https://doi.org/10.31891/csit-2024-3-3
UDC 004.942:004.8

Kyrylo KRASNIKOV
Dniprovskyi state technical university, Kamianske, Ukraine

AGENT Q-LEARNING SPEED DEPENDENCY ON THE FACTORS IN TIC-TAC-TOE

Reinforcement learning of program agents has widespread usage today. Especially, Q-learning, a model-free reinforcement learning technique has shown great results in various applications, like games, self-driving car and robot control. Regarding turn-based games many scientists successfully applied it to train artificial intelligence and to make competitive opponent for human player. While algorithms are well-known there is a room for parametrical optimization to achieve maximum learning speed considering specific problem like turn-based board game. As results of this research show the speed can vary significantly. Tic-Tac-Toe is a very simple and old game that gives an opportunity to try Q-learning without excessive efforts. The algorithm is universal and can be applied to more complex games. It worth noting that core of learning algorithm is the same for any similar game – only rules and board size are changed, which is one of the important properties of Q-learning.

This paper investigates the impact of learning rate and discount factor on the speed of learning of the Q-learning program agent in Tic-Tac-Toe board game. It is conducted a series of experiments using developed computer implementation of the algorithm to analyze the correlation between learning rate, discount factor, and the convergence rate of Q-learning in the specified game. So the experimental field consists of two factors, each of which has three levels, and full factorial experiment has nine combinations of these factors. The learning speed dependency on the each factor is presented. The findings reveal strong relationships between these parameters and convergence speed. For example, the speed is increased proportionally to the both factors, but in the case of discount factor the increase is about 1.4 times lower. Practical significance of the research is in the optimization of factors to achieve effective training of the software agent in order to save processing time, the payment of which is one of the main expenses of enterprises in the field of information technologies. In addition, the research contributes to a better understanding of how Q-learning performs in different game scenarios and provides guidelines for parameter selection in similar applications.

Keywords: Q-learning, learning rate, discount factor, convergence speed, Tic-Tac-Toe.

Кирило КРАСНІКОВ
Дніпровський державний технічний університет, Кам'янське, Україна

ЗАЛЕЖНІСТЬ ШВИДКОСТІ Q-НАВЧАННЯ АГЕНТА ВІД ФАКТОРІВ У ХРЕСТИКАХ-НУЛИКАХ

Машинне навчання програмних агентів з підкріпленням має сьогодні широке застосування. Зокрема, Q-навчання, яке є безмодельною технікою навчання з підкріпленням, показало чудові результати в різних напрямках, таких як ігри, керування самокерованими автомобілями та роботами. Що стосується покрокових ігор, то багато вчених успішно застосовують його для навчання штучного інтелекту або для створення конкурентного суперника гравця-людини. Хоча алгоритми Q-навчання є добре вивченими, існує можливість параметричної оптимізації для досягнення максимальної швидкості навчання, враховуючи специфіку конкретної задачі, такої як покрокова настільна гра. Як показують результати цього дослідження, швидкість навчання може суттєво змінюватися в залежності від факторів. «Хрестики-нулики» – стара і проста гра, яка дає можливість спробувати Q-навчання без надмірних зусиль. Алгоритм є універсальним і може бути використаний до більш складних ігор. Варто зазначити, що ядро алгоритму навчання однакове для будь-якої схожої гри – змінюються лише правила та розмір дошки – що є однією з важливих властивостей Q-навчання.

У статті досліджено вплив швидкості навчання та коефіцієнта запам'ятовування (дисконтування) на швидкість Q-навчання програмного агента для настільної гри «Хрестики-нулики». Проведено серію експериментів з використанням розробленої комп'ютерної реалізації алгоритму для аналізу кореляції між швидкістю навчання, коефіцієнтом запам'ятовування та швидкістю збіжності Q-навчання у вказаній грі. Представлено залежність швидкості навчання від кожного фактора. Результати показують суттєвий зв'язок між цими параметрами та швидкістю збіжності. Наприклад, швидкість збіжності зростає пропорційно до обох факторів, але у випадку фактора запам'ятовування приріст є приблизно в 1,4 рази менший. Практичне значення дослідження полягає в оптимізації факторів для досягнення ефективного навчання програмного агента з метою економії процесорного часу, оплата якого є однією з основних статей витрат підприємств у галузі інформаційних технологій. Крім того, дослідження сприяє кращому розумінню того, як працює Q-навчання в різних ігрових сценаріях, і надає рекомендації щодо вибору параметрів у застосуваннях, подібних до розглянутого.

