Imperial College London

Action Branching Architectures for Deep Reinforcement Learning

Arash Tavakoli, Fabio Pardo & Petar Kormushev

1. Motivation and Objective

Goal:
- ▶ Enable the application of discrete-action reinforcement learning algorithms to domains with high-dimensional discrete or continuous action spaces

Motivation:
- ▶ Many problems have naturally discrete action spaces
- ▶ Discrete-action algorithms are central to many recent successes of deep reinforcement learning (e.g. DQN)

Problem: Their application is limited due to the combinatorial increase of the number of actions with increasing action dimensionality:
- ▶ Difficult to explore efficiently
- ▶ Computationally expensive

2. Action Branching Architecture

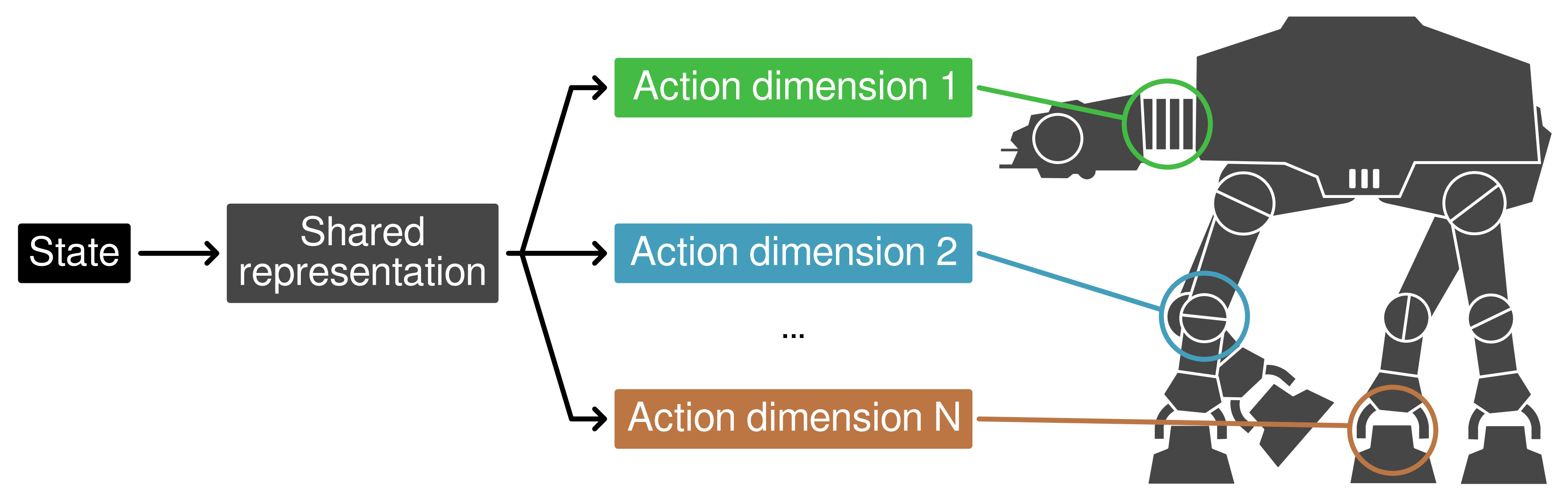


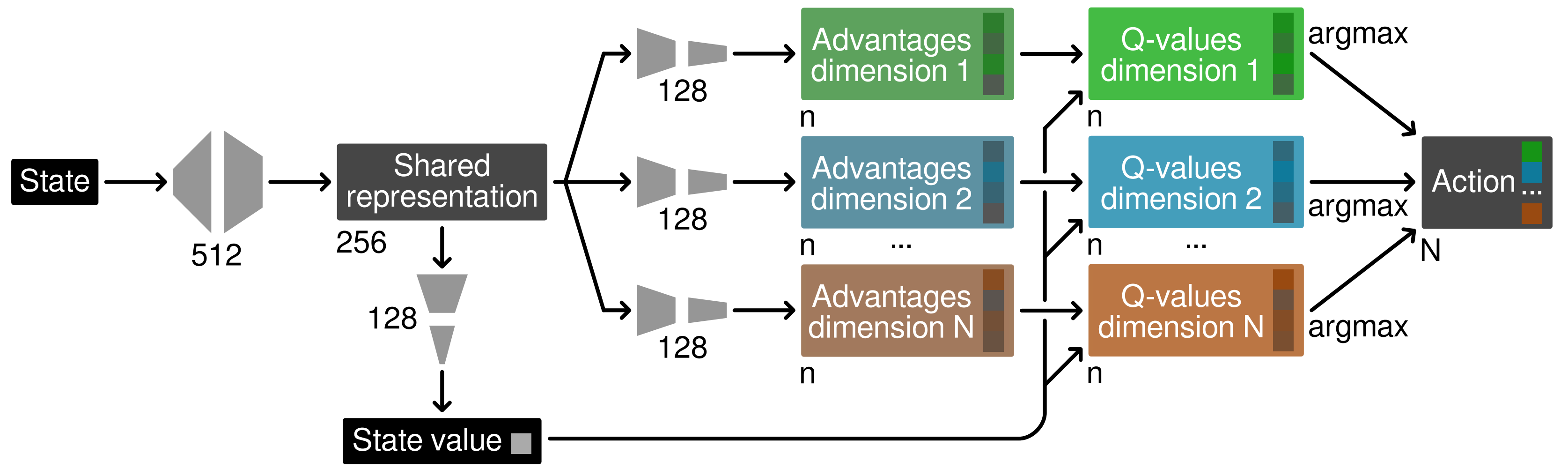
Illustration of the proposed action branching architecture.

