Technische Hochschule Ingolstadt

Bachelor Thesis

in the study program
B. Eng. in Autonomous Vehicle Engineering
Faculty of Electrical Engineering and Information Technology

Leveraging Criticality in Reinforcement Learning for Effective Transfer Learning in Autonomous Driving

First, Last name	:	**Daniil Navodey**
Matriculation number	:	00126504
First Examiner	:	Prof. Dr. Lenz Belzner
Second Examiner	:	"Your second examiner here"
External guide	:	"Your external guide here"
Institute	:	THI Ingolstadt

Affidavit

I declare that I have authored the present thesis independently and that I have not used any other sources/resources than the ones declared here. I have explicitly indicated all the material which has been quoted either literally or by consent from the sources used and I have marked verbatim and indirect quotations as such. I also declare that I have not presented the work elsewhere for examination purpose.

Ingolstadt, ______________________

Daniil Navodey

Acknowledgments

"Your acknowledgements here."

Daniil Navodey
Ingolstadt, Germany

Abstract

"Your Abstract here"

Confidentiality clause

"Talk to BMW"

Ingolstadt, ____________________________
(Date)

(Signature)
Daniil Navodey

List of figures

List of tables

Table of contents

1. Introduction

1.1. Motivation

Transportation nowadays has acquired a new round of development with the introduction of autonomous driving technologies based on machine learning (ML), neural networks (NN), and artificial intelligence (AI). This changed how we think about safety, efficiency and accessibility of transport. The key challenge of a self-driving vehicle is the number of decisions, required for an algorithm to be taken in real-time which will cause endangering consequences for drivers, passengers and other traffic participants. Reinforcement Learning (RL) is a promising approach to tackle this challenge because it allows systems to learn by trying different actions and improving over time. However, to train RL sufficiently for real-world driving it needs a lot of time and big data sets for the learning process.
The major drawback of RL is the lack of adaptivity of models to new situations. This requires an additional learning process, which is time-consuming and inefficient. This is where Transfer Learning (TL) can help. TL enables the ability of a system to take what it has learned in one situation and then apply gathered knowledge to a new one, with less effort and time taken. Emphasizing the focus of TL to be trained on critical states of the system - the moments where making the right decision matters for safety and performance - can lead to an exclusive ability to learn faster and perform better under real-world conditions.
This thesis explores the approach to make RL better at knowledge transfer in the field of autonomous driving by applying critical situations to the models' training process. The aim is to improve the performance of models, broadening the variety of challenges that could be handled, and ensuring the effectiveness of learning. By addressing these issues, this research could contribute to the further development of autonomous driving systems by making them more practical and reliable.

1.2. Objective

1.3. Scope

1.4. Commonly used terms

1. **Commonly used term 1**
2. **Commonly used term 2**
3. **Commonly used term 3**

1.5. Thesis-outline

I Chapter **1**: Introduction

II Chapter **2**: Literature Survey

III Chapter **3**: Method

IV Chapter **4**: Experiments, Results and Discussion

V Chapter **5**: Conclusion

2. Literature survey

2.1. Criticality

One small branch that can be applied to any type of learning which is not explored deep enough and not widely applied yet is criticality. The criticality has varios ways to be computed when used in reinforcement learning and autonomous driving. Despite that the definition of criticality is general for all the possible implementation fields: the higher vallue of criticality corresponce to the lower value of safety and higher threat level[1] in a chosen situation.

In a field of autonomous driving the knowledge of criticality, especially the ability of the machine to approximate this value, is crusial for the algorithms that are either already used or are still under development. The ability of models that control driving behavior to process the environment, observations and disturbances to evaluate the measure for the threat level of the situation in a real-time can boost the performance and increase the reliability of autonomous driving technologies.

