A Deep Ensemble Method for Multi-Agent Reinforcement Learning: A Case Study on Air Traffic Control

Supriyo Ghosh\textsuperscript{1}, Sean Laguna\textsuperscript{1}, Shiau Hong Lim\textsuperscript{1}, Laura Wynter\textsuperscript{1} and Hasan Poonawala\textsuperscript{2}

\textsuperscript{1}IBM Research AI, Singapore
\textsuperscript{2}Amazon Web Services (AWS), UK
Motivation

- **Air traffic control (ATC)**
  - Monitor current state of aircrafts and recommend **real-time** decisions.
  - Need to optimize a complex objective function
    - Minimize congestion, conflicts, arrival delay and fuel consumption cost.
  - Heavy traffic volume might lead to (human) operational errors.
  - A **sequential decision-making** problem involving **multiple actors** influencing each other.
Our Contributions

- Modelled ATC problem within a multi-agent reinforcement learning (MARL) framework.

- Solved the MARL problem with a model-based Kernel RL and a model-free Deep RL methods.

- Proposed a general-purpose novel deep ensemble MARL method to combine the power of deep RL and kernel RL.

- Demonstrated the efficacy of ensemble MARL method on a real-world dataset consisting of ~1600 active aircrafts.
Multi-agent Reinforcement Learning (MARL)

- Single Agent RL:
  \[ < S, A, P, R, \gamma > \]
  Learn a policy \( \pi \) to maximize long term reward \( Q^*(s, a) \):
  \[
  \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \big| s_0 = s, a_0 = a \right]
  \]

- Multi-agent RL
  \[ < S, O_1...O_N, A_1...A_N, P, R, \gamma > \]
  Observation: \( o_i : S \rightarrow O_i \)
  Transition: \( P : S \times A_1 \times ...A_N \rightarrow S \)
  Reward: \( r_i : O_i \times A_i \rightarrow \mathbb{R} \)

Centralized learning & decentralized execution in MARL [Gupta et. al., 2017]
**MARL Formulation for ATC**

- **State space**
  - Local features: Aircraft’s location, speed, direction, timeliness
  - Neighborhood features: N nearest aircrafts’ relative velocity & direction
  - Extended feature: Coarse and fine grid image information (for deep RL)

- **Action space (deviate speed by $$\delta$$)**
  $$A_t = \{\max(v_{min}, (v_{t-1} - \delta)), v_{t-1}, \min(v_{max}, (v_{t-1} + \delta))\}$$

- **Reward Function**

\[
\begin{align*}
\hat{r}_t^i &= \alpha \cdot I(o_t^i, R^s) + \beta \cdot I(o_t^i, R^c, N^c) + \gamma \cdot \max(0, \tilde{d}_t^i - d_t^i) + \delta \cdot F(v_t^i - v_0^i) \\
\end{align*}
\]

- **Penalty Weights**
- **Conflicting Radius**
- **Congestion Radius**
- **Expected Schedule**
- **Optimal Velocity**

- **Reward Function**

$$\text{Fuel cost penalty structure}$$
Kernel and Deep MARL for ATC

Model based kernel RL

1. Inputs: \( S^a = \{ s^a_k, r^a_k, \hat{s}^a_k \mid k = 1, \ldots, n_a \} \forall a \in A \)
2. Generate \( m \) representative states with K-means clustering: \( \tilde{S} = \{ \tilde{s}_1, \ldots, \tilde{s}_m \} \)
3. Define Gaussian kernel \( \kappa, \tilde{\kappa} \) using distance from original to representative state
4. Compute \( D^a : d^a_{ij} = \kappa_{\tilde{s}_i}(s^a_i, s^a_j) \)
5. Compute \( K^a : k^a_{ij} = \kappa_{\tilde{s}_i}(s^a_i, \tilde{s}^a_j) \)
6. Compute transition probability: \( P^a = K^a D^a \)
7. Compute reward \( r^a : r^a_i = \sum k^a_{ij} r^a_j \)
8. Solve MDP\( \{ \tilde{S}, A, P^a, r^a, \gamma = .99 \} \) & obtain \( Q^* \)