Idea: We propose a novel neural architecture featuring a *shared decision module* followed by several network *branches*:
- ▶ Each action branch is responsible for controlling a degree of freedom
- ▶ Concatenation of the selected sub-actions results in a joint-action tuple
- ▶ Optimize for each action dimension with a degree of independence
- ▶ Achieves linear increase of the number of outputs with increasing action dimensionality

Hypothesis: The shared decision module can help coordinate the action branches and, thus, can stabilize training.
- ▶ Due to the backpropagation of the gradients originating from all the branches
- ▶ Without the shared part, agent is subject to issues of independent learning

Empirical Verification: We design a proof-of-concept DQN-based agent, called Branching Dueling Q-Network (BDQ), and compare it against its independent counterpart (i.e. without the shared part), called Independent Dueling Q-Network (IDQ).
- ▶ While specific to DQN, such verification suggests potential generalization to other reinforcement learning methods (e.g. policy gradient)

3. Branching Dueling Q-Network



Visualization of the action branching network implemented for BDQ.

We report the best performing methods used for incorporating the action branching architecture into the Dueling Double DQN (Dueling DDQN) agent:

▶ **Common State-Value Estimator:** We express a branch's Q-value at state s and sub-action a_d, in terms of the common state value $V(s)$ and the sub-action advantage $A_d(s, a_d)$, by:

$$Q_d(s, a_d) = V(s) + \left(A_d(s, a_d) - \frac{1}{n} \sum_{a'_d \in \mathcal{A}_d} A_d(s, a'_d) \right), \qquad (1)$$

where $d \in \{1, ..., N\}$ is an action dimension with n discrete sub-actions.

▶ **Temporal-Difference Target:** At every training step, we generate a single global temporal-difference (TD) target for all the action branches:

$$y = r + \gamma \frac{1}{N} \sum_{d} Q^-_d(s', \arg\max_{a'_d \in \mathcal{A}_d} Q_d(s', a'_d)), \qquad (2)$$

with Q^-_d denoting the branch d of the target network Q^-.

▶ **Loss:** We specify the training loss to be:

$$L = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}} \left[\frac{1}{N} \sum_{d} \left(y - Q_d(s, a_d)\right)^2 \right], \qquad (3)$$

where $\mathcal{D}$ denotes a (prioritized) replay buffer and a is a joint-action tuple $(a_1, a_2, ..., a_N)$.

▶ **Error for Experience Prioritization:** To adapt the prioritized replay, we aggregate the distributed TD errors (of a single transition) into a unified one using the following:

$$e_D(s, a, r, s') = \sum_{d} |y - Q_d(s, a_d)|, \qquad (4)$$

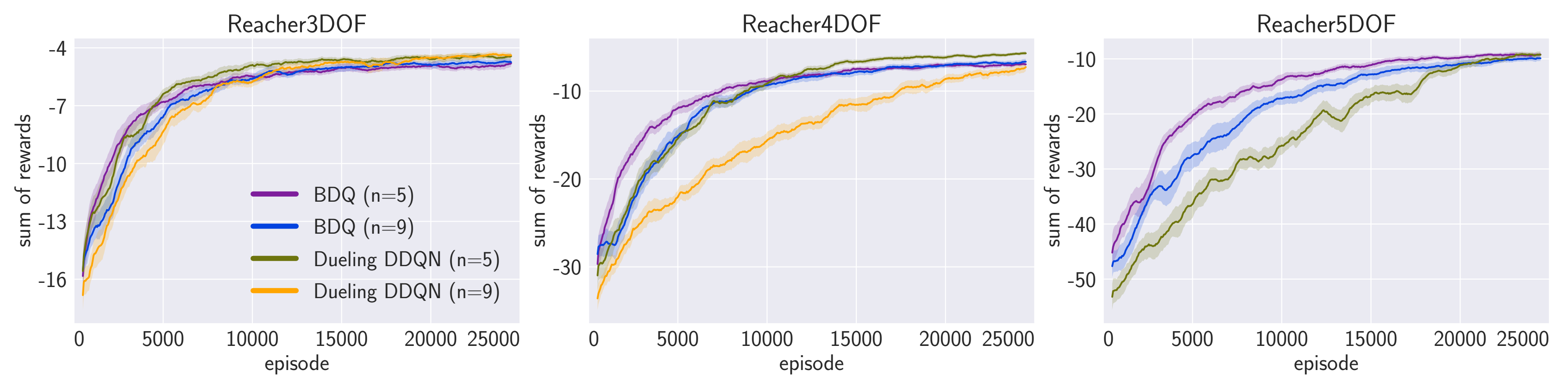
where $e_D(s, a, r, s')$ denotes the error used for prioritization of the transition tuple (s, a, r, s').

▶ **Gradient Rescaling:** Since all the branches backpropagate gradients through the shared decision module, we rescale the combined gradient prior to entering the shared part by $1/(N + 1)$.

4. Results

Custom N-Dimensional Action-Space Problems

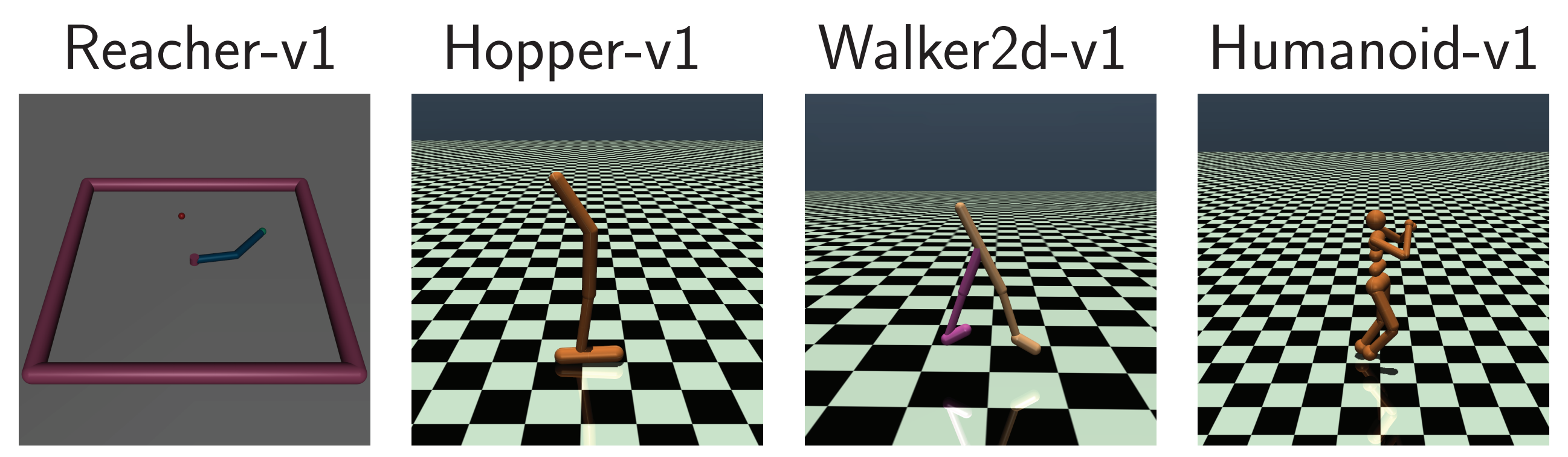
- ▶ We compare BDQ against its non-branching variant, Dueling DDQN
- ▶ **Tasks:** Physical reaching tasks with 3, 4, and 5 action dimensions, under two discretization levels: $n = \{5, 9\}$