The policy can be a base to calculate necessary value. The criticality of a state can be accessed throught the comparison of an entropy of the policy's output action distribution captured at a state `s` with the threshold value. In this method the threshold value `t` is the one controlling the number of states to be considered "critical" [2]:

$$C_\pi = \{s \mid \mathcal{H}(\pi(\cdot \mid s)) < t\}$$

Instead of a policy, in actor-critic methods it can be useful to consider the the action-value funciton(Q-function). Mainly the formula decides if the state is critical according to following condition: if the result produced by the random decision-making process is much worse than the result of acting optimally, then the state is critical.

$$C_\pi = \{s \mid \max_a Q^\pi(s,a) - \frac{1}{|A|}\sum_a Q^\pi(s,a) > t\}$$

There are many other numerical methods to define the criticality value to be used in different algorithms and systems, however, for this thesis there was a simpler solution chosen. The reliable indicator of the criticality is a variance of the Q-function [3]. The higher the variance of the Q-function among all pollible actions in a particular state, the more one action/s is probable to be selected by the trained policy than rest. The state, where the probability of one state/s is reasonably higher than others can then be labeled "critical".

Considering the existing research into different methods of calculation, evaluation or approximation of criticality the target of this bachelor work was chosen not to develop a more complex, optimised or effective technique to measure criticaliy, not at all. Current research tries to go deeper into applications of criticality and, more precisely critical states, in transfer learning.

2.2. Transfer Learning

Training a machine learning algorithm of an optimal quality and performance requires big sets of training and evaluation data. Those sets are usually hard to gather or generate, as it costs time and money. The approach to be used as a better means of data collection is Transer Learning. The philosophy of this approach is to utilize models trained for tasks related to the target challenge so that the solution does not need to be developed from scratch [4]. Transfer learning tends help the Machine learning to become more universal and adaptive to new tasks as it creates models which own a collection of knowledge of other models.

The approach of Transfer Learning can easily be found in real life. For instance, in linguistics, if a person has knowledge of one language, it becomes easier for them to learn a new language from the same language group. In the scope of this thesis, transfer learning was performed between two environments: `Highway-Env` and `Merge-Env`.

The necessity and privilege of using Transfer Learning can be seen when applying the approach to tasks related to one another. The autonomous driving task of the agent is to be equally applicable to both environments used throughout the thesis. Moreover, there are some feature sets that those environments share, such as:

- Operating between multiple lanes,
- Surrounding vehicles capable of accelerating/decelerating and changing lanes.

Looking into the scientific field, an analogy can represent the exact approach of Transfer Learning: studying math and physics. If a person who studied math starts deepening their knowledge in physics, this can be called Transfer Learning, as the math expertise makes it easier to acquire new knowledge in the field of physics. However, some themes in math will not be required while learning physics. Compared to the learning process without prior math knowledge, the time and effort investments are noticeably less when Transfer Learning is applied.

2.2.1. Q-Learning

3. Method

The methodology chapter outlines the approaches and tools used in this bachelor thesis. The primary objective of the study is to leverage the criticality in RL with the use of Deep Q-Networks (DQN) to obtain effective transfer learning in conjunction with `Highway-Env` and `Merge-Env` environments. For development and implementation purposes of the RL agent, which will be capable of navigating and interacting with the given environment, the `Stable-Baselines3` framework was used.

3.1. Environments: Highway-Env and Merge-Env

The operation of the reinforcement learning agents in this thesis is realised within two environments: `Highway-Env` [5] and `Merge-Env`. Those are realistic yet controlled simulations of autonomous driving tasks, that provide a setting for evaluating RL algorithms in various driving scenarios. Both environments model traffic behaviour, including surrounding vehicles and multiple lanes, allowing the agent to navigate dynamically and interact with nearby traffic. The environment gives a reward depending on the RL agent's behaviour on a highway; the agent has to avoid crashes while optimizing its speed and lane position.

Another environment was used to imply transfer learning: `Merge-Env`. This is a modification of the `Highway-Env`, where the focus is shifted to merging scenarios between vehicles from on-ramps with the highway traffic. There, the goal of the agent is to make decisions towards ensuring smooth and crashless merging by adjusting the speed and judging traffic gaps, which creates more challenging driving scenarios to overcome. With `Merge-Env`, the controlled vehicle experiences an environment which mostly obeys the rules of `Highway-Env`, but also includes different observations and complexities caused by vehicles from incoming lanes.