- **Advantages**
  - Performs well in neighborhood of dense training
  - Strong theoretical bounds

- **Limitations**
  - Extrapolates poorly

Model free deep RL (PPO)

1. Initialize policy network with parameter \( \theta_0 \)
2. For each episode \( k \), store a set of transition samples \( (s^i_t, a^i_t, r^i_t, s^i_{t+1}) \) for each agent \( i \) in buffer \( D \) by simulating policy \( \pi(\theta_k) \)
3. Update \( \theta_k \) with minibatch of transitions from \( D \) for \( M \) rounds to optimize PPO objective:
   \[
   \mathbb{E}_t \left[ \min(r_t(\theta) A_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon) A_t) \right]
   \]
   \( r_t(\theta) \) is ratio between \( \pi_\theta(a_t|s_t), \pi_{\theta_{old}}(a_t|s_t) \)
   \( A_t := R_t - V(s_t) \) is advantage function

- **Advantages**
  - Flexible and generalizes well
  - Can deal with richer state space

- **Limitations**
  - Can be brittle even in dense training neighborhood
Deep Ensemble MARL

- Existing ensemble methods are either not feasible for model-based methods or unable to take multi-agent interactions in account.
- We train a separate deep neural network that efficiently learns to arbitrate between decisions of pre-trained kernel and deep MARL.

Inputs: Kernel model $\tilde{K}$, PPO model $\tilde{\pi}(\tilde{\theta})$

1. Initialize ensemble policy to $\pi(\theta_0)$
2. For each episode $k$, run line 3-7
3. For each time $t$ and agent $i$, sample ensemble action $a^i_t$ for observation $s^i_t$
4. If $a^i_t$ is 0 then get action $\tilde{a}^i_t$ from $\tilde{K}$, otherwise get $\tilde{a}^i_t$ using $\tilde{\pi}(\tilde{\theta})$
5. Execute joint action $\tilde{a}_t = (\tilde{a}_t^1, ..., \tilde{a}_t^N_t)$
6. Store transitions $(s^i_t, a^i_t, r^i_t, s^i_{t+1})$ in $D$
7. For $M$ rounds update $\theta_k$ with minibatch of transitions from $D$
Experimental Settings

- **Datasets (from Southern Europe)**
  - 24-hours schedule for 1600 active flights
  - 300 training days and 30 testing days
  - 3 fuel cost settings considered: low, medium (from Airbus), high

- **Benchmark Approaches**
  - Baseline: simulate default schedule (no penalty for fuel & delay)
  - Local search: Each aircraft chooses a myopic best action
  - DDMARL (Brittain et. al., 2019): Only consider conflict penalty

- **Air traffic simulator**
  - An open-source simulator developed by Eurocontrol
  - We develop a message passing adapter between the simulator and our RL agent
Training Performance

- Training performance of our deep MARL is at par with DDMARL

- Our ensemble MARL has better training performance than both deep MARL and DDMARL across the board
Testing Performance

- Ensemble MARL always outperforms kernel and deep RL.
- Ensemble MARL provides \( \sim 9\% \) gain in reward in a realistic fuel cost setting.

- Benchmark approaches have skewed action distribution.
- Ensemble MARL diversifies actions to maximize overall reward value.
Conclusion

- **Summary:**
  - We formulated the ATC problem using MARL framework.
  - Proposed a novel deep ensemble MARL method to combine the power of a model-based kernel RL and model-free deep RL.
  - Ensemble MARL method improves the ATC objective by 9% over existing benchmarks on a real-world dataset.

- **Future Directions:**
  - Extend action space to incorporate additional controls such as directional and altitude changes.
  - Extend state space to handle take-off and landing scenarios.
  - Extend ensemble MARL to combine power of multiple methods.