Ключові слова: Q-навчання, швидкість навчання, коефіцієнт запам'ятовування, швидкість збіжності, хрестики-нулики.

Introduction

Recently the field of reinforcement learning has got advancements, showcased by the effectiveness of the Q-learning algorithm in various applications. It has been widely used to train agents for decision-making tasks. In the context of board games like Tic-Tac-Toe this algorithm realization provides a platform to explore convergence behavior.

In computer realizations Q-learning performance can vary and depends on hyperparameters such as the learning rate and discount factor. The learning rate determines the extent to which the algorithm updates Q-values based on new experiences, while the discount factor balances immediate and future rewards. The interaction between these parameters and their influence on convergence speed remains an active area of investigation.

Literature review

The influence of hyperparameters on Q-learning has been extensively studied in various contexts. Sutton's work on temporal difference learning [1] laid the foundation for Q-learning development. Research by Dayan and Berridge revises the concept of the Pavlovian revaluation experiments [2], demonstrating its potential for model-based evaluation. Also it is considered the learning optimal policies in Markov decision processes.

Numerous studies have explored the impact of the learning rate and discount factor on Q-learning convergence and stability. Mahmood et al. [3] investigated the interaction between these parameters in robotic control tasks, emphasizing the need for adaptive parameter tuning. Squires and Luke [4] proposed a theoretical analysis of Q-learning convergence under various parameter settings.

In game-related domains, Q-learning has been applied to Go [5], and other board games. However, few studies have examined its behavior in Tic-Tac-Toe and Draughts. Notably, Silver et al. [6] demonstrated the potential of Q-learning in the context of larger board games using deep neural networks.

Seijen et al. [7] compared learning speed of online algorithms and regular ones. They presented results that former methods are better than latter. The paper by Mnih et al. [8] presents a groundbreaking approach to reinforcement learning by introducing deep Q-networks (DQNs). The authors demonstrate the use of deep neural networks to approximate Q-values, enabling agents to learn complex policies directly from raw sensory input.

The paper by Szita [9] gives details of reinforcement learning within the context of games. It explores the application of the Cross-Entropy Method to learn policies for playing the game of Tetris. The study focuses on how this method adapts and learns optimal strategies for game playing. So the noisy cross-entropy method has potential in training agents to play complex games efficiently.

The paper [10] presents application of four learning methods to inventory management problem. The comparison shows sufficiently low error of dynamic programming method. Also the speed of learning is optimized for this problem. The author in his paper [11] optimizes learning rate, input data volume, learning momentum and weights in neural network applied to image recognition problem. The authors of the paper [12] investigate application of machine learning to program agents (game characters). Usage of GPUs significantly accelerated the learning. The paper presents results of experiment, which almost 2x speedup.

The related works provide valuable studies and results. However, there is a lack of comprehensive research on the interaction between learning rate, discount factor, and convergence speed in specific game scenarios. This paper aims to bridge this gap by conducting a systematic experimental analysis using computer simulations. We seek to uncover the optimal parameter settings that facilitate rapid and efficient learning in these game scenarios. The results of this study not only contribute to the fundamental understanding of Q-learning but also provide practical insights for parameter tuning in reinforcement learning applications.

Methodology

The methodology consists of equation and algorithm specification, experimental design and evaluation metrics used to assess convergence speed. To train agents for playing the Tic-Tac-Toe we have discrete state and agent`s action spaces. The state space represents the current game board configuration.

At each time step, the agent selects an action based on the current state and updates its Q-values using the Bellman equation:

$$Q(s,a) = (1-\alpha)\cdot Q(s,a) + \alpha \cdot \left(r + \gamma \cdot \max_{a^*} Q(s^*, a^*)\right), \quad (1)$$

where s – the current state, a – the selected action, r – the reward received, s^* – the next state, α – the learning rate (controls the weight of the new information relative to the existing), γ – the discount factor.

The equation and Q-learning algorithm are implemented in desktop program using Dart programming language, leveraging available packages for numerical computations and data manipulation. The implementation uses standard libraries for random number generation and game mechanics. The algorithm was structured to accommodate the discrete state and action spaces of any turn-based board game. The implementation involves to the following steps:

1. Each game state is represented as a unique identifier, capturing the configuration of the game board. The available actions at each state are defined, allowing the agent to select a move.