Selected environments are implemented based on the Gymnasium API, which is compatible with existing libraries of reinforcement learning implementations like `Stable-Baselines3`.

3.1.1. Action Space

Action Space defines possible actions to be considered in training, decision-making and The `Highway-Env` and `Merge-Env` are implementations where the action space can be chosen to be either Discrete, Continuous, or Discrete Meta-Actions. The continuous action space in these environments means mainly that velocity and steering angle of the agent can take arbitrary values in a predefined range. By controlling velocity and steering angle values, lane change, braking, overtaking, and other actions can be realized throughout the simulated traffic. However, the variety of actions makes the training and decision-making process more complex as the agent can leave the borders of the simulated highway.

The discrete action space is a quantized continuous space, which is represented as a uniformly distributed grid of possible values for throttle and steering angle. Despite the size of the action space being gradually decreased compared to the continuous space, there is a third, simplest controlling action space to be used for this thesis: the Discrete Meta-Actions space.

The Discrete Meta-Action space consists of 5 predefined actions which can replace the continuous action space. These actions are a tolerable replacement of continuous and discrete actions, which make the action space smaller and cover the most demanding actions of the agent. The available actions are [6]:

- `IDLE` – the velocity and steering angle are not adapted compared to the previous state;
- `LANE_LEFT` – the velocity value is saved unchanged, while the agent steers the vehicle to the left, changing the lane completely. In case the agent occupies the far-left lane, this action is not included in the possible action space; if the action is taken despite being unavailable, the `IDLE` action is performed;
- `LANE_RIGHT` – the velocity value is saved unchanged, while the agent steers the vehicle to the right, changing the lane completely. In case the agent occupies the far-right lane, this action is

not included in the possible action space; if the action is taken despite being unavailable, the `IDLE` action is performed;

- `FASTER` – the steering angle remains unchanged, while velocity is increased;
- `SLOWER` – the steering angle remains unchanged, while velocity is decreased.

3.2. Stable-Baselines3

The `Stable-Baselines3` (SB3) is a library that includes some of the most popular, efficient and reliable reinforcement learning algorithms, that can be easily applied and used for different environment types like `Highway-Env` and `Merge-Env`. The list of implemented algorithms consists of Proximal Policy Optimization (PPO), Soft-Actor-Critic (SAC), and the algorithm that was used in this paper, Deep Q-Network (DQN). The privilege of the SB3 is the simplicity of the Application Programming Interface (API), built on top of PyTorch, providing the ease of use of implemented models, requiring only a few lines of code to initialise, train and utilise the RL, which will be able to control the agent based on the learned policy [7]. Moreover, the developed API is not a black-box structure, but has open code, clear user guides and a huge number of active users that published struggles and solutions online, which was crucial during the implication of criticality throughout this thesis paper.

3.3. Deep Q-Network (DQN)

The Reinforcement Learning algorithm used is Deep Q-Network (DQN). This algorithm combines the properties of "reinforcement learning with a class of artificial neural networks" [8]. Following the combination of different reinforcement learning approaches makes it possible for the DQN to handle environments with large state spaces, where the autonomous driving task is one example of such. The interaction with the environment is the main source of experience for the agent to learn an optimal policy. This algorithm was originally developed by Mnih et al. [8], where they introduced it as a method to address the reinforcement learning challenges with high-dimensional state spaces through the use of deep learning, especially to estimate the Q-function throughout the learning process.

