Paper Critique

Shuvrajeet Das, DA24D402

Course: DA7400, Fall 2024, IITM
Paper: [Quantifying Generalization in Reinforcement Learning]
Date: [25-10-2024]

Make sure your critique Address the following points:
1. The problem the paper is trying to address
2. Key contributions of the paper
3. Proposed algorithm/framework
4. How the proposed algorithm addressed the described problem
Note: Be concise with your explanations. Unnecessary verbosity will be penalized. Please don't exceed 2 pages.

---

1 The problem the paper is trying to address

Given a reinforcement learning agent trained on a set of environments $\mathcal{E}_{\text{train}}$, the goal is to quantify its generalization performance when evaluated on an unseen test set of environments $\mathcal{E}_{\text{test}}$.

The generalization gap can be defined as:

$$\text{Generalization Gap} = \mathbb{E}_{\mathcal{E}_{\text{test}}}[\text{Performance}] - \mathbb{E}_{\mathcal{E}_{\text{train}}}[\text{Performance}]$$

where $\mathbb{E}$ denotes the expected performance of the agent across the respective environment sets.

2 Key contributions of the paper

The key contributions of the paper are as follows:

1. The paper shows that the number of training environments required for good generalization is much larger than previously assumed in transfer learning for reinforcement learning.

2. It proposes a new environment, *CoinRun*, as a benchmark for generalization in reinforcement learning, along with a generalization metric to evaluate agent performance.

3. The paper evaluates the impact of different convolutional architectures and regularization techniques (e.g., L2 regularization, dropout, batch normalization, data augmentation) on generalization performance, showing that these methods significantly improve generalization.

3 Proposed algorithm/framework

The proposed framework in the paper is structured as follows:

1. **Environment:** The authors introduce *CoinRun*, a procedurally generated platformer environment, to evaluate generalization. Levels are generated using seeds, allowing for a large and varied set of training environments $\mathcal{E}_{\text{train}}$ and distinct test environments $\mathcal{E}_{\text{test}}$.

2. **Training:**

- Agents are trained using Proximal Policy Optimization (PPO) on $\mathcal{E}_{\text{train}}$ with convolutional architectures (Nature-CNN or IMPALA-CNN).
- Each agent interacts with different levels sampled from $\mathcal{E}_{\text{train}}$ for a fixed number of timesteps.

3. **Generalization Metric:**

$$\text{Generalization Gap} = \mathbb{E}_{\mathcal{E}_{\text{test}}}[\text{Performance}] - \mathbb{E}_{\mathcal{E}_{\text{train}}}[\text{Performance}]$$

where the test set $\mathcal{E}_{\text{test}}$ consists of unseen levels, and the generalization gap measures the performance difference between the training and test sets.

4. **Regularization and Architectures:** The framework incorporates regularization techniques such as L2 regularization, dropout, data augmentation, and batch normalization, to reduce overfitting and improve generalization.

4 How the proposed algorithm addressed the problem

1. **Procedural Generation of Environments:** By using the procedurally generated *CoinRun* environment, the algorithm ensures access to a large and varied set of training levels, $\mathcal{E}_{\text{train}}$, and distinct test levels, $\mathcal{E}_{\text{test}}$. This separation of training and test sets helps measure the agent's ability to generalize to unseen environments, addressing the issue of overfitting to a specific set of training environments.

2. **Generalization Metric:** The generalization gap, defined as:

$$\text{Generalization Gap} = \mathbb{E}_{\mathcal{E}_{\text{test}}}[\text{Performance}] - \mathbb{E}_{\mathcal{E}_{\text{train}}}[\text{Performance}],$$

quantifies how well an agent trained on $\mathcal{E}_{\text{train}}$ can perform on unseen environments $\mathcal{E}_{\text{test}}$. The larger the gap, the worse the generalization. This metric allows iterative improvements to agent architectures and training methods based on test performance, rather than solely on training performance.

3. **Architectural Improvements:** The algorithm leverages deeper convolutional architectures (e.g., IMPALA-CNN) that are shown to perform better in terms of generalization compared to simpler architectures like Nature-CNN. This helps reduce the tendency of the agent to overfit to the training environments.

4. **Regularization Techniques:** Incorporating techniques such as L2 regularization, dropout, batch normalization, and data augmentation helps mitigate overfitting. These methods, inspired by supervised learning, are shown to improve generalization by ensuring the agent learns more robust and generalizable features from the training environments.

5. **Training on Larger Environment Sets:** The algorithm demonstrates that training on a significantly larger number of environments helps reduce overfitting. As the number of unique training levels increases, the generalization gap decreases, addressing the need for larger, more varied training sets.