Ministry of Education and Science of Ukraine
National University “Kyiv-Mohyla Academy”
Faculty of Informatics
Informatics Department

Master's thesis
educational level – master

on the topic: “**ANALYSIS OF CURRICULUM LEARNING METHODS IN REINFORCEMENT LEARNING**”

By: 2-nd year student

of the educational program “Computer Science”, 121

Orel Danyil

Supervisor: Glybovets Mykola

Doctor of Physics and Mathematics, Professor

Reviewer: _____

The master's thesis was defended

with a grade _____

EC secretary _____

«____» _____ 2024 year

Kyiv – 2024

INDIVIDUAL TASK

FOR THE STUDENT'S THESIS

Danyil Orel

Topic “Analysis of Curriculum Learning methods in Reinforcement Learning”

Supervisor Glybovets Mykola, Doctor of Physics and Mathematics, Professor

Content of the master’s thesis:

SCHEDULE OF PREPARATION OF THE THESIS FOR THE DEFENSE

/p	Name of the stage of the master's thesis	The deadline for the stage	Note
1	Obtaining the topic of the master's thesis.	11.02.2023	
2	Search for thematic literature	15.03.2023	
3	Acquaintance with thematic materials and construction of the structure of the practical and theoretical parts of the master's thesis	30.03.2023	
4	Writing the introduction and content of the work	27.04.2023	
5	Acquaintance with existing principles and approaches to problem solving, search for algorithms that meet the requirements	17.05.2023	
6	Testing the functionality of the created application and making changes	23.04.2024	
7	Writing the second and third chapters based on the knowledge gained	27.05.2024	
8	Formatting the master's thesis in accordance with the specified requirements	04.06.2024	
9	Creating a presentation and writing a report to defend a master's thesis	05.06.2024	
10	Coordination of the previous version of the work with the manager	05.06.2024	
11	Making changes to the master's thesis in accordance with the supervisor's comments	05.06.2024	
12	Defense of the master's thesis	12.06.2024	

Schedule is agreed «___» _____ 2024 year

Supervisor Glybovets Mykola

Executor of master's thesis Orel Danyil

CONTENTS

ABSTRACT

This thesis presents a systematic evaluation of various Curriculum Learning methods across different simulated environments – CartPole, MountainCar, and Boxing – to analyze their efficacy in response to the unique challenges and requirements that each present. This research demonstrates that the complexity and dynamics of each task environment significantly influence the effectiveness of CL strategies, emphasizing the need for a tailored selection of methods. For simple, deterministic environments like CartPole, SPL methods like Polynomial and Logistic methods were found to enable rapid adaptation and achieve high performance due to their robustness in straightforward dynamics. In contrast, environments characterized by sparse and delayed rewards, such as MountainCar, required SPL methods like Polynomial that balance exploration and stability effectively. For complex, interactive settings like Boxing, Transfer learning and Anti-curriculum methods were preferred for their superior decision-making efficiency and reward maximization capabilities. The adaptability of algorithms also played a crucial role, with Linear and Teacher learning methods demonstrating rapid convergence ideal for time-sensitive scenarios, and SPL CL methods like Logistic or Polynomial excelling in environments with frequent negative outcomes. The insights derived offer a structured framework for selecting CL methods aligned with the specific characteristics of learning environments, enhancing the learning process's effectiveness while ensuring robust adaptability for real-world applications.

INTRODUCTION

The concept of Curriculum Learning (i.e., CL) was proposed by Bengio et al [1]. The paper introduced a fundamental mathematical definition of the CL and examples of its effectiveness for specific tasks. After a while, a lot of efforts were made to design various CL algorithms for different domains and modalities proving the advantages of the CL approach over non-CL and anti-CL methods [2]. But there is an open question about how to choose the most suitable CL algorithm in real-world applications. Based on the following questions from paper [2], there is an ongoing direction of CL worth of attention - benchmark evaluation. Various CL methods have been offered and demonstrated effective, but a few efforts were made on evaluating them with benchmarks.

CL literature divides CL methods into several categories by the level of automation [2]. Each category has its' own set of CL methods, which will be the **object of this research**:

- Pre-defined learning: root-p, one-pass.
- Self-paced learning: linear, logarithmic, logistic, polynomial, mixture, hard.
- Transfer learning.
- Teacher learning.
- Anti-curriculum.

In Reinforcement Learning (i.e., RL), environments and metrics from CL literature vary across applications [3]. For example, CIFAR dataset evaluate CL techniques in

---

[1] Bengio Y. Curriculum learning. *ICML*. 2009. P. 1–8. URL: https://dl.acm.org/doi/10.1145/1553374.1553380 (date of access: 01.10.2023).

[2] Wang X., Chen Y., Zhu W. A Survey on Curriculum learning. *IEEE*. 2021. P. 1–22. URL: https://ieeexplore.ieee.org/document/9392296/ (date of access: 12.10.2023).

[3] Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey / S. Narvekar et al. *Journal of Machine Learning Research 21(181)*. 2020. P. 1–50. URL: https://doi.org/10.48550/arXiv.2003.04960 (date of access: 16.11.2023).

image classification, meanwhile more complex environments like CartPole, MountainCar, and Atari games assess CL methods in RL. Each type of environment may use different metrics to evaluate the effectiveness and efficiency of CL strategies.

Designing agent-agnostic metrics to evaluate and compare CL methods in RL brings unique challenges. Benchmarks might include different RL environments characterized by varying levels of complexity, sparsity, and noise. Evaluation metrics should reflect performance improvements, convergence speed, and computational cost.

This research focuses on creating a benchmark evaluation for the RL problem using two primary approaches:

- *Isolated environments*: this research involves the systematic comparison of CL methodologies across a spectrum of RL environments, ranging from relatively straightforward settings such as CartPole and MountainCar to more complex scenarios presented in Atari games Boxing environment. This comparative analysis aims to highlight the effectiveness and adaptability of various CL approaches by examining their performance in environments that differ significantly in terms of state space complexity, action space diversity, sparsity of the rewards, and game time dynamics.
- *Benchmarking*: this study aims to refine the evaluation of RL strategies by expanding the traditional metric set, which typically includes learning stability, mean and standard deviation of rewards. To address the complexities of various RL environments, specialized metrics such as Average Adjusted Returns and Safe Exploration Score will be integrated by the methodologies outlined in works by Smith and Jones [4]. These additions are meant to provide a more detailed assessment of algorithm performance and safety across diverse settings.

Therefore, the **aim of this work** is to provide a comprehensive comparison of CL methods in RL across various scenarios and benchmark environments.

---

[4] B-Pref: Benchmarking Preference-Based Reinforcement Learning / K. Lee et al. 2021. P. 1–25. URL: https://doi.org/10.48550/arXiv.2111.03026 (date of access: 06.12.2023).

Research methods: analysis of scientific literature including papers and scientific journals.

Objectives of the study:

1. Study the concept of CL methods and review existing industry knowledge about benchmarking.
2. Conduct an analysis based on isolated environments and performance metrics to measure CL methods effectiveness in various environment conditions.
3. Develop generalized and intuitive strategies for selecting the most suitable CL algorithm based on the specific properties of environments.

In *the first chapter*, study contains the summarised industry knowledge about CL appliance best practices in context of RL and other domains which will be used across experiments to validate the obtained results.

In *the second chapter,* research contains description of environments, generalized metrics and RL architectures which would be employed across experiments.

The third chapter is devoted to experimental part containing tabular data for each experiment and corresponding analysis of CL methods under certain RL conditions.

The work consists of an introduction, three chapters, conclusions, and a list of references.

The scientific novelty of the obtained results: the growing complexity and diversity in the application of RL across various domains necessitate advanced methodologies to enhance learning efficiency and effectiveness. CL in RL offers a structured progression in learning tasks, potentially leading to faster convergence and better performance in complex environments. The actuality of the research lies in benchmarking and evaluating various CL methods systematically across different RL scenarios. This approach helps in identifying effective and unified strategies for training agents in diverse and dynamic environments, addressing the gap in existing literature where comprehensive, environment-specific benchmarks for CL are sparse.

The practical significance of the results obtained: the research is focused on developing and refining benchmarks that could guide the application of CL methods in real-world RL tasks. The comparative analysis across a spectrum of environments from simple (e.g., CartPole) to complex (e.g., Atari games) is critical. By integrating nuanced metrics such as Average Adjusted Returns and Safe Exploration Score, the research aims to offer practical insights into the adaptability and efficiency of CL methods under varying conditions. The practical significance of the study extends to providing a framework that can be utilized by researchers and practitioners to select appropriate CL strategies based on specific environmental characteristics. This work could facilitate more informed decision-making in deploying RL solutions, thus enhancing the potential for real-world applicability and impact.

CHAPTER 1: FOUNDATIONS AND FRAMEWORKS

1.1. Industry knowledge

CL was originally conceptualized by Bengio et al. [5], introducing a mathematical framework alongside practical evidence of its efficacy in task-specific settings. Subsequent research has expanded these foundational concepts to various domains, demonstrating the versatility and potential of CL to enhance learning algorithms. Despite extensive studies, the optimal selection of CL methods for real-world applications remains an unresolved question, particularly within the domain of RL.

In RL, CL has been explored to address the inherent complexities and learning inefficiencies presented by environments with sparse rewards and high dimensional state spaces. The adaptation of CL to RL has been marked by the development of specialized CL methods that cater to the unique requirements of RL tasks, such as sequential decision-making and exploration versus exploitation dilemmas.

For instance, research has shown that predefined learning schedules can significantly impact the learning curve in RL scenarios. Cirik et al. [6] demonstrated that specific predefined curriculum schedules could optimize training in sequence prediction tasks. In more dynamic settings, such as those encountered in RL, these schedules help in modulating the difficulty of tasks presented to the agent, thereby potentially accelerating the learning process.