The core idea behind DQN is the process of the approximation of the Q-function based on the outputs of the deep neural network, giving an estimated reward based on the given current state and the action performed by the agent in this state. This algorithm performs self-education based on the gained experience and tries to maximize the cumulative rewards by making decisions optimizing its actions. Additionally, there is a feature introduced with the DQN—the replay buffer, which played a crucial role in the implementation of the criticality in my thesis. The replay buffer offers the opportunity to store past experiences in the following form: firstly, the observations of the current state are saved; then the action that was taken by the policy based on the observation of the state; and finally, rewards that were obtained as a result of the action taken in the current state, followed by the observations of the resultant state. With the use of this feature, the experience can be not only saved but also replayed. This innovation assists during the process of approximation of the Q-values, mitigating the issues of overestimation of those, and helping agents to maintain effective and efficient learning in complex environments.

In this study, the DQN algorithm was used to train the agent in `Highway-Env` [5] and `Merge-Env` environments for it to be able to navigate the highway autonomously, maximizing rewards and minimizing penalties, for example, due to collisions. To optimize the training process and to reach better performance, some parameters of the DQN policy were tuned. For instance, hyperparameters, including the learning rate and exploration rate. The exploration rate shows the fraction of the actions where the exploration epsilon-greedy policy will be applied [9]. This means if the exploration rate is p (taking the values from 0 to 1), during $100 \cdot p$ percent of the training iterations (N), the following behavior of decision making will occur: in iteration 1, actions will be taken randomly in 100% of steps; further, up until iteration $N \cdot p$, the proportion of actions taken randomly, not depending on the learned policy, will decrease proportionally, and at the point of iteration $N \cdot p$ it will reach 0%. The efficacy of exploration (random actions) compared to exploitation (following the learned policy) is extremely high at the beginning of the training, where the policy does not act effectively, and exploration helps in gaining various experiences.

The learning rate is another key hyperparameter that determines the training performance of a Deep Q-learning (DQN) model. It defines how much the model parameters are changed during the update after each training step. In the DQN algorithm, the learning rate controls the magnitude of adjustments of the neural network weight, based on gradient descent. During training, the weight update is performed according to the following rule:

$$\theta \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}(\theta)$$

where:

- α is the learning rate,
- $\mathcal{L}(\theta)$ is the loss function (e.g., mean squared error).

Training becomes unstable if the learning rate is too high (α is large) because the model weights change too much, which can lead to oscillations or divergence. If the learning rate is too low (α is small), the learning process becomes slow and the model can get stuck in local minima or take excessive time to reach an acceptable level of performance. In the context of the thesis, using a correct learning rate in the DQN algorithm allows the agent to effectively learn complex scenarios such as highway driving or merging manoeuvres, minimizing errors and providing high-quality results.

3.4. Transfer Learning

The base model for knowledge transfer was trained on the `Highway-Env`. While training the model, the agent was taught to:

- Maintain a safe distance from the vehicle in front,
- Assess the possibility of performing a lane change,
- Execute lane change maneuvers,
- Accelerate or decelerate to maneuver in traffic and receive corresponding rewards,
- Behave in traffic to maximize rewards under particular conditions.

The obtained model had "experience" on the highway but none in merge scenarios. Despite this, the model could make decisions based on previous knowledge, handle operations in pre- and post-merge scenarios at a good level, and avoid collisions during merging, whether by chance or through learned behavior.

The target was, however, to obtain a policy that would operate the agent inside `Merge-Env` efficiently enough for the policy to be called safe. This could have been achieved via direct learning, following the same path as with the highway. While this approach would not require additional implementations, it would be as expensive and time-consuming as training the first model. Since this thesis uses environments that roughly simulate real-life scenarios, the scale of research into reinforcement learning allowed experiments like training the model purely on `Merge-Env` for comparison purposes. In real-life cases, however, such experiments are impossible on a larger scale, which highlights the effectiveness of Transfer Learning.

One approach is to use the pre-trained model for straightforward but less intensive learning. Instead of starting with an empty policy, the policy trained for the highway task is used as the initial policy state for the training process in merging. Because reinforcement learning recursively uses the policy's experience and knowledge, inputting the trained model into the DQN accelerates the model's learning process. Therefore, the second model was trained on `Merge-Env` using 10 times fewer iterations than the initial highway model. This resulted in a model to be validated in future work.