2. During initialization the Q-values for all state-action pairs have zeros. These Q-values are stored in a key-value map data structure for fast access and updates during the learning process.

3. At each time step, the agent decides whether to explore new actions or exploit its current knowledge. We use the ε-greedy policy, which instructs the agent to select the action with the highest Q-value with a probability of (1 – ε) and explores a random action with a constant probability of ε (15%).

4. After taking an action, the agent receives a reward based on the game outcome. For Tic-Tac-Toe, the rewards are +1 for a win, –1 for a loss, and 0 for a draw.

5. Using the Bellman equation, the Q-values of the selected state-action pair are updated based on the received reward and the estimated maximum Q-value of the next state.

6. At the end of each episode (10 thousands of training games), the agent performance is evaluated (percentage of wins and states exploration), and output is generated for further analysis (charts, table).

The computer implementation was designed to be modular and adaptable to different games with discrete state and action spaces. The implemented software for smartphones or tablets is freely available to download from the Google Play website [13]. Also there is an alternative in-browser version [14]. The software comes with trained agents, which can play on the highest master level (nevertheless, there is always a tiny chance to win). Also the undo of move is implemented , which could be useful to analyze the match and agent's performance in different board position.

It is designed a full factorial experiment to investigate the impact of learning rate (α) and discount factor (γ) on the convergence speed of the algorithm in Tic-Tac-Toe. For each game, experiments are conducted with a range of learning rates (e.g., 0.1, 0.3, 0.5) and discount factors (e.g., 0.7, 0.9, 0.99). These ranges are selected as the most popular and interesting according to literature sources. Of course we can investigate additional factor values. Thus, there are 3 levels of each factor and 9 experiments of full factorial experiment. Each experiment was run for a fixed number of episodes, during which the agent played against a random-turning opponent. It is recorded the Q-values and cumulative rewards obtained by the agent in each episode.

To evaluate the convergence speed of the Q-learning algorithm, two main metrics are selected – the learning curve and the convergence time. Learning curve depicts the agent's performance over episodes. It illustrates how the cumulative rewards achieved by the agent change as the learning process progresses. A steeper learning curve indicates faster convergence.

For our case, it is represented by the number of wins in each episode (10 thousands of games). Convergence time is the number of training games required for the agent performance to stabilize and reach a near-optimal level. It represents the point at which the learning curve flattens out, indicating that the agent has learned an effective policy. For epsilon equaled 15%, the convergence time is considered optimal when wins percentage reaches 87%, even though it can get even 90% in the end of training.

In addition to these metrics, also it is analyzed the behavior of Q-values over episodes to understand how different learning rates and discount factors affect the update process and the rate of convergence. If needed, a new metric can be easily added.

Experiments

The Table 1 presents the full series of the experiments with the results. Each experiment runs a half of million training games starting from empty Q-value table to filled one (when almost all states are fully explored). The table shows an influence of learning rate (α) and discount factor (γ) on convergence speed.

Table 1

Experiment data and results for Tic-Tac-Toe

№	Learning rate (α)	Discount factor (γ)	Convergence time
1	0.1	0.7	~ 160 000 games
2	0.1	0.9	~ 200 000 games
3	0.1	0.99	~ 220 000 games
4	0.3	0.7	~ 110 000 games
5	0.3	0.9	~ 140 000 games
6	0.3	0.99	~ 160 000 games
7	0.5	0.7	~ 160 000 games
8	0.5	0.9	~ 180 000 games
9	0.5	0.99	~ 200 000 games

To see learning process dynamics it is presented three dependency charts (Fig. 1-3) with the steepest learning curve. At the charts a horizontal axis represents a current game from the range 0-500K and a vertical axis – percentage of wins and exploration 0-100%. The black vertical line represents milestone of reaching convergence.

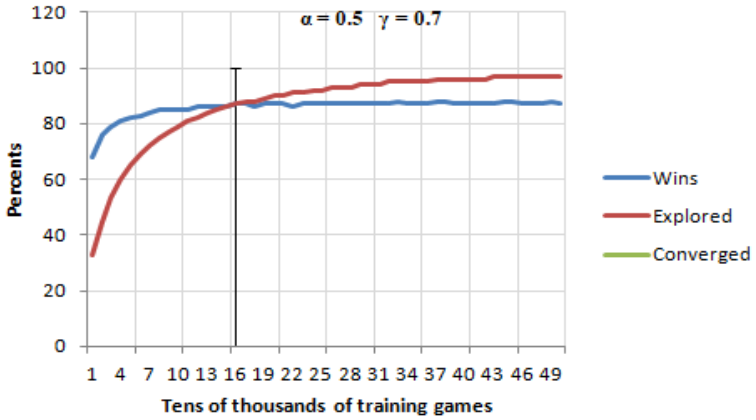

Fig. 1. Learning process illustration for the 7th experiment

Discussion

As seen on the table and figures, the most successful factors combination is 0.3 for learning rate and 0.7 – for discount. However, exploration of states significantly depends on learning rate too and in this combination convergence is reached when only 75% of explored states. This fact needs to be accounted when one chooses optimal combination of factors.

All experiments show a super fast learning at the first 4 thousands of games. This behavior can have relation to a small number of possible states of the game Tic-Tac-Toe, because the game is very simple. In addition it shows effectiveness of Q-learning, so even after 4 thousands of training games agent is able to play on a sufficiently high level. A little jagging in curves is related to a little randomness (15%) of agent moves, so it allows losing but it also allows exploring new states.

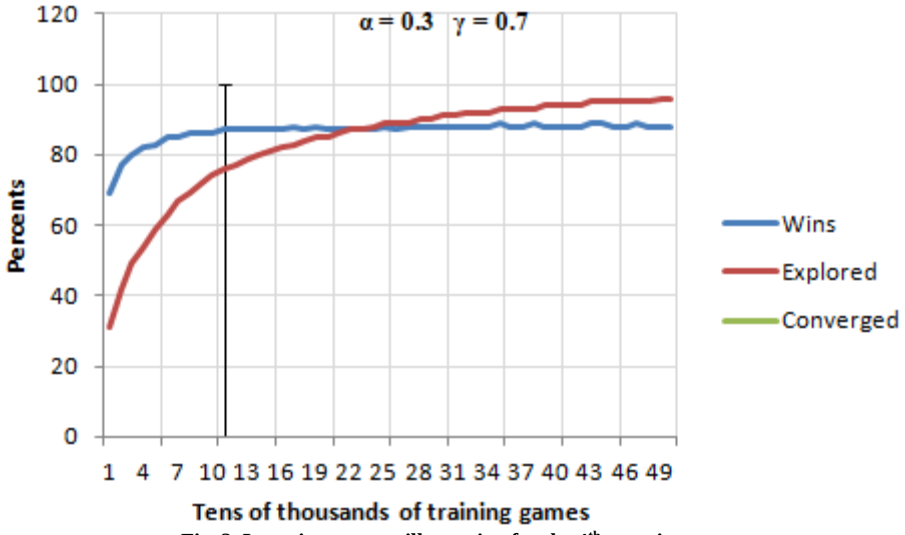

Fig. 2. Learning process illustration for the 4th experiment

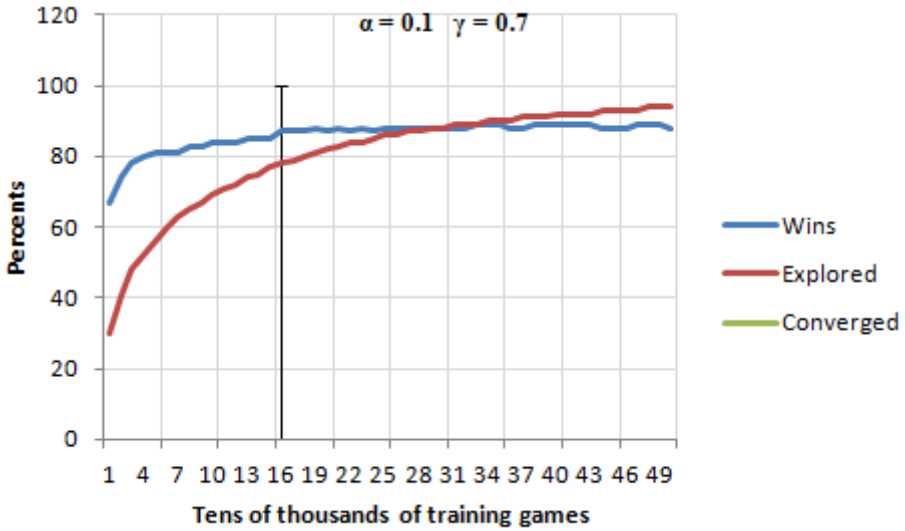

Fig. 3. Learning process illustration for the 1st experiment

The observed trends align with the expectations derived from the theoretical foundations of Q-learning. The results underscore the necessity of further findings of a balance between exploration and exploitation of knowledge.

Conclusions

It is explored the relationship between learning and discount rate, and convergence speed in Tic-Tac-Toe. The complex relationship between them and exploration rate highlights the challenge of hyperparameter tuning. For example, the convergence speed is increased proportionally to the both learning and discount rate, but in the case of discount factor the increase is about 1.4 times lower.

It is found that learning rates significantly influence the agent exploration ability as it can drop by 7-11% when learning rate is halved or quartered. Also the selection of a large learning rate and discount factor is not always

brings the best convergence time. By the way, the low learning rate can ensure more stable convergence but at the cost of slower learning. Therefore, it is optimal to choose a middle value between the two extremes.

The insights gained from this research have practical significance for the application of Q-learning in various domains. Also the findings provide guidelines for parameter selection in similar applications. They can be used to tune learning rates and discount factors to achieve desired convergence speed. This is particularly relevant in scenarios with quick decision-making.

References

1. Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction (2nd ed.). The MIT Press. 548 p. ISBN: 978-0-262-19398-6
2. Dayan, P., Berridge, K.C. (2014) Model-based and model-free Pavlovian reward learning: Revaluation, revision, and revelation. Cogn Affect Behav Neurosci 14, 473–492 (2014). https://doi.org/10.3758/s13415-014-0277-8.
3. Mahmood, A.R., & Sutton, R.S. (2015). Off-policy learning based on weighted importance sampling with linear computational complexity. Conference on Uncertainty in Artificial Intelligence, 27-35.
4. Squires, W., Luke, S. (2020). Scalable Heterogeneous Multiagent Learning from Demon-stration. In: Demazeau, Y., Holvoet, T., Corchado, J., Costantini, S. (eds) Advances in Practical Applications of Agents, Multi-Agent Systems, and Trustworthiness. The PAAMS Collection. PAAMS 2020. Lecture Notes in Computer Science(), vol 12092. Springer, Cham. https://doi.org/10.1007/978-3-030-49778-1_21
5. Silver, D., et al. (2018). A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. Science, 362, 1140-1144. DOI:10.1126/science.aar6404
6. Silver, D., Huang, A., Maddison, C. et al. (2016). Mastering the game of Go with deep neural networks and tree search. Nature 529, 484–489. https://doi.org/10.1038/nature16961
7. Seijen, H., Mahmood, A. R., Pilarski, P. M., Machado, M. C., Sutton, R. S. (2016). True online temporal-difference learning. Journal of Machine Learning Research 17(145):1-40.
8. Mnih, V., et al. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529-533. https://doi.org/10.1038/nature14236
9. Szita, I. (2012). Reinforcement Learning in Games. https://doi.org/10.1007/978-3-642-27645-3_17.
10. Fedorov , E., Nechyporenko, O., Neskorodieva, T., & Leshchenko, M. (2024). Dynamic programming for solve the inventory management problem in logistics. Computer Systems and Information Technologies, (1), 118–126. https://doi.org/10.31891/csit-2024-1-14
11. Radiuk P. (2020). An approach to accelerate the training of convolutional neural networks by tuning the hyperparameters of learning. Computer Systems and Information Technologies, (2), 32–37. https://doi.org/10.31891/CSIT-2020-2-5
12. Hnatchuk, Y. ., Sierhieiev, Y. ., & Hnatchuk, A. . (2021). Using artificial intelligence accelerators to train computer game characters. Computer Systems and Information Technologies, (1), 63–70. https://doi.org/10.31891/CSIT-2021-3-9
13. Mobile application with implemented Q-learnt agent for Tic-Tac-Toe board game. URL: https://play.google.com/store/apps/details?id=com.tensorion.boardgames
14. In-browser web application with implemented Q-learnt agent. URL: https://tensorion.itch.io/board-games

Kyrylo Krasnikov **Кирило Красніков**	PhD, Associate Professor of Systems Software Department, Dniprovskyi State Technical University, Kamianske, Ukraine e-mail: kir_kras@ukr.net https://orcid.org/0000-0002-4241-0572 Scopus Author ID 57205319971 ResearcherID: Q-8835-2017 Google Scholar	кандидат технічних наук, доцент кафедри програмного забезпечення систем, Дніпровський державний технічний університет, Кам'янське, Україна