Further, the work by Zhang et al. [7] and Hacohen et al. [8] explores the nuances of implementing CL in complex learning environments. Zhang et al. evaluated various predefined difficulty measures and training schedulers in the context of neural machine

---

[5] Bengio Y. Curriculum learning. *ICML*. 2009. P. 1–8. URL: https://dl.acm.org/doi/10.1145/1553374.1553380 (date of access: 01.10.2023).

[6] Cirik V., Hovy E., Morency L.-P. Visualizing and understanding curriculum learning for long short-term memory networks. 2016. P. 1–7. URL: https://doi.org/10.48550/arXiv.1611.06204 (date of access: 13.12.2023).

[7] An Empirical Exploration of Curriculum Learning for Neural Machine Translation / X. Zhang et al. P. 1–16. URL: https://doi.org/10.48550/arXiv.1811.00739 (date of access: 04.01.2024).

[8] Hacohen G., Weinshall D. On The Power of Curriculum Learning in Training Deep Networks. *ICML*. 2019. P. 1–13. URL: https://doi.org/10.48550/arXiv.1904.03626 (date of access: 08.01.2024).

translation, a domain closely related to RL in terms of sequential data processing. Their findings suggest that the effectiveness of CL is highly contingent on the precise configuration of the difficulty measurer and scheduler. Moreover, Hacohen et al. provide insights into the comparative effectiveness of Self-paced learning, Anti-curriculum, and Transfer learning approaches in RL-like settings, particularly highlighting how transfer learning can leverage pre-existing knowledge to facilitate more effective learning strategies.

The empirical studies mentioned provide a foundation for applying and validating CL methods in RL, guiding the selection of appropriate strategies based on the characteristics of the RL environment and the specific learning objectives. This research aims to extend these foundational studies by developing a nuanced benchmarking framework that assesses the efficacy of various CL strategies in distinct RL scenarios, thus contributing to a more systematic understanding and application of CL in RL.

1.2. Existing approaches

The adoption of CL in RL has been influenced by foundational research and empirical studies that demonstrate the efficacy of structured learning progressions in complex environments. This section outlines several prominent approaches to CL within RL, detailing the methodologies and specific applications that have shaped current practices.

- *Predefined Curriculum Learning:* In RL, predefined curriculum strategies often involve the gradual introduction of challenges to the learning agent. For instance, Narvekar et al. [9] illustrated how incrementally increasing the difficulty of tasks could significantly enhance learning efficiency in environments like CartPole and MountainCar. These strategies are particularly effective in scenarios where agents must learn complex sequences of actions before achieving proficiency.

[9] Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey / S. Narvekar et al. *Journal of Machine Learning Research 21(181)*. 2020. P. 1–50. URL: https://doi.org/10.48550/arXiv.2003.04960 (date of access: 16.11.2023).

- *Self-Paced Learning (SPL):* SPL has been tailored for RL through the adaptive adjustment of learning challenges based on the agent's performance. Research by Abel et al. [10] demonstrated that SPL could optimize exploration strategies in RL, allowing agents to focus on learning from the most informative experiences as dictated by their current state of knowledge.
- *Transfer Learning and Teacher Approaches:* Incorporating transfer learning within CL frameworks involves leveraging knowledge from simpler tasks to accelerate learning in more complex scenarios. A significant example includes the work by Taylor et al. [11] who employed transfer learning to improve agent adaptability across different RL tasks, showcasing marked improvements in learning speeds and overall task performance.
- *Automated Curriculum Learning:* Fully automated CL methods, such as those employing RL techniques to determine the sequence of training challenges, represent the cutting edge in adaptive learning strategies. Riemer et al. [12] explored meta-learning approaches within an RL framework to dynamically adjust curriculum difficulty, effectively tailoring the learning process to the evolving capabilities of the agent.

Each of these approaches offers unique advantages and has been proven effective in various RL settings. However, the choice of a particular curriculum strategy often depends on specific task requirements, available computational resources, and the desired speed of convergence. As such, ongoing research continues to refine these strategies, aiming to optimize their integration and effectiveness across a broader spectrum of RL applications:

---

[10] Exploratory Gradient Boosting for Reinforcement Learning in Complex Domains / D. Abel et al. 2016. P. 1–8. URL: https://doi.org/10.48550/arXiv.1603.04119 (date of access: 06.02.2024).

[11] Taylor M., Stone P. Transfer Learning for Reinforcement Learning Domains: A Survey. *Journal of Machine Learning Research 10*. 2009. P. 1–53. URL: https://dl.acm.org/doi/10.5555/1577069.1755839 (date of access: 23.02.2024).

[12] Learning to Learn without Forgetting by Maximizing Transfer and Minimizing Interference / M. Riemer et al. *ICLR*. 2018. P. 1–31. URL: https://doi.org/10.48550/arXiv.1810.11910 (date of access: 03.03.2024).

- *Comparative Studies and Benchmarks:* Recent studies have begun to establish benchmarks that compare the effectiveness of different CL methods in RL. These benchmarks are crucial for evaluating the relative performance of curriculum strategies under standardized conditions, providing insights that help refine CL applications in RL. The work by Justesen et al. [13] provides a comprehensive analysis of CL methods across multiple game-based RL environments, highlighting the conditions under which certain CL strategies outperform others.
- *Future Directions*: As the field progresses, the integration of sophisticated AI models like deep neural networks with CL poses a promising avenue for further research. This integration has the potential to significantly enhance the autonomy and efficiency of learning agents, especially in high-dimensional and complex RL environments.

Conclusions to chapter 1

The empirical studies mentioned, such as those by Cirik et al. [14], Zhang et al. [15], and Hacohen et al. [16], emphasize the necessity of grounding CL methods in solid empirical evidence. Benchmarking offers a way to systematically validate different CL strategies across diverse RL scenarios, ensuring that theoretical advancements are backed by practical results.

Through benchmarking, researchers can identify the most effective CL strategies for specific types of RL environments and learning objectives. This is particularly

---

[13] Illuminating Generalization in Deep Reinforcement Learning through Procedural Level Generation / N. Justesen et al. *NeurIPS Deep RL Workshop 2018*. 2018. P. 1–10. URL: https://doi.org/10.48550/arXiv.1806.10729 (date of access: 14.03.2024).

[14] Cirik V., Hovy E., Morency L.-P. Visualizing and understanding curriculum learning for long short-term memory networks. 2016. P. 1–7. URL: https://doi.org/10.48550/arXiv.1611.06204 (date of access: 13.12.2023).

[15] An Empirical Exploration of Curriculum Learning for Neural Machine Translation / X. Zhang et al. P. 1–16. URL: https://doi.org/10.48550/arXiv.1811.00739 (date of access: 04.01.2024).

[16] Hacohen G., Weinshall D. On The Power of Curriculum Learning in Training Deep Networks. *ICML*. 2019. P. 1–13. URL: https://doi.org/10.48550/arXiv.1904.03626 (date of access: 08.01.2024).

important in RL due to its inherent complexities, such as sparse rewards and high-dimensional state spaces. Benchmarking helps in fine-tuning learning schedules and curriculum configurations that best suit the unique requirements of different RL tasks.

CHAPTER 2: EXPERIMENTAL SETUP

2.1. Isolated environments

Various environments have various properties which can possibly impact the results of the measurements and consequently the assumptions that are made based on the results. There is a need to isolate measurements and provide a set of datasets with diverse properties. Here are the ones used to research the RL problem in this study: CartPole, MountainCar, and Atari games (e.g., Boxing).

2.1.1. Environments properties

CartPole is a fundamental benchmark problem in RL, characterized by its relative simplicity in state space, action space, and simple reward mechanism.

The state in CartPole is defined by four continuous variables:

- *Cart position* on a horizontal track with the midpoint as zero.
- *Cart velocity,* which may be positive or negative.
- *Pole's angle,* expressed in radians, with respect to vertical, starting upright at zero degrees.
- *Pole's angular velocity,* which indicates the tilt rate.

The are only two possible actions in the discrete action space:

- *Pushing the cart left,* which moves it leftward by applying force.
- *Pushing the cart right,* which moves it in the opposite direction.

The reward system is straightforward, awarding a single point for each timestep the pole remains upright, with the objective of maintaining balance for up to 200 timesteps.

An episode terminates if the pole's angle exceeds ±12 degrees, the cart's position strays beyond ±2.4 units from the center, or if 200 steps are reached. These conditions are designed to prevent trivial solutions and encourage efficient balancing strategies.

The system dynamics are relatively simple and deterministic. The goal is to balance the pole on the cart, which makes it an episodic task with potentially short episodes if the pole falls quickly.

MountainCar a classic benchmark in RL, which represents a problem where an underpowered car must drive up a steep hill. The car is initially positioned between two hills and the goal is to reach the top of the right hill which is marked with a flag. This environment is typically used to test algorithms' ability to find solutions in situations with *sparse* or *delayed* rewards.

The state space of the MountainCar environment is two-dimensional:

- *Car position,* which is the car's horizontal position along the track, bounded between two values, typically (-1.2) and (0.6). This represents the most leftward and rightward extents of the track, respectively.
- *Car velocity,* which is varying between (-0.07) and (0.07). This variable captures the rate of change in the car's position over time.

The action space in this environment is discrete and consists of three possible actions:

- *Accelerate to the left*, which increases the car's velocity in the leftward direction.
- *Accelerate to the right*, which increases the car's velocity in the rightward direction.
- *Do nothing* leaves the car's velocity unchanged.