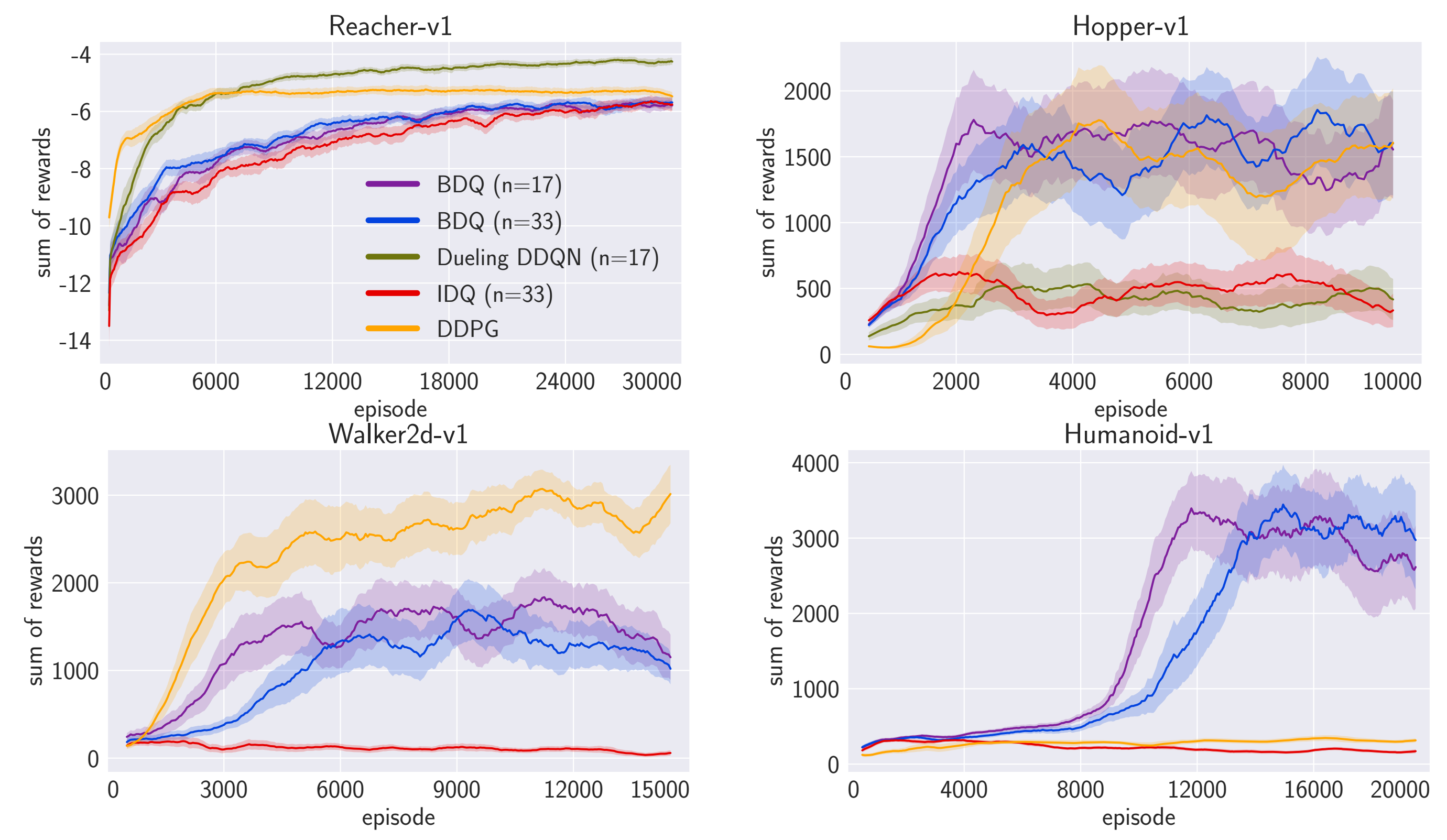


- ▶ BDQ scales robustly and outperforms Dueling DDQN with increasing action dimensionality and discretization granularity
- ▶ Unable to run Dueling DDQN ($n = 9$) for Reacher5DOF ($N = 5$) $\rightarrow$ computationally expensive (59049 actions in output)

Standard Continuous Control Benchmarks

- ▶ We compare BDQ against its independent counterpart, IDQ, and DDPG
- ▶ **Tasks:** Benchmark domains from the OpenAI's MuJoCo Gym collection with 2, 3, 6, and 17 degrees of freedom





- ▶ IDQ fails beyond $N = 2$ while BDQ successfully scales to $N = 17$
  - ▷ Verifies the significance of the shared part in coordination of actions
- ▶ BDQ is competitive to DDPG, outperforming it in Humanoid-v1
  - ▷ With a maximum of $\approx 6.5 \times 10^{25}$ possible actions for BDQ

Accepted for publication at the 32nd AAAI Conference on Artificial Intelligence (AAAI 2018), New Orleans, Louisiana, USA.
a.tavakoli@imperial.ac.uk