Reducing the number of iterations due to the "experienced" input model provides cost and time advantages. However, the training data used during DQN learning is still random. The approach explored throughout this thesis is to organize the training data more effectively. For this purpose, criticality principles will be used.

3.5. Criticality

The decision making process is highly power consuming for people, because different actions taken in the particular situation could lead to drastically diverging results. Identical principle is to be considered for the decision making performed by the machine. Exceptional meaning to wrongly taken actions should be taken into account during the autonomous driving. Decisions taken by the autonomous vehicle are in the end to be brought to real road scenarios, where lifes of drivers, passengers and other traffic participants are of exceptional value. Thus, a special attention is to be payed to pollible dangerous situation that might occure in varios driving scenarios.

Unfortunately, it is impossible to create a list of all potential situations that can be considered to be unsafe. However there is one approach that can help machine to identify if the state is dangerous, more precisely if any of available actions in the given state can lead to critical situation. The states where one or some of possible actions lead to such situations is to be called critical state [10]. For example, the state shown in the figure 1 can be called critical as ...

!!!!!!!!!!!!!Insert the Figure of the critical state example!!! and explain till the end why this state is called critivcal and which actions will be called critical actions in this scenario!!!

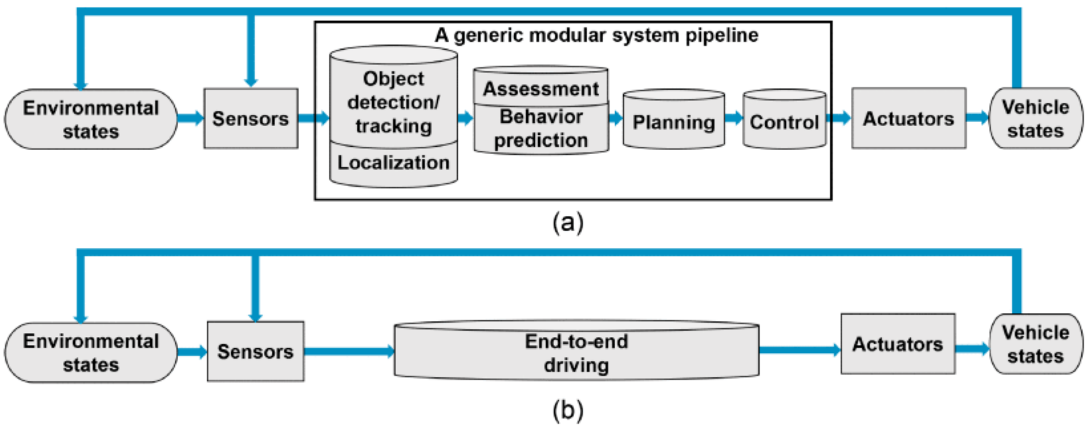

Figure 1: Autonomous driving architecture: a) Modular system b) End-to-end system **[ad_pipeline]**

The crusial role in decision making is played by action-values(Q-values). For a given state of the environment each of possible actions has its own Q-value that was calculated through accumulating the possible prospected reward to be received if the action is taken. The calculations are performed according to the following formula [11]:

$$Q_\pi(s,a) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a, \pi \right]$$

In DQN the approximation of the action-value for every action is performed by the neural network. Then the policy performs the action which has the highest Q-value, according to

$$Q^*(s,a) = \max_\pi Q_\pi(s,a)$$

.

To leverage the criticality in transfer learning there was a complex procedure performed. The base-model, trained purely on the `Highway-Env`, was applied to perform "testdrives" in the `Merge-Env` to collect necessary data for the future applications of transfer learning. The most important data to be collected for criticality purposes was: observations of all the completed steps through out episodes. The model is able to give the Q-net of the observation out, which represents the Q-values for each of 5 possible actions. Based on the maximum action-value among those in Q-net, as described earlier, the algorythm was making decisions about which action to perform for a given observation. However, now, the Q-net was used in a different way: there was a statistical value of variance[12] of Q-values found for the purpose of criticality. The variance is calculated by averaging squares of deviations from the distribution mean [13]:

$$\sigma^2 = \frac{\sum(X-\mu)^2}{N}$$

The purpose of calculating the variance is to find the deviation of values inside of a Q-net, mainly to identify if one of action-values was reasonably higher than all others. Higher variance represents that one of actions was much likely to be taken by the policy than others. The Q-net variances of all the observations recorded should have been further analysed, ...