The reward function in the MountainCar is straightforward but challenging due to its sparsity: a reward of (-1) is received for each time step until the car reaches the flag, encouraging the agent to solve the problem as quickly as possible. The episode ends when time step has reached 1,000 values. As soon as the environment has sparse rewards, most of mean reward values will contain evaluations around -1,000 meaning that the agent has not received enough exploitative data to achieve the target optimal value.

The dynamics are somewhat challenging due to the requirement of building up momentum to reach the goal. The reward is sparse and only given when the car reaches the top of the mountain, introducing significant reward delay. This requires a more strategic approach compared to CartPole.

Atari games a set of game environments, which are commonly used for RL benchmarking. The state space in Atari games is significantly more complex, usually represented by the pixel data from the game screen or a reduced representation of the game state. This makes the state space high-dimensional and diverse across different games.

Boxing a popular Atari 2600 game, challenges players in a digital boxing ring. Players control a boxer and aim to score points by hitting the opponent. This environment tests the AI's ability to handle direct adversarial confrontation and strategic positioning.

The state space in Boxing is multi-dimensional and includes:

- *Boxer position,* which is horizontal and vertical position of both the player and the opponent within the ring.
- *Boxer action* defines current actions being performed, such as punching or blocking.

The action space in this game is discrete and involves several options:

- *Move left or right* helps to dodge or align for punches.
- *Punch* attempts to score by hitting the opponent.
- *Block* reduces the potential score from opponent's punches.

The reward function in Boxing awards points for successful punches landed on the opponent, with different scores based on the type and precision of the punch. The game's challenge lies in effectively managing offensive and defensive strategies to maximize the score while minimizing hits received from the opponent.

The game dynamics are more complex in comparison with CartPole or MountainCar environments due to the interaction with an opponent and continuous scoring throughout the game.

2.1.2. Environments comparison

- *State Space Complexity*: Boxing have potentially higher complexity in state space due to the high dimensionality of raw pixel data compared to the more

structured and lower-dimensional state descriptions in CartPole and MountainCar.

- *Action Space Diversity*: Boxing has the most diverse action space with multiple movement and attack combinations, followed by CartPole and MountainCar have simpler action spaces.
- *Sparsity of rewards*: CartPole and Boxing are characterized by dense reward structures, where feedback is immediate and frequent, simplifying the learning process as the agent quickly learns the consequences of its actions. MountainCar, on the other hand, presents a sparse reward challenge, requiring the agent to develop a more explorative strategy and possibly utilize techniques like reward shaping, or use of an intrinsic motivation mechanism to overcome the lack of feedback and encourage beneficial behaviors that do not yield immediate rewards.
- *Temporal Dynamics*: Boxing offer more complex dynamics with interactive elements and progressive difficulty. CartPole and MountainCar have simpler and more predictable dynamics, although MountainCar introduces complexity with its requirement for strategic momentum build-up.

2.2. Metric design

In the field of RL, the assessment of an agent's performance and the effectiveness of its learning strategy are crucial for understanding and improving its capabilities. Metrics play a pivotal role in this process, providing quantitative measures that capture various aspects of agent behavior and learning dynamics. These metrics not only help in fine-tuning the algorithms but also facilitate a deeper understanding of how agents interact with complex environments. This section introduces and elaborates on several key metrics that are commonly used to evaluate the performance of RL agents. Each metric addresses a specific aspect of learning, from the efficiency of reward acquisition to the safety and consistency of the agent's behavior and decision-making processes.

The following discussion describes a set of evaluative metrics selected to provide comparative analysis of chosen CL methods and their implementation within various

RL agent architectures. This selection aims to ensure a comprehensive assessment across multiple dimensions of agent performance and learning efficiency.

- *Average Adjusted Returns (AAR):* metric quantifies the average effectiveness of an RL agent by adjusting the rewards received during an episode based on the difficulty of the tasks performed. This adjustment aims to normalize the performance across different scenarios or levels of challenge, facilitating a fair comparison between various training sessions or models.
- *Safe Exploration Score (SES):* evaluates the safety of the agent's exploration strategies within its environment. It is calculated by assessing the proportion of actions taken that do not lead to the termination of an episode, normalized by the total number of exploratory actions. A higher SES indicates that the agent is effectively exploring its environment while minimizing the risk of taking actions that would lead to failure or unsafe outcomes.
- *Learning Stability*: metric that reflects the consistency and reliability of an agent's training process over time. It is typically measured by the variance or standard deviation of the total rewards obtained across a series of episodes during training. Lower variance indicates more stable learning, suggesting that the agent is consistently applying its learned strategies effectively across different episodes.
- *Mean Reward:* mean reward that an agent accumulates over all episodes within a specified period or over the entirety of the evaluation sessions. This metric averages up all the rewards collected by the agent, providing a comprehensive view of its overall effectiveness in achieving the goals set within the environment throughout its evaluation process.
- *Std Reward:* standard deviation of the reward that an agent accumulates over all episodes within a specified period or over the entirety of the evaluation sessions. This metric defines an overview of its overall stability during evaluation process.
- *Max Reward:* maximum value of the reward that an agent receives over all episodes within a specified period or over the entirety of the evaluation sessions. This metric defines an overview of its extremum performance capabilities.

2.3. Agent architectures

The diversity of the environmental setups in this study necessitates a varied selection of RL agents, each with its architectural strengths and limitations. Detailed discussions of the RL architectures employed for this analysis are provided below, highlighting their operational mechanisms and applicability:

Q-Learning architecture utilizes a tabular approach where the actions are columns, and the states are rows within the table. It inherently suffers from the curse of dimensionality, making it less feasible for complex real-world applications with large state spaces. However, for environments with a small number of discrete states, or where state spaces can be effectively discretized, Q-Learning remains a viable and straightforward approach for introducing basic RL principles and strategies.

Double-Q Network (DQN) is an extension of the basic Q-Learning algorithm, the DQN utilizes two separate neural networks to estimate the Q-value functions. This architecture helps to reduce the overestimation bias often observed in traditional Q-Learning by decoupling the selection of the action from its evaluation. DQNs are particularly effective in handling environments with high-dimensional state spaces, such as those represented by pixel data from video games, making them suitable for complex tasks where state abstraction is necessary. Meanwhile, the architecture for all experimental setups with DQN is represented with such components:

- *Loss Function*: Smooth L1 was selected as the target function for the optimization task of DQN agent. It is preferred choice due to the ability to accommodate to small and large errors without being overly influenced by outliers.
- *Optimizer*: Adam was used to enhance training speed in deep neural networks, facilitating rapid convergence.
- *Learning Rate Scheduler*: An exponential scheduler with a gamma value fixed at 0.0001 was employed to finely adjust the weights of the DQN quality network.
- *Gradient Clipping*: Parameters were clipped with a maximum norm of 1.0 to ensure normalized weight updates and prevent gradient explosion.

Proximal-Policy Optimization (PPO): is an advanced policy gradient approach that optimizes a special objective function designed to take small steps to improve policy while ensuring the new policy is not too far from the old. This method employs neural networks to directly model the policy and estimate the value functions, making it robust to a wide range of environments. Its greatest strength lies in its stability and effectiveness in continuous action spaces, which are common in more sophisticated simulations and real-world scenarios. The drawback of PPO usage in current experimental setup is that SES metric cannot be recorded for current agent due to its lack of a specific exploration mechanism, unlike the DQN agent, which allows the adjustment of the exploration rate through parameters like epsilon in epsilon-greedy exploration. In DQN, the exploration rate controls the trade-off between exploration (choosing random actions) and exploitation (choosing the best-known actions), which directly impacts the sample efficiency. Since PPO does not have a comparable mechanism to explicitly adjust exploration in the same way, the SES metric measurements are not applicable and thus excluded from the measurements and experimental results. Nevertheless, here is the architecture decisions which were employed for all experimental setups with PPO:

- *Loss Function*: Smooth L1 was selected as the target function for the optimization task. It is preferred choice due to the ability to accommodate to small and large errors without being overly influenced by outliers.
- *Optimizer*: Adam was used to enhance training speed in deep neural networks, facilitating rapid convergence.
- *Learning Rate Scheduler*: An exponential scheduler with a gamma value fixed at 0.0001 was employed to finely adjust the weights of the PPO neural network.
- *Gradient Clipping*: Parameters were clipped with a maximum norm of 1.0 to ensure normalized weight updates and prevent gradient explosion.
- *Clip Epsilon:* Fixed at 0.2, this clipping value helps in maintaining the trust region during updates.
- *GAE Lambda:* Set to 0.99, it balances bias and variance in the advantage estimates.

Each of these architectures offers distinct advantages depending on the specific requirements of the task and the complexity of the environment. Q-Learning Tables are best suited for simple, discrete tasks; DQNs are preferred for environments requiring significant state processing, while PPO provides state-of-the-art performance in scenarios demanding sophisticated policy inference and stability.

Conclusions to chapter 2

The analysis provides a comprehensive overview of various RL environments, illustrating the diverse challenges and requirements each presents. By comparing environments like CartPole, MountainCar, and Boxing, we see significant differences in their complexity and the strategies required for agent success. Additionally, the chapter details the performance metrics and architectures essential for evaluating and enhancing RL agents' effectiveness across these varied settings. This foundation supports deeper exploration into how different RL methods perform under varying conditions, outlining configurations and challenges for the following experiments and benchmarking.

CHAPTER 3: EXPERIMENTS

From a software perspective, experiments were conducted using a PyTorch framework [17] and NumPy library [18]. For benchmarking RL tasks, the Gym [19] library was utilized, which provides a diverse collection of environments suited for RL research, including CartPole, MountainCar, and Atari games Boxing environments.

