Duale Hochschule Baden-Württemberg Mannheim

Projektbericht

Reinforcement Learning – Flappy Bird Agent

Studiengang Wirtschaftsinformatik

Studienrichtung Data Science

Verfasser:	Thanadon Chiangkham
Matrikelnummer:	9002977
Kurs:	WWI22DSB
Fach:	Reinforcement Learning
Bearbeitungszeitraum:	19.05. – 31.07.2025

Inhaltsverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

RL - Reinforcement Learning

DQN - Deep Q-Network

1. Einleitung

In diesem Projekt wurde ein Deep-Q-Network-(DQN)-Agent entwickelt, der das Spiel *Flappy Bird* erlernen soll. Reinforcement Learning (RL) basiert auf der Interaktion eines Agenten mit einer Umgebung, wobei er Belohnungen für erfolgreiche Aktionen erhält und seine Strategie iterativ verbessert.

DQN kombiniert Q-Learning mit neuronalen Netzen und eignet sich besonders für Umgebungen mit kontinuierlichen Zustandsräumen (vgl. Mnih et al. 2015, Seiten 529–533).

2. Wahl und Beschreibung der RL-Umgebung

Für das Projekt wurde eine selbst implementierte Flappy-Bird-Umgebung erstellt. Die Wahl dieser Umgebung basiert auf mehreren Gründen:

1. Relevanz und Einfachheit: Flappy Bird ist ein bekanntes, aber dennoch herausforderndes Spiel, das ideal für RL-Experimente geeignet ist, da es nur zwei mögliche Aktionen (Flügelschlag oder keine Aktion) und eine leicht verständliche Spielmechanik hat (vgl. Jander 2022, S. 3).
2. Schnelle Ausführbarkeit: Die Umgebung kann lokal ohne zusätzliche Abhängigkeiten außer Pygame ausgeführt werden, wodurch ein schneller Entwicklungszyklus möglich ist.
3. Persönliche Motivation: Flappy Bird ist ein klassisches RL-Beispiel, das sich gut eignet, um den Lernprozess eines Agenten anschaulich darzustellen.

Die Umgebung wurde in Python mit Pygame selbst implementiert. Sie enthält die Funktionen reset(), step(), get_state() und render(). Der Zustandsvektor umfasst vertikale Position, vertikale Geschwindigkeit, horizontale Distanz zur nächsten Pipe und Differenz zur Lückenmitte – wie standardmäßig in DQN-Umgebungen aus Studien übernommen (vgl. Mnih et al. 2015, S. 529–531)

Als Aktionen stehen Flügelschlag (1) oder Nicht-Fliegen (0) zur Verfügung. Der Reward setzt sich aus +10 Punkten für das Passieren einer Pipe sowie einem Zusatzreward für eine zentrale Flugbahn zusammen. Kollisionen führen zu -50 Punkten und beenden die Episode.

3. Entwicklung des RL-Agenten

Das neuronale Netz des DQN besteht aus zwei verdeckten Schichten mit je 128 Neuronen (ReLU-Aktivierung) und gibt Q-Werte für die zwei möglichen Aktionen aus (vgl. Mnih et al. 2015, S. 529–533). Trainiert wurde mit einem Replay Buffer (100.000 Einträge), Target-Netzwerk-Updates alle 10 Episoden und einem ε-greedy-Policy, wobei ε von 1.0 auf 0.05 abnahm.

Über 780 Episoden wurde der Agent trainiert. Rewards, Scores und ε-Werte wurden aufgezeichnet, um den Lernverlauf zu analysieren

Das Training erfolgt nach dem klassischen DQN-Ansatz:

- Ein Replay Buffer speichert Übergänge (state, action, reward, next_state, done) und ermöglicht ein stabileres Training durch zufälliges Sampling (vgl. Mnih et al. 2015, S. 530).
- Ein Target-Netzwerk wird alle 10 Episoden aktualisiert, um stabile Q-Werte zu gewährleisten.
- Die Aktionsauswahl erfolgt nach dem ε-greedy-Verfahren, wobei ε zu Beginn 1.0 beträgt und mit einem Faktor von 0.995 bis zu einem Minimum von 0.05 reduziert wird.
- Die Q-Werte werden mithilfe der Bellman-Gleichung aktualisiert:

$$V(s) = \max_a (R(s,a) + \gamma V(s'))$$

Abbildung 1: Bellman Equation

4. Bewertung und Visualisierung

Die Visualisierungen zeigen, dass der Agent im Verlauf der Episoden deutliche Fortschritte macht. Die Reward-Kurve steigt mit zunehmendem Training (Abbildung 2), und die gleitende Durchschnittskurve (Abbildung 3) verdeutlicht eine stetige Leistungssteigerung. Das Histogramm (Abbildung 4) zeigt, dass die meisten Episoden Rewards im niedrigen Bereich aufweisen, aber auch hohe Belohnungen erreicht werden.