!!!Describe the python package to find the variance and also describe the python package and the whole method how local peaks were found !!!!

Now when observations of critical states are collected, the next goal is to use them for transfer learning of the initial Highway policy. The ordinary DQN policy is trained in the way, when the desired number of training epochs the algorithm uses initial states of the system. Those are fed into the algorithm, and based on them the policy is trained. It either explores or exploits the initial state and the following states triggered by the action taken by the existing policy. Based on this experience the agent is trained to perform the decision making in favor of one action in every given state, thus maximising rewards and, by that, the driving performance. However the crusial point of initial states ampling for the training is that this is done randomly. Meaning every initial state of the system is randomly generated: the lane of the agent, positions of surrounding vehicles, its' velocities and the velocity of agent car, everything is initialised randomly in every iteration.

The theory of this bachelor paper is, instead of set of randomised states, to feed the DQN algorithm with a set of critical states of a smaller lenth and evaluate if the acceptable performance can be acheived via this method. The set of input states is generated by the DQN interrnally. This is performed by the Replay Buffer.

3.6. Replay Buffer

As critical states are collected the next processing step is feeding this set into the algorithm for purposes of transfer learning. The target is to controll the input of the DQN by replacing random state samples with a list of critical states. This was realised with the use of in-build component of DQN - called Replay Buffer(or Experience Replay).

Whole idea behind the principle of experience replay is grounded from the neuroscience. Observing and exploring rodents, scientists suggest that brain replays sequences of prior experience during its sleeping and awake resting phases [14]. Based on this knowledge DQN was extended by the replay buffer, where the design is following: experience, completed once will not be erased after one iteration, but on the contrary will be stored inside of the replay buffer to be reapplied later [15]. Moreover, it gets more probable for the rare-appearing experience samples to be learned multiple times. Utilising a mix of a recent and older experience stabilises the training process.

Google DeepMind "Prioritized Experience Replay" research towards the optimisation of the usage of replay buffer in the DQN served as an inspiration of an implementation within this thesis. The paper explores one possible improvement of a general replay buffer realisation, where the the DQN will utilise the set of random states in a manner where some states will be prioritized over others. Purpose of the research is brilliantly described in one sentence: "This paper addresses only the latter: making the most effective use of the replay memory for learning, assuming that its contents are outside of our control (but see also Section 6)." [15] After digging into the prioritised replay approach, the concept of overwriting the whole replay buffer data came to mind. Generaly, the intensions that were pursued during developement of replay buffer are to reduce the amount of experience required for the effective learning, whereas this thesis will try to take a full control of experience set and reduce it even further.

List of observations that create the environment of a stateis not all the information required to create a working buffer. Additionally, the action taken under conditions of the state, the received reward and the resultant observation are required. To obtain following data, the pre-trained highway model was used, the same model that collected critical states. Before the start of training process of the transfer model, the experience buffer is filled with all neccessary values and connected to a system to be employed by DQN algorithm.

In the conclusion as the future plans to better an algorithm you can write: Playing with hyperparameters of the Replay buffer: Earlier works investigating replay buffers have often focused on individual hyperparameters, such as the capacity of the replay buffer (Zhang and Sutton, 2017), which has typically been preserved since the seminal work in this field of Mnih et al. (2013; 2015). We begin by identifying that several such hyperparameters, such as buffer capacity and rate of data throughput, are interlinked in the manner that they affect experience replay, modifying both the amount of data available to an agent and the typical age of that data.

4. Experiments, Results and Discussion

5. Conclusion

In the present chapter the results, accomplishments and come shortcomings of the present thesis are summarized in Section 5.1. Some ideas for future research work are highlighted in Section 5.2.

5.1. Summary

5.2. Future prospects

References

[1] Yuanfei Lin and Matthias Althoff. “CommonRoad-CriMe: A Toolbox for Criticality Measures of Autonomous Vehicles”. In: *2023 IEEE Intelligent Vehicles Symposium (IV)*. 2023, pp. 1–8. DOI: 10.1109/IV55152.2023.10186673.

[2] Sandy H. Huang et al. *Establishing Appropriate Trust via Critical States*. 2018. arXiv: 1810.08174 [cs.RO]. URL: https://arxiv.org/abs/1810.08174.

[3] Yitzhak Spielberg and Amos Azaria. “Criticality-Based Advice in Reinforcement Learning (Student Abstract)”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 36.11 (2022), pp. 13057–13058. DOI: 10.1609/aaai.v36i11.21665. URL: https://ojs.aaai.org/index.php/AAAI/article/view/21665.

[4] Asmaul Hosna et al. “Transfer learning: a friendly introduction”. In: *Journal of Big Data* 9.1 (2022), p. 102. DOI: 10.1186/s40537-022-00652-w. URL: https://doi.org/10.1186/s40537-022-00652-w.

[5] Edouard Leurent. *An Environment for Autonomous Driving Decision-Making*. https://github.com/eleurent/highway-env. 2018.

[6] Farama Foundation. *Highway-Env Documentation: Actions*. https://highway-env.farama.org/actions/. Accessed: 2025-01-11. 2025.

[7] Arthur Raffin, Adrian Hill, Alex Gleave, et al. *Stable-Baselines3: Reliable Reinforcement Learning Implementations*. 2021. URL: https://github.com/DLR-RM/stable-baselines3.

[8] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, et al. “Human-level control through deep reinforcement learning”. In: *Nature* 518.7540 (2015), pp. 529–533. DOI: 10.1038/nature14236.

[9] Michael Tokic. “Adaptive ϵ-greedy exploration in reinforcement learning based on value differences”. In: *Proceedings of the 33rd Annual German Conference on Advances in Artificial Intelligence* (2011), pp. 203–210.

[10] Yitzhak Spielberg and Amos Azaria. “The Concept of Criticality in Reinforcement Learning”. In: *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*. 2019, pp. 251–258. DOI: 10.1109/ICTAI.2019.00043.

[11] Yuan Xue, Megha Khosla, and Daniel Kudenko. “Regulating Action Value Estimation in Deep Reinforcement Learning”. In: *Proceedings of the Adaptive and Learning Agents Workshop (ALA 2023)*. London, UK, 2023. URL: https://alaworkshop2023.github.io/papers/ALA2023_paper_62.pdf.

[12] Izumi Karino, Yoshiyuki Ohmura, and Yasuo Kuniyoshi. “Identifying Critical States by the Action-Based Variance of Expected Return”. In: *Artificial Neural Networks and Machine Learning – ICANN 2020*. Springer International Publishing, 2020, 366–378. ISBN: 9783030616090. DOI: 10.1007/978-3-030-61609-0_29. URL: http://dx.doi.org/10.1007/978-3-030-61609-0_29.

[13] Dorothy Musselwhite Thompson and Brian Wesolowski. “Variance”. In: (Jan. 2018). DOI: 10.4135/9781506326139.n737.

[14] David J. Foster and Matthew A. Wilson. “Reverse replay of behavioural sequences in hippocampal place cells during the awake state”. In: *Nature* 440.7084 (2006), pp. 680–683. DOI: 10.1038/nature04587. URL: https://doi.org/10.1038/nature04587.

[15] Tom Schaul et al. *Prioritized Experience Replay*. 2016. arXiv: 1511.05952 [cs.LG]. URL: https://arxiv.org/abs/1511.05952.

A. First appendix

B. Second appendix