From a hardware perspective, experiments were executed on a single GPU core of an RTX 3090 machine, which supports GPU acceleration. This configuration was chosen to speed up the training process of RL models, which are computationally intensive and time-consuming on CPU-only setups. GPU acceleration was applied exclusively to DQN and PPO agents due to their reliance on tensor operations and batch processing. In contrast, the Q-learning agent did not gain significant performance from GPU utilization.

From a theoretical perspective, Henderson et al. [20] state that a general practice in benchmarking various RL algorithms involves the repetition of experiments to mitigate variability and enhance consistency across experimental runs. Employing evaluation on top-N trials to be selected among several trials or average only small number of trials ($N < 5$) leads to potential issues in evaluations. In this research, to assure the reliability of the results, each experiment was meticulously replicated five times ($N = 5$) for each agent and each CL method, establishing a robust statistical foundation for the conclusions drawn. Additionally, to ensure the reliability of benchmarking, each

---

[17] Paszke A., Gross S., Massa F. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *NeurIPS*. 2019. P. 1–12. URL: https://doi.org/10.48550/arXiv.1912.01703 (date of access: 05.06.2024).

[18] Harris C., Millman J., van der Walt S. Array Programming with NumPy. *Nature*. 2020. P. 1–8. URL: https://doi.org/10.48550/arXiv.2006.10256 (date of access: 05.06.2024).

[19] OpenAI Gym / G. Brockman et al. 2016. P. 1–3. URL: https://doi.org/10.48550/arXiv.1606.01540 (date of access: 05.06.2024).

[20] Deep Reinforcement Learning That Matters / Henderson et al. *Proceedings of the AAAI Conference on Artificial Intelligence*. 2017. P. 1–8. URL: https://doi.org/10.1609/aaai.v32i1.11694 (date of access: 04.06.2024).

agent was allocated its own training environment while sharing a common evaluation environment. The training environments were configured with curriculum adjustments tailored to each CL method, serving as sandboxes to improve pattern learning and accelerate the agents' experiential learning. After each training episode, all agents were evaluated in the original environment which does not contain any adjustments of environmental or curriculum parameters.

3.1. CartPole: experimental results

In this setting, a set of metrics enhance with additional measures that provide insight into each agent's performance, specifically under the curriculum that best facilitates its training:

- *Convergence Speed*: this metric represents the first instance of achieving a reward indicative of optimal agent performance. In this environment, convergence speed is measured by the first occurrence of evaluation required for the agent to achieve a mean reward of 500.0 in reward distribution during evaluation.

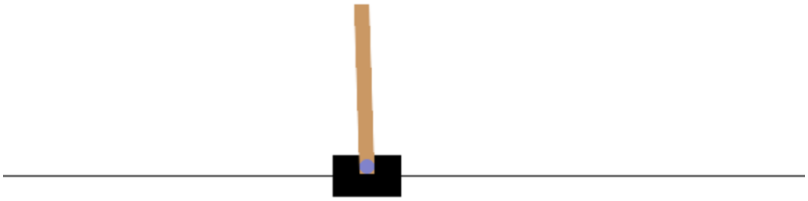


Fig 1: Visualization of a single training episode of a pre-trained DQN agent trained on Teacher learning CL in CartPole environment (created by author)

To define a curriculum for the CartPole environment, from *Fig 1*, the decision has been made to adjust the pole length. A shorter pole length should enable the cart to

learn the balancing patterns harder. Conversely, as the pole length increases, the environment and the task become easier. Here is an example of Linear CL method:

$$\text{length}_{new}\,(\text{ep}) = \text{length}_{max} - (\text{length}_{max} - \text{length}_{min}) \times \frac{\text{ep}}{total\ eps}$$

where:

- *length* – CL parameter which relates to pole length: *max* corresponds to maximum possible pole length in experimental setup, set at 1.0, and *min* corresponds to minimum one, set at 0.5, meanwhile *new* is bounded by the range and corresponds to the value which should be computed and employed for the next training iteration.
- *ep* – current training episode on which update to pole length is done.
- *total eps* – amount of training episodes which must be done by the agent (a hyperparameter).

After finalising a comprehensive set of evaluation metrics and defining a CL strategy for the environment, an analysis for each agent individually can be provided.

3.1.1. Q-Learning

Agent is optimized to facilitate robust learning performance in RL tasks. The setup parameters were determined through preliminary trials to maximize efficiency. The configuration is as follows:

- *Total Episodes:* The agent is trained across 50,000 episodes to ensure adequate learning over diverse state-action spaces.
- *Alpha (α):* The learning rate is set to 0.1, balancing the trade-off between learning speed and stability.
- *Gamma (γ):* The discount factor is fixed at 0.95, cautiously prioritizing long-term rewards.
- *Initial Epsilon (ϵ):* Set at 1.0 to promote a robust exploratory approach at the beginning of the learning process.

- *Minimum Epsilon:* The epsilon value decays to a floor of 0.005, ensuring that the agent retains a minimal level of exploration.
- *Epsilon Decay:* The decay rate for epsilon is set to 0.99995, facilitating a gradual transition from exploration to exploitation.

Following the collection of 60,000 data points from experiments on a Q-Learning agent across different CL configurations, the following observations have been systematically recorded in *Table 1*.

Table 1

Q-Learning	Metrics						
	AAR	SES	Learning Stability	Mean Reward	Std Reward	Max Reward	Convergence Speed
Anti-curriculum	1.39	0.89	20.64	38.69	16.92	71.90	N/A
Baseline	1.00	0.90	13.32	28.11	14.93	50.90	N/A
Hard	1.43	0.93	9.28	16.62	10.25	38.10	N/A
Linear	1.39	0.93	9.96	16.91	8.51	37.00	N/A
Logarithmic	1.07	0.91	15.78	32.79	18.35	62.40	N/A
Logistic	1.43	**0.95**	23.16	**72.65**	65.36	**212.00**	N/A
Mixture	1.48	0.94	10.23	20.77	9.69	37.50	N/A
One-pass	1.39	0.93	6.27	19.89	**4.13**	26.40	N/A
Polynomial	1.57	0.94	9.10	32.02	19.05	64.90	N/A
Root-p	1.33	0.93	22.92	51.84	24.38	80.50	N/A
Teacher learning	**1.90**	0.92	**5.61**	20.67	9.78	41.00	N/A
Transfer learning	1.19	0.93	8.44	22.61	10.61	39.20	N/A

Table 1: Comparative analysis of CL methods based on Q-Learning agent in CartPole environment. The table displays AAR, SES, Learning Stability, Mean Reward, Standard Deviation of Reward (Std Reward), Max Reward, and Convergence Speed for distinct CL strategies. Highlighted values indicate the highest performance in each metric,

illustrating the efficacy and particular strengths of each configuration in specific aspects of learning and exploration.

Here is a summary of the results from *Table 1*:

- *AAR:* The least positive AAR is observed in the Teacher learning method (1.90), indicating its relatively better performance in decision-making under the defined setup.
- *SES*: Logistic method displays very high safe exploration score (0.95), suggesting that these methods are more cautious, minimizing risks during exploration.
- *Learning Stability*: Logistic method shows the highest learning stability metric (23.16), indicating a more explorative and possibly less consistent reward signal, which might be suitable for environments that require exploratory strategies. Conversely, Teacher learning approach yields the lowest Learning Stability level (5.61) which corresponds to higher consistency in training.
- *Mean Reward:* Logistic has shown the highest mean reward value (72.65) suggesting the best adoptability during learning.
- *Std Reward:* One-pass methods demonstrate lower standard deviations in rewards (4.13), indicating more predictable and stable performance during evaluations. Meanwhile Logistic method exhibits the highest oscillations in rewards (65.36) during evaluation meaning less stability and unpredictability.
- *Max Reward:* Logistic method achieves the highest maximum reward (212.00), suggesting potential for achieving the best outcomes under certain conditions.
- *Convergence Speed:* Since no methods have reached the defined threshold for convergence speed, it remains NaN for all methods.

3.1.2. DQN

Agent optimization ensures robust performance in RL tasks. Configuration parameters were refined through initial testing to enhance efficiency. Below is a detailed configuration outline:

- *Total Episodes:* The agent undergoes training over 5,000 episodes to ensure comprehensive adaptation across varied state-action spaces.
- *Alpha (α):* The learning rate is meticulously set at 0.0015 to maintain a balance between learning speed and system stability.
- *Gamma (γ):* The discount factor is steadfast at 0.99, emphasizing the significance of long-term rewards.
- *Initial Epsilon (ϵ):* Initiated at 1.0, this parameter encourages an exploratory strategy in the early stages of training.
- *Minimum Epsilon:* Epsilon decays to a minimum threshold of 0.01, preserving an essential level of exploration throughout the learning phase.
- *Epsilon Decay:* The epsilon decay rate is established at 0.9999, ensuring a smooth transition from exploration to exploitation as training progresses.
- *Buffer Size:* The replay buffer is designed to store up to 50,000 experiences, enabling the DQN agent to learn from a vast array of past interactions.
- *Batch Size:* The agent processes batches of 128 experiences from the replay buffer to refine its policy, optimizing decision-making processes.
- *Update Interval:* The model update frequency is set at every 500 episodes, aligning the target model closely with the operational model to enhance performance consistency.

Following the acquisition of 1,212 data points from a DQN agent across diverse CL configurations, the subsequent observations have been documented from the experiments in *Table 2*.

