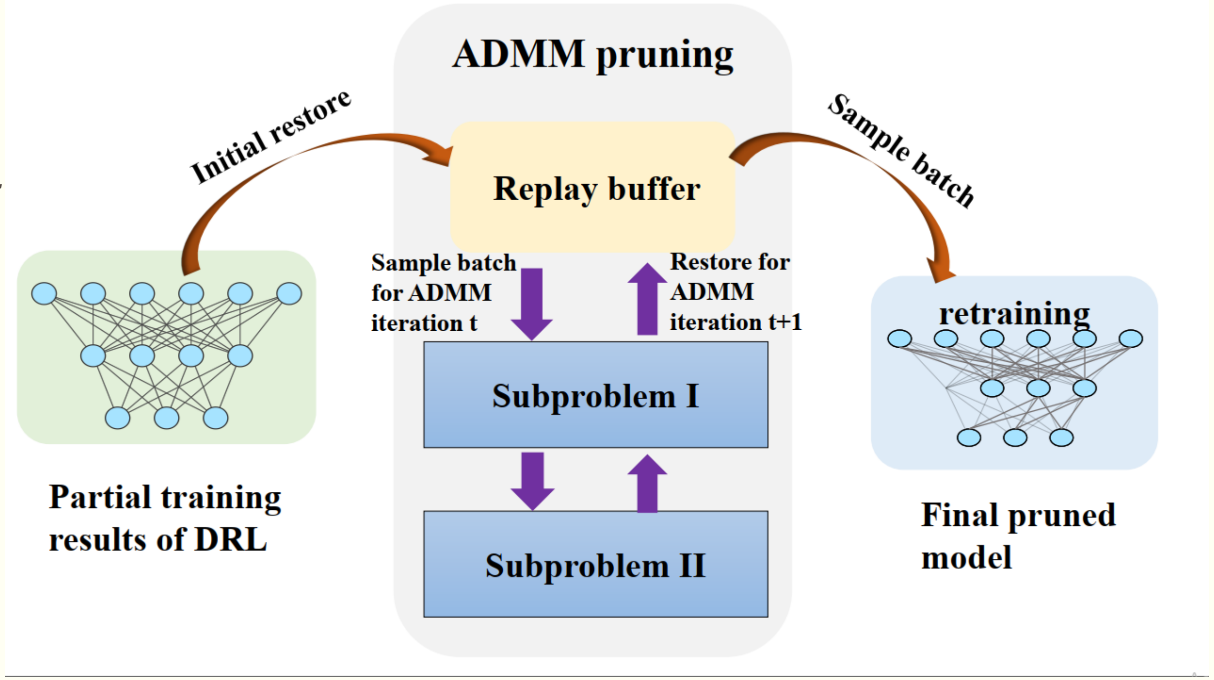
- The objective function is:

$$f(W,b) = \frac{1}{|batch| \cdot \mathcal{A}} \sum_{j \in batch} \max\left(\sum_{a=0}^{\mathcal{A}-1} |q_{aj} - \hat{q}_{aj}|, \sum_{a=0}^{\mathcal{A}-1} (q_{aj} - \hat{q}_{aj})^2\right) + \sum_{i=0}^{L-1} \|w_i\|_F^2$$

- Finally, the optimization problem can be decomposed into two subproblems:

$$\min_{(W,b)} f(W,b) + \sum_{i=0}^{L-1} \frac{\rho_i}{2} \left(\|w_i - z_i + \mu_i\|_F^2 + \lambda \|b_i\|_F^2\right)$$
$$\min_{z_i} \sum_{i=0}^{L-1} f_i(z_i) + \sum_{i=0}^{L-1} \frac{\rho_i}{2} \left(\|w_i - z_i + \mu_i\|_F^2 + \lambda \|b_i\|_F^2\right)$$

- Each time after the ADMM prune converges, a pruned model will be simulated, and the replay buffer will be updated



An iterative prune-restore procedure

Experimental Results II

- Comparing different layer-wise pruning combinations
  - All the experiments are built on top of the same partially trained initial model with success rate of 87.5%
  - The original model achieves 96.8% success rate.
  - The system increases the success rate to 97.3% with 70% weight reduction in conv2 layer and 82% reduction in conv3 layer increases success rate to 97.3%.
  - With a marginal success rate loss of 1.8% compared with unpruned model, a total sparsity of 73.44% is achieved, translating into 3.764x weight pruning.

Structured pruning	Pruned layers					Prune rate	Success rate
	conv1	conv2	conv3	fc1	fc2		
Filter	-	-	-	-	-	-	96.80%
Filter	-	-	50%	50%	-	1.460x	95.60%
Channel	-	-	50%	50%	-		
Filter	50%	50%	50%	50%	50%	2.507x	95.00%
Channel	-	-	50%	50%	-		
Filter	-	70%	82%	-	-	3.045x	97.30%
Filter	10%	70%	80%	80%	50%	3.764x	95.00%

Experimental Results III

- How the selection of the initial model impact the training cost and success rate
  - All the four cases has the same prune ratio(70% weight reduction in conv2 layer and 82% reduction in conv3 layer), but different success rate when start pruning
  - The best success rate increases to 97.3%, and the total FLOPs for both training and pruning is reduced by 33.33%
  - With a marginal success rate loss of 2.1% compared with best model, we can start pruning when the pretrained model reaches 49.7% success rate. The total FLOPs drop to 3.009e+13 which is a decrease of 79.17%

Success rate of initial model	Success rate after pruning	Pretrain FLOPs		Weight pruning FLOPs	Total training FLOPs
		Conv layers	FC layers		
96.8%	-	1.377e+14	2.182e+12	-	1.399e+14
96.8%	97.1%	1.377e+14	2.182e+12	9.401e+11	1.409e+14
87.5%	97.3%	9.182e+13	1.455e+12	9.401e+11	9.422e+13
65.4%	96.7%	4.591e+13	7.275e+11	9.401e+11	4.758e+13
49.7%	95.2%	2.870e+13	4.549e+11	9.401e+11	3.009e+13

Experimental Results IV

- Comparing the best pruned model with the prior work
  - The unpruned fast DRL
    - success rate of 96.8%, which is higher than the original model.
    - The average number of selected waypoints is 25.56% less than the original model, with a 47.3% FLOPs decrease.
  - The pruned model
    - success rate of 97.3%.
    - The average number of selected waypoints is 23.11% less than the original model, with a 57.18% FLOPs decrease.
    - The inference time decrease from 2.414ms to 1.427ms, having 40.8% reduction

Success rate of initial model	Prune rate	Model sparsity	Success rate	Achieve rate	Average waypoints	Inference FLOPs	Average measured inference time(ms)
prior work	-	0%	95.6%	96%	9.0	1.844e+7	2.454
Un-pruned of our work	-	0%	96.8%	98.0%	6.7	9.716e+6	2.414
Our work	3.045x	67.16%	97.3%	98.6%	6.92	4.160e+6	1.427

Conclusion

- We present an early-phase integrated neural network compression for DRL based trajectory planning system.

- A new reward function with stochastic action exploration helps improving the convergence speed

- An early phase integrated structured weight compression technique is introduced.

- The pruned model not only saves training effort, but also speeds up the inference time

THANK YOU!