Table 2

DQN	Metrics						
	AAR	SES	Learning Stability	Mean Reward	Std Reward	Max Reward	Convergence Speed
Anti-curriculum	1.39	0.98	40.75	123.25	16.00	376.40	N/A
Baseline	1.00	**0.99**	53.62	60.57	16.31	119.14	N/A
Hard	1.34	0.98	29.15	95.22	12.71	159.42	N/A
Linear	1.39	0.98	53.20	78.34	18.03	155.68	N/A
Logarithmic	1.12	0.98	55.65	128.79	14.31	255.58	N/A
Logistic	1.43	**0.99**	61.53	178.47	18.01	355.94	N/A
Mixture	1.51	**0.99**	52.65	73.23	17.03	146.46	N/A
One-pass	1.38	0.98	54.41	32.48	15.34	64.96	N/A
Polynomial	**1.57**	0.98	42.31	232.81	15.46	465.62	N/A
Root-p	1.33	**0.99**	41.38	42.83	**10.51**	85.66	N/A
Teacher learning	1.44	0.98	51.61	**250.00**	12.82	**500.00**	**3**
Transfer learning	1.19	0.98	**23.44**	222.44	12.10	444.88	N/A

Table 2: Comparative analysis of CL methods based on DQN agent trained in CartPole environment. The table displays the AAR, SES, Learning Stability, Mean Reward, Standard Deviation of Reward (Std Reward), Maximum Reward, and Convergence Speed. Highlighted values indicate the highest performance in each metric, illustrating the efficacy and particular strengths of each configuration in specific aspects of learning and exploration.

Given the *Table 2* results, the analysis was conducted and documented as following:

- *AAR*: Polynomial method achieves the highest AAR of 1.57, indicating superior decision quality and adaptation to the environment compared to other methods. This high AAR suggests that the Polynomial method is particularly effective in understanding and responding to the complexities of the CartPole environment.

- *SES*: Baseline, Mixture, Linear and Root-p methods show the highest SES of 0.99, suggesting optimal safety during exploration. This indicates that they maintain a safer exploration strategy than other CL methods.
- *Learning Stability*: Transfer learning method exhibits the lowest Learning Stability of 23.44, indicating that it retrieves rewards in a more stable manner during training. Conversely, Logistic method, with the highest Learning Stability metric of 61.53, receives more variable signals, which suggests a higher degree of exploration and potentially higher adaptability in dynamic environments during training.
- *Mean Reward:* Teacher learning method outperforms other methods in terms of mean reward, achieving 250.0, which highlights its effectiveness in maximizing returns during evaluation sessions.
- *Std Reward*: Linear method shows the highest variability in rewards, achieving 18.03, which might suggest less predictability in its evaluation outcomes. Meanwhile Root-p method yields the most stable evaluation results.
- *Convergence Speed*: Teacher learning method is the fastest to converge to high rewards, with a convergence speed of only 3 evaluations, demonstrating its capability for quick adaptation and efficient learning.

Analysis of CL methods in CartPole environment

A combination of experimental observations from *Table 1* and *Table 2* gives such set of insights:

- Polynomial and Logistic methods provide the most robust performance in terms of high AAR and maximum rewards per evaluation episodes, making them ideal for scenarios requiring rapid adaptation and high performance which confirms thoughts of Abel et al. [21] suggesting that SPL methods provide more informative experience during learning.

[21] Exploratory Gradient Boosting for Reinforcement Learning in Complex Domains / D. Abel et al. 2016. P. 1–8. URL: https://doi.org/10.48550/arXiv.1603.04119 (date of access: 06.02.2024).

- Teacher learning methods is the only method which converged to optimal solution in DQN setup, it offers the highest mean reward making it suitable for environments where consistent performance and quick convergence is critical which confirms Taylor et al. [22] ideas about improvements of performance Teacher learning and Transfer learning CL methods for simpler tasks like CartPole environment.
- Root-p shows the most stable and consistent results based on standard deviation of rewards in the Q-Learning agent and One-pass is the most stable CL method in context of evaluation of DQN agent performance which confirms the ideas of Narvekar et al. [23] which states that Pre-defined CL methods need to take bigger sequence of actions to converge.
- Finally, the Anti-curriculum method, with the highest SES, would be optimal for training environments where safety and reliability are prioritized. This finding can be explained in the way that Anti-curriculum implies less exploration at the beginning in simpler environments to satisfy progression of rewards retrieval.

3.2. MountainCar: experimental results

In this setting, a set of metrics was not enhanced with additional measures that provide insight into each agent's performance, specifically under the curriculum that best facilitates its training. Convergence speed metric represents the first instance of achieving a reward indicative of optimal agent performance. In this environment, convergence speed is measured by the number of evaluations required for the agent to achieve a mean reward of -110.0 in reward distribution during evaluation. Given that the convergence speed condition is very strict and highly complex to achieve due to the sparsity of the rewards in MountainCar environment and the defined RL

[22] Taylor M., Stone P. Transfer Learning for Reinforcement Learning Domains: A Survey. *Journal of Machine Learning Research 10*. 2009. P. 1–53. URL: https://dl.acm.org/doi/10.5555/1577069.1755839 (date of access: 23.02.2024).

[23] Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey / S. Narvekar et al. *Journal of Machine Learning Research 21(181)*. 2020. P. 1–50. URL: https://doi.org/10.48550/arXiv.2003.04960 (date of access: 16.11.2023).

architectures struggle to converge for this task to near optimal or optimal solutions with a mean reward of -110, the focus of the experiments was shifted to evaluating Learning Stability, AAR, and SES metrics to investigate the models' adaptability.

Also, in context of complex environments where reward signal is sparse, there is a best practice to tweak a reward function which helps the agent to learn better by using mutated reward signal. Such a reward shaping approach was designed specifically to solve MountainCar problem:

$$\text{reward shaping}(r, s, s') = r + 1000 \cdot (0.9 \cdot s' - s)$$

where:

- *s* – current state of the agent.
- *s'* – next state of the agent.
- *r* – reward which was gained through transition from *s* to *s'*.

This function adjusts the reward based on the velocity of the car. Specifically, it provides a positive reward proportional to the increase in the car's velocity. This is beneficial because higher velocities are generally associated with more effective swings, which help the car reach the goal faster. This method can be considered safe reward shaping because it adheres to the principle of potential-based reward shaping, which is guaranteed not to alter the optimal policy. According to Ng et al. [24] potential-based reward shaping adds a potential function to the reward, which depends on the current state and the next state. This ensures that the optimal policy remains invariant, and the agent is guided towards beneficial behaviors without introducing unintended biases.

[24] Ng A., Russell S. J., Harada D. Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. *ICML*. 1999. P. 1–10. URL: https://dl.acm.org/doi/10.5555/645528.657613 (date of access: 03.06.2024).

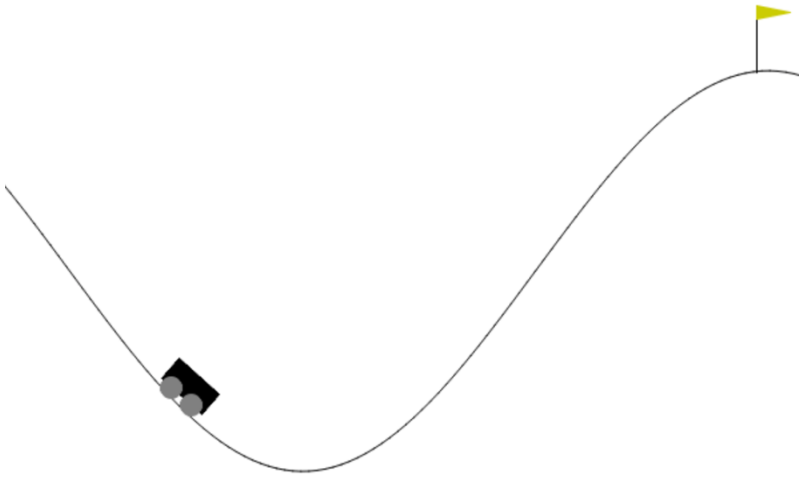


Fig 2: Visualization of a single training episode of a pre-trained PPO agent trained on Teacher learning CL in MountainCar environment (created by author)

MountainCar environment, from *Fig 2,* has a few parameters for customization including gravity as one of possible environmental parameters. Gravity adjustment was chosen for current experimental setup. A lower gravity enables the car to ride faster and explore more environmental states in a shorter period. Conversely, as the gravity increases, the environment and the task become more complex. Here is an example of Linear CL method:

$$\text{gravity}_{new}\,(\text{ep}) \;=\; \text{gravity}_{min} \;+\; (\text{gravity}_{max} - \text{gravity}_{min}) \times \frac{\text{ep}}{total\ eps}$$

where:

- *gravity* – CL parameter which relates to gravitational power: *max* corresponds to maximum possible gravity in experimental setup, set to 0.00025, and *min* corresponds to minimum one, set to 0.0025, meanwhile *new* is bounded by the range and corresponds to value which should be updated for the next episode.
- *ep* – current training episode on which update to gravity is done.
- *total eps* – amount of training episodes which must be done by the agent (a hyperparameter).

After defining a unified CL strategy based on gravity environmental parameter for all CL methods to tweak task complexity, let's proceed to analyze each agent and CL methods individually.

3.2.1. Q-Learning

To solve a challenging MountainCar problem for pre-defined RL architectures, this environment requires specific adjustments to the hyperparameters to ensure effective learning. The configuration used is detailed below:

- *Total Episodes*: The agent undergoes training for 3,000 episodes, tailored to the complexity and requirements of the MountainCar environment.
- *Alpha (α):* The learning rate remains at 0.1, suitable for balancing the acquisition and retention of knowledge.
- *Gamma (γ):* The discount factor is elevated to 0.99, placing a high emphasis on future rewards, which is crucial for success in this task.
- *Initial Epsilon (ϵ):* Set at 1.0 to encourage exploration, which is vital in discovering successful strategies in the MountainCar's diverse state space.
- *Minimum Epsilon*: The epsilon value declines to a minimum of 0.01, ensuring that the agent continues to explore at a low level even as it learns to exploit its environment.
- *Epsilon Decay:* The decay rate for epsilon is set to 0.99999, promoting a very gradual reduction in exploration as learning progresses.

Following the collection of 1212 evaluations from experiments on a Q-Learning agent, the following observations have been systematically recorded in *Table 3*.

Table 3

Q-Learning	Metrics		
	AAR	SES	Learning Stability
Anti-curriculum	-20.411	**1.0**	373.058
Baseline	**-18.548**	**1.0**	380.557
Hard	-166.498	0.9996	397.063
Linear	-167.975	0.9991	433.567
Logarithmic	-168.603	0.9993	443.969
Logistic	-158.588	0.9994	466.445
Mixture	-168.533	0.9995	395.380
One-pass	-165.358	0.9996	**262.948**
Polynomial	-166.387	0.9997	312.050
Root-p	-37.631	0.9999	343.676
Teacher learning	-19.230	**1.0**	284.948
Transfer learning	-170.412	0.9995	453.497

Table 3: Comparative Analysis of CL methods performance metrics based on Q-Learning agent performance in MountainCar. The table displays AAR, SES, and Learning Stability for distinct CL strategies. Highlighted values indicate the highest performance in each metric, illustrating the strengths of each configuration in adaptability and exploration.

The following observations were derived from *Table 3*:

- *AAR*: Despite overall negative values, the least negative AAR is observed in the Baseline method (-18.548), indicating its relatively better performance in decision-making under the defined setup. Following closely are the Teacher learning (-19.230) and Anti-curriculum (-20.411) methods, which also exhibit relatively less negative AAR values compared to others.

- *SES*: Baseline, Anti-curriculum, and Teacher learning methods display very high safe exploration scores (1.0), suggesting that these methods are more cautious, minimizing risks during exploration. The Root-p method also shows a high SES score (0.9999), indicating a strong focus on safe exploration.
- *Learning Stability*: The Logistic method shows the highest Learning Stability metric (466.445), indicating a more explorative and possibly less consistent learning, which might be suitable for environments that require adaptive learning strategies. Conversely, the One-pass method yields the highest stability level (262.948) across the other algorithms, indicating more consistent performance during training.

3.2.2. DQN

The hyperparameters for DQN agent to solve MountainCar problem are described as follows:

- *Total Episodes:* The training extends over 500 episodes, optimizing for swift convergence and computational efficiency.
- *Alpha (α):* The learning rate is fine-tuned to 0.0001, ensuring smooth updates that help in policy stability.
- *Gamma (γ):* A high discount factor of 0.99 supports the agent's focus on long-term rewards.
- *Replay Buffer Size*: Set at 10000, this large buffer allows for effective learning from past experiences.
- *Batch Size*: Utilizing batches of 128, balancing the training speed and memory usage efficiently.

Following the collection of 1212 evaluation episodes from experiments on a DQN agent across different CL configurations, the following observations have been systematically recorded in *Table 4*.

Table 4

DQN	Metrics		
	AAR	SES	Learning Stability
Anti-curriculum	-32.81	**1.00**	116.99
Baseline	**-16.34**	**1.00**	302.12
Hard	-66.00	0.99	234.70
Linear	-36.86	**1.00**	131.89
Logarithmic	-20.13	**1.00**	153.42
Logistic	-48.94	**1.00**	134.76
Mixture	-69.28	0.99	225.66
One-pass	-36.82	**1.00**	142.64
Polynomial	-58.08	**1.00**	122.45
Root-p	-34.00	**1.00**	**103.50**
Teacher learning	-957.09	**1.00**	120.05
Transfer learning	-31.11	**1.00**	221.41

Table 4: Comparative Analysis of CL methods performance metrics based on DQN agent performance in MountainCar environment. The table displays the AAR, SES, and Learning Stability for various CL strategies. Highlighted values indicate the highest performance in each metric, illustrating the strengths of each configuration in adaptability and exploration.

Here is a summary based on experiments from *Table 4*:

- *AAR*: Despite overall negative values, the Baseline method demonstrates the least negative AAR (-16.34), suggesting its relatively superior decision-making capability under the defined setup. Following this are Logarithmic (-20.13), Transfer learning (-31.11), and Anti-curriculum methods, which also show less negative AAR values compared to others, indicating their effectiveness in adapting to the environment.
- *SES*: All methods except for Hard and Mixture display the maximum safe exploration score (1.00), illustrating a more cautious approach that minimizes

risks during exploration. This high SES indicates a strong emphasis on safe exploration across these curriculum strategies.

- *Learning Stability*: Roo-p method exhibits the lowest Learning Stability (103.50), which indicates a less volatile, more consistent reward signal during training. This could be particularly beneficial in environments requiring reliable adaptation strategies. Conversely, Baseline, with the highest learning stability (302.12), suggests more explorative behavior and potentially less consistent performance.

3.2.3. PPO

The hyperparameters for current agent to solve MountainCar problem could be summarized as follows:

- *Total Episodes:* The training extends over 250 episodes, optimizing for swift convergence and computational efficiency.
- *Alpha (α):* The learning rate is fine-tuned to 0.001, ensuring gradual updates that help in policy stability.
- *Gamma (γ):* A high discount factor of 0.99 supports the agent's focus on long-term rewards.
- *Replay Buffer Size*: Set at 2000, this large buffer allows for effective learning from past experiences.
- *Batch Size*: Utilizing batches of 128, balancing the training speed and memory usage efficiently.

Following the collection of 312 estimations from experiments on a PPO agent across different CL configurations, the following observations have been systematically recorded in *Table 5*.

Table 5

PPO	Metrics				
	AAR	Learning Stability	Mean Reward	Std Reward	Max Reward
Anti-curriculum	-53.21	349.65	-1000.0	0.0	-1000.0
Baseline	-20.00	348.22	**-998.25**	5.25	**-964.3**
Hard	-68.33	**285.80**	-1000.0	0.0	-1000.0
Linear	-49.72	323.57	-1000.0	0.0	-1000.0
Logarithmic	-32.04	330.44	-1000.0	0.0	-1000.0
Logistic	-63.72	319.66	-1000.0	0.0	-1000.0
Mixture	-62.03	347.93	-1000.0	0.0	-1000.0
One-pass	-48.00	324.76	-1000.0	0.0	-1000.0
Polynomial	-76.35	297.00	-998.68	**3.96**	-977.8
Root-p	-35.14	328.70	-1000.0	0.0	-1000.0
Teacher learning	**3.09**	468.39	-1000.0	0.0	-1000.0
Transfer learning	-35.16	319.40	-1000.0	0.0	-1000.0

Table 5: Comparative analysis of CL methods based on PPO agent performance in MountainCar environment. The table displays the AAR, Learning Stability, Mean Reward, Standard Deviation of Reward (Std Reward), Maximum Reward for distinct CL strategies. Highlighted values indicate the highest performance in each metric, illustrating the efficacy and particular strengths of each configuration in specific aspects of learning and exploration.

The following conclusions of the CL comparison is derived from *Table 5*:

- *AAR*: The least negative AAR is observed in the Teacher learning method at 3.09, suggesting it performs relatively better in decision-making under the defined setup. The Baseline method also shows less negative AAR at -20.00, indicating a decent performance.
- *Learning Stability*: Teacher learning method shows the highest Learning Stability metric at 468.39, which could indicate a more explorative and possibly

less consistent reward signal, making it suitable for environments that require adaptive learning strategies. Meanwhile, Hard method displays the lowest Learning Stability metric (285.00), suggesting robustness in training.

- *Mean Reward:* While all methods show negative mean rewards, Baseline method has the least negative mean reward at -998.25, suggesting better performance in minimizing losses in this challenging environment. Another method which succeeded in this environment is Polynomial SPL method with mean reward at -998.68.
- *Std Reward:* Baseline method demonstrates a higher standard deviation of rewards at 5.25, indicating less predictable and stable performance during evaluations than in Polynomial method with a standard deviation of rewards at 3.96.
- *Max Reward:* Baseline method achieves the highest evaluation reward at -964.3, indicating its potential for achieving the best outcomes.

Analysis of CL methods in MountainCar environment

A combination of experimental observations from Table 3, Table 4, and Table 5 yields the following set of insights:

- Baseline method demonstrates the most robust performance in terms of high AAR and maximum rewards per evaluation episodes in PPO and Q-Learning setups. This led to an insight that CL tuning of the training environment did not improve speed of convergence given such a CL parameter like gravity in a MountainCar environment. The only CL method which prevailed during evaluation is Polynomial which corresponds to the views of Abel et al. [25], who suggest that SPL methods facilitate more informative experiences during training.

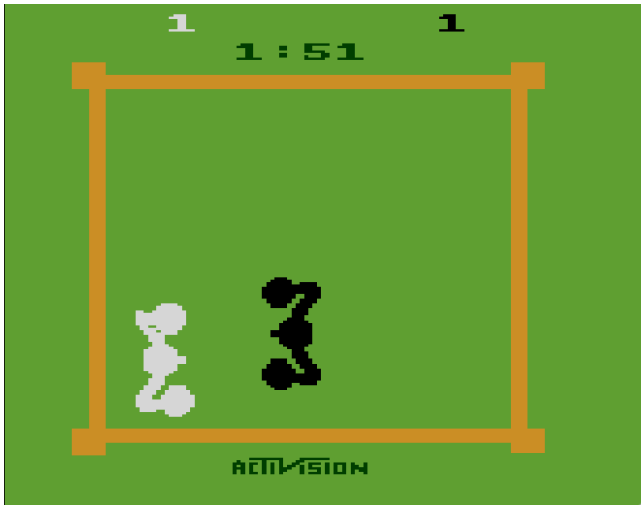
[25] Exploratory Gradient Boosting for Reinforcement Learning in Complex Domains / D. Abel et al. 2016. P. 1–8. URL: https://doi.org/10.48550/arXiv.1603.04119 (date of access: 06.02.2024).

- Root-p and One-pass CL methods display the most stable and consistent results based on the standard deviation of evaluation rewards in the Q-Learning and DQN setups. These findings align with Narvekar et al. [26] point of view, who agreed on ability of Pre-defined CL methods to converge with stability. Another interesting point is that SPL methods like Polynomial can be also classified as methods with high learning stability which is a new founding in this research. This behaviour can be explained as such that environments with sparse rewards like MountainCar require more aggressive CL strategy to make more explorations at the beginning of training.

3.3. Boxing: experimental results

In this experiment, a series of core metrics, augmented with additional measures, provides insights into the performance of each RL agent individually, particularly within the curriculum that most effectively supports its training:

- *Convergence Speed:* this metric denotes the point at which the agent first achieves a reward indicative of optimal performance. For this environment, convergence speed is gauged by the initial occurrence when the agent attains a mean reward of 0.0 during evaluation.



[26] Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey / S. Narvekar et al. *Journal of Machine Learning Research 21(181)*. 2020. P. 1–50. URL: https://doi.org/10.48550/arXiv.2003.04960 (date of access: 16.11.2023).

Fig 3: Visualization of a single training episode of a pre-trained PPO agent trained on Teacher learning CL in Atari games Boxing environment (created by author)

The Boxing environment, from *Fig 3*, has the only one adjustable environmental parameter – frame skip frequency. A lower frame skip frequency enables the agent to make decisions more quickly and explore more environmental states in a short time. Conversely, as the skip frame frequency increases, the environment and the task become more complex. Here is an example of the Linear CL method:

$$\text{frame}_{new}\,(\text{ep}) \;=\; \text{frame}_{min} \;+\; (\text{frame}_{max} - \text{frame}_{min}) \;\times \frac{\text{ep}}{total\ eps}$$

where:

- *frame* – CL parameter which relates to the number of frames that agent skips to read the next signal from environment: *max* corresponds to maximum possible number of frames skips in experimental setup, set to 10, and *min* corresponds to minimum one, set to 1, meanwhile *new* is bounded by the range and corresponds to the value which for the next episode.
- *ep* – current training episode on which update to frame skip parameter is done.
- *total eps* – amount of training episodes which must be done by the agent (a hyperparameter).

Upon establishing a comprehensive set of evaluation metrics and curriculum strategy based on skip frame frequency on which the agent consumes signals from environment, a detailed analysis of each agent can be conducted individually.

3.3.1. DQN

Agent is optimized for robust performance in RL tasks. The hyperparameters were meticulously tuned for the Boxing environment, summarized as follows:

- *Total Episodes:* The training extends over 100 episodes, optimizing for swift convergence and computational efficiency.

- *Alpha (α):* The learning rate is fine-tuned to 0.002, ensuring gradual updates that help in policy stability.
- *Gamma (γ):* A high discount factor of 0.99 supports the agent's focus on long-term rewards.
- *Initial Epsilon (ϵ):* Set at 1.0 to encourage exploration, which is vital in discovering successful strategies in the MountainCar's diverse state space.
- *Minimum Epsilon*: The epsilon value declines to a minimum of 0.01, ensuring that the agent continues to explore at a low level even as it learns to exploit its environment.
- *Epsilon Decay:* The decay rate for epsilon is set to 0.995, promoting a very gradual reduction in exploration as learning progresses.
- *Replay Buffer Size*: Set at 100000, this large buffer allows for effective learning from past experiences.
- *Batch Size*: Utilizing batches of 256, balancing the training speed and memory usage efficiently.

Following the collection of 252 evaluation episodes from experiments on a DQN agent across different CL configurations, the following observations have been systematically recorded in *Table 6*.

Table 6

DQN	Metrics						
	AAR	SES	Learning Stability	Mean Reward	Std Reward	Max Reward	Convergence Speed
Anti-curriculum	-0.00094	0.99922	6.62	-6.90	5.97	1.2	**0**
Baseline	-0.00087	0.99967	7.83	-8.70	8.20	0.8	8
Hard	-0.00086	0.99761	6.20	-6.82	7.27	-3.0	N/A
Linear	-0.00555	0.99950	16.37	-28.90	14.85	1.0	**0**
Logarithmic	-0.00421	0.99915	15.09	-27.31	11.19	-7.2	N/A
Logistic	-0.00280	**0.99999**	9.29	-20.82	8.15	-11.6	N/A
Mixture	-0.00116	0.99955	**5.56**	-9.70	4.81	-2.8	N/A
One-pass	-0.00134	0.99968	6.49	-5.50	**4.58**	-2.2	N/A
Polynomial	-0.00231	0.99932	10.99	-14.80	10.49	-7.6	N/A
Root-p	-0.00120	0.99870	7.64	-5.75	6.33	3.4	**0**
Teacher learning	-0.00135	0.99840	9.78	-9.21	7.33	-0.6	N/A
Transfer learning	**-0.00015**	0.99888	6.03	**-1.88**	6.11	**8.8**	2

Table 6: Comparative analysis of CL methods based on DQN agent performance for Atari games Boxing environment. The table displays the AAR, SES, Learning Stability, Mean Reward, Standard Deviation of Reward (Std Reward), Maximum Reward, and Convergence Speed for distinct CL methods measured during DQN agent learning. Highlighted values indicate the highest performance in each metric, illustrating the efficacy and particular strengths of each configuration in specific aspects of learning and exploration.

Following the results from *Table 6,* here is an analysis of each metric and corresponding CL methods which prevailed in context of the metric:

- *AAR*: Despite overall negative values, the least negative AAR is observed in the Transfer learning method (-0.00015), indicating its relatively better performance in decision-making under the defined setup.
- *SES*: Baseline (0.99967), Anti-curriculum (0.99922), and Logistic (0.99999) methods display very high SES, suggesting that these methods are more cautious, minimizing risks during exploration.
- *Learning Stability:* Linear method shows the highest Learning Stability metric (16.37), indicating a more explorative strategy, which might be suitable for environments that require adaptive learning strategies. Conversely, Mixture approach yields the lowest stability level (5.56) among the other algorithms which indicates at its' ability to stabilize training.
- *Mean Reward:* While all methods show negative mean rewards, Transfer learning has the least negative mean reward (-1.88), suggesting the best performance in minimizing losses in this challenging environment.
- *Std Reward:* One-pass (4.58) and Mixture (4.81) methods demonstrate lower standard deviations in rewards, indicating more predictable and stable performance during evaluations.
- *Max Reward:* Transfer learning achieves the highest maximum reward (8.8), suggesting potential for achieving the best outcomes in combination with its' low mean reward underscoring stability.
- *Convergence Speed:* Anti-curriculum, Linear, and Root-p methods converge the fastest (0), indicating their ability to quickly adapt to the environment.

3.3.2. PPO

Agent is optimized for robust performance in RL tasks. The hyperparameters were meticulously tuned for the Boxing environment, summarized as follows:

- *Total Episodes:* The training extends over 50 episodes, optimizing for swift convergence and computational efficiency.

- *Alpha (α):* The learning rate is fine-tuned to 0.002, ensuring gradual updates that help in policy stability.
- *Gamma (γ):* A high discount factor of 0.99 supports the agent's focus on long-term rewards.
- *Replay Buffer Size*: Set at 100000, this large buffer allows for effective learning from past experiences.
- *Batch Size*: Utilizing batches of 256, balancing the training speed and memory usage efficiently.

Following the collection of 132 evaluation measurements from experiments on a PPO agent across different CL configurations, the following observations have been systematically recorded in *Table 7*.

Table 7

PPO	Metrics					
	AAR	Learning Stability	Mean Reward	Std Reward	Max Reward	Convergence Speed
Mixture	-0.000089	5.81	2.02	2.36	**6.4**	**0**
Anti-curriculum	**0.000122**	5.05	0.35	2.94	**6.4**	**0**
Teacher learning	-0.000300	6.48	-0.53	2.85	5.2	2
Baseline	0.000104	4.49	-0.11	1.90	3.6	3
Logistic	-0.000318	4.60	-0.80	2.03	3.0	**0**
Root-p	0.000041	4.58	-0.40	1.66	2.6	**0**
Hard	-0.000440	6.39	-2.09	2.84	2.6	6
Transfer learning	-0.000280	4.98	-1.47	2.87	2.2	**0**
Linear	-0.000384	4.88	-0.87	1.73	2.2	2
Logarithmic	-0.000163	**3.98**	-0.49	**1.36**	1.2	5
Polynomial	-0.000471	4.68	-3.22	2.16	0.6	3
One-pass	-0.000667	4.58	-5.56	2.64	-1.8	N/A

Table 7: Comparative analysis of CL methods based on PPO agent performance in the Atari games Boxing environment. The table displays the AAR, SES, Learning Stability, Mean Reward, Standard Deviation of Reward (Std Reward), Maximum Reward, and Convergence Speed for distinct CL methods measured during PPO agent learning. Highlighted values indicate the highest or best performance in each metric, illustrating the efficacy and particular strengths of each curriculum strategy in specific aspects of learning and exploration.

Summarizing the experiments from *Table 7* allowed to build a set of observations:

- *AAR:* Anti-curriculum method has the highest AAR (0.000122), indicating relatively better decision-making performance in this setup. In contrast, methods like One-pass (-0.000667) show the poorest performance.
- *Learning Stability:* Teacher learning shows the highest Learning Stability metric (6.48), suggesting it is better at exploration and signal receiving. Conversely, Logarithmic (3.98) exhibits the lowest Learning Stability indicating a robust stability and ability to learn consistently.
- *Mean Reward:* Mixture method has the highest mean reward (2.02), implying it performs the best in achieving the end goal. One-pass (-5.56) has the lowest mean reward, indicating poor performance of current CL method.
- *Std Reward:* Logarithmic (1.36) and Root-p (0.35) methods have lower standard deviations, suggesting more predictable and stable performance. On the other hand, Anti-curriculum (2.94) shows the highest standard deviation, indicating less predictability.
- *Max Reward:* Anti-curriculum (6.4) and Mixture (6.4) achieve the highest maximum rewards, indicating their potential for achieving the best outcomes.
- *Convergence Speed:* Baseline (3), Logarithmic (5), and Hard (6) methods converge relatively slowly, while others like Mixture, Anti-curriculum, Logistic, and Root-p show faster convergence speed (0), indicating their quick adaptation to the environment.

Analysis of CL methods in Boxing environment

Given a set of experimental results from *Table 6* and *Table 7,* the following conclusions can be derived:

- Transfer learning method in both experimental setups with DQN and PPO agents shows an ability to converge fast and to find performant solutions making it suitable for environments where consistent performance and quick convergence is critical which extends Taylor et al. [27] ideas about improvements of performance Transfer learning CL method not only within simpler tasks like CartPole environment, but with more complex ones like Boxing environment from Atari games. Moreover, this framework can include Anti-curriculum method which exceedingly good exploration safety, performance and convergence speed properties on this environmental configuration.
- One-pass, which belongs to Pre-defined method class, display one of the most stable learning steeps which underlines its strength in cautious exploration and adaptability. This observation can be confirmed via SES metric measurements which in DQN experiments proved to be at the topmost level. This finding aligns with Narvekar et al. [28] thoughts about ability of Pre-defined CL methods to converge with stability.
- On the other hand, most of the SPL methods like Linear, Logarithmic, Logistic, and Polynomial failed to show fast convergence and robust pattern learning due to low deviation of rewards across all experimental results.

Conclusions to chapter 3

In this chapter, a comprehensive series of experiments were conducted across various environments, which were classified based on their complexity, dynamics, and

[27] Taylor M., Stone P. Transfer Learning for Reinforcement Learning Domains: A Survey. *Journal of Machine Learning Research 10*. 2009. P. 1–53. URL: https://dl.acm.org/doi/10.5555/1577069.1755839 (date of access: 23.02.2024).

[28] Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey / S. Narvekar et al. *Journal of Machine Learning Research 21(181)*. 2020. P. 1–50. URL: https://doi.org/10.48550/arXiv.2003.04960 (date of access: 16.11.2023).

the adaptability of different algorithms. The experiments focused on evaluating the performance of CL methods when coupled with various RL agents. To ensure statistical significance and reliability of obtained results, each experimental measurement was repeated five times, and the results were averaged and validated with the industry knowledge obtained in context of RL tasks.

The findings from these experiments serve as a basis for formulating a set of rules for selecting the appropriate CL method depending on the specific characteristics and demands of the environment and the objectives of the RL agent. This chapter effectively bridges the gap between theoretical aspects of CL methods and their practical applicability, providing a structured approach to choosing CL methods that are best suited for enhancing the learning efficiency and effectiveness of RL agents in complex and dynamic settings.

The conclusions derived from these experiments, along with the established rules for method selection, will be elaborated upon in the final conclusions of the research, aiming to offer actionable insights and guidelines for deploying CL methods in varied RL contexts.

CONCLUSIONS

Discussion

In this research various CL methods were systematically evaluated across different simulated environments, including CartPole, MountainCar, and Boxing. Each environment poses unique challenges and requirements, influencing the effectiveness of the CL strategies implemented. Conducted experiments underscore the importance of tailoring the selection of CL methods to the complexity and dynamics of the specific task environment, as well as the adaptability of algorithms to achieve optimal learning outcomes.

Complexity and dynamics of the environment:

- *Simple, deterministic environments (e.g., CartPole):* use SPL CL methods that support rapid adaptation and high performance, such as Polynomial and Logistic methods, which show robust performance in environments with relatively straightforward dynamics.
- *Challenging environments with sparse rewards (e.g., MountainCar):* select methods that are adaptive and demonstrate high values of AAR like the non-CL Baseline method or CL-based Polynomial method. These methods balance exploration and stability, crucial in environments with delayed rewards. If the choice is stability of learning, Pre-defined CL methods should be a preferable solution to fit the task with sparse or delayed rewards.
- *Complex, interactive environments (e.g., Boxing):* prefer methods like Transfer learning and Anti-curriculum, which excel in decision-making efficiency and reward maximization, important in dynamic and interactive settings.

Algorithm adaptability:

- *Rapid convergence:* in time-sensitive or computationally constrained scenarios, prioritize algorithms like Linear and Teacher learning methods, which has demonstrated quick convergence to optimal solutions.

- *Adaptation to negative outcomes*: in environments where negative outcomes are common or particularly punitive, consider SPL CL methods like Logistic or Polynomial with better performance in managing and adapting to these outcomes, such as those with higher AAR.

The insights derived from the analysis provide a framework for selecting CL methods that are best suited to the specific characteristics of the learning environment. This targeted approach not only enhances the effectiveness of the learning process but also ensures that the algorithms are robust and adaptable to the complex nature of real-world tasks.

Future work

To enhance the relevance and impact, there is an important research way to emphasize the scalability of the benchmarks and metrics to other domains beyond the ones tested in the paper. Discussing potential adaptations or extensions of current methodologies could provide a clearer path for future research and application, broadening the scope and utility of the findings.

Also, another research path is to extend the comparison of these CL methods within more complex, real-world scenarios such as autonomous driving, drone navigation, or robotics. These applications present unique challenges like real-time decision-making and interaction with unpredictable environments, which can significantly benefit from refined CL strategies.

At last, the future work should strengthen the theoretical understanding of CL by developing formal models that describe the conditions under which certain CL strategies are guaranteed to succeed or fail. This could involve mathematical analysis and proofs that provide a deeper insight into the mechanics of CL.

REFERENCES

1. Bengio Y. Curriculum learning. *ICML*. 2009. P. 1–8. URL: https://dl.acm.org/doi/10.1145/1553374.1553380 (date of access: 01.10.2023).
2. Wang X., Chen Y., Zhu W. A Survey on Curriculum learning. *IEEE*. 2021. P. 1–22. URL: https://ieeexplore.ieee.org/document/9392296/ (date of access: 12.10.2023).
3. Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey / S. Narvekar et al. *Journal of Machine Learning Research 21(181)*. 2020. P. 1–50. URL: https://doi.org/10.48550/arXiv.2003.04960 (date of access: 16.11.2023).
4. B-Pref: Benchmarking Preference-Based Reinforcement Learning / K. Lee et al. 2021. P. 1–25. URL: https://doi.org/10.48550/arXiv.2111.03026 (date of access: 06.12.2023).
5. Cirik V., Hovy E., Morency L.-P. Visualizing and understanding curriculum learning for long short-term memory networks. 2016. P. 1–7. URL: https://doi.org/10.48550/arXiv.1611.06204 (date of access: 13.12.2023).
6. An Empirical Exploration of Curriculum Learning for Neural Machine Translation / X. Zhang et al. P. 1–16. URL: https://doi.org/10.48550/arXiv.1811.00739 (date of access: 04.01.2024).
7. Hacohen G., Weinshall D. On The Power of Curriculum Learning in Training Deep Networks. *ICML*. 2019. P. 1–13. URL: https://doi.org/10.48550/arXiv.1904.03626 (date of access: 08.01.2024).
8. Exploratory Gradient Boosting for Reinforcement Learning in Complex Domains / D. Abel et al. 2016. P. 1–8.

URL: https://doi.org/10.48550/arXiv.1603.04119 (date of access: 06.02.2024).

9. Taylor M., Stone P. Transfer Learning for Reinforcement Learning Domains: A Survey. *Journal of Machine Learning Research 10*. 2009. P. 1–53. URL: https://dl.acm.org/doi/10.5555/1577069.1755839 (date of access: 23.02.2024).
10. Learning to Learn without Forgetting by Maximizing Transfer and Minimizing Interference / M. Riemer et al. *ICLR*. 2018. P. 1–31. URL: https://doi.org/10.48550/arXiv.1810.11910 (date of access: 03.03.2024).
11. Illuminating Generalization in Deep Reinforcement Learning through Procedural Level Generation / N. Justesen et al. *NeurIPS Deep RL Workshop 2018*. 2018. P. 1–10. URL: https://doi.org/10.48550/arXiv.1806.10729 (date of access: 14.03.2024).
12. Ng A., Russell S. J., Harada D. Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. *ICML*. 1999. P. 1–10. URL: https://dl.acm.org/doi/10.5555/645528.657613 (date of access: 03.06.2024).
13. Paszke A., Gross S., Massa F. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *NeurIPS*. 2019. P. 1–12. URL: https://doi.org/10.48550/arXiv.1912.01703 (date of access: 04.06.2024).
14. Harris C., Millman J., van der Walt S. Array Programming with NumPy. *Nature*. 2020. P. 1–8. URL: https://doi.org/10.48550/arXiv.2006.10256 (date of access: 05.06.2024).
15. OpenAI Gym / G. Brockman et al. 2016. P. 1–3. URL: https://doi.org/10.48550/arXiv.1606.01540 (date of access: 05.06.2024).

16. Deep Reinforcement Learning That Matters / Henderson et al. *Proceedings of the AAAI Conference on Artificial Intelligence*. 2017. P. 1–8. URL: https://doi.org/10.1609/aaai.v32i1.11694 (date of access: 04.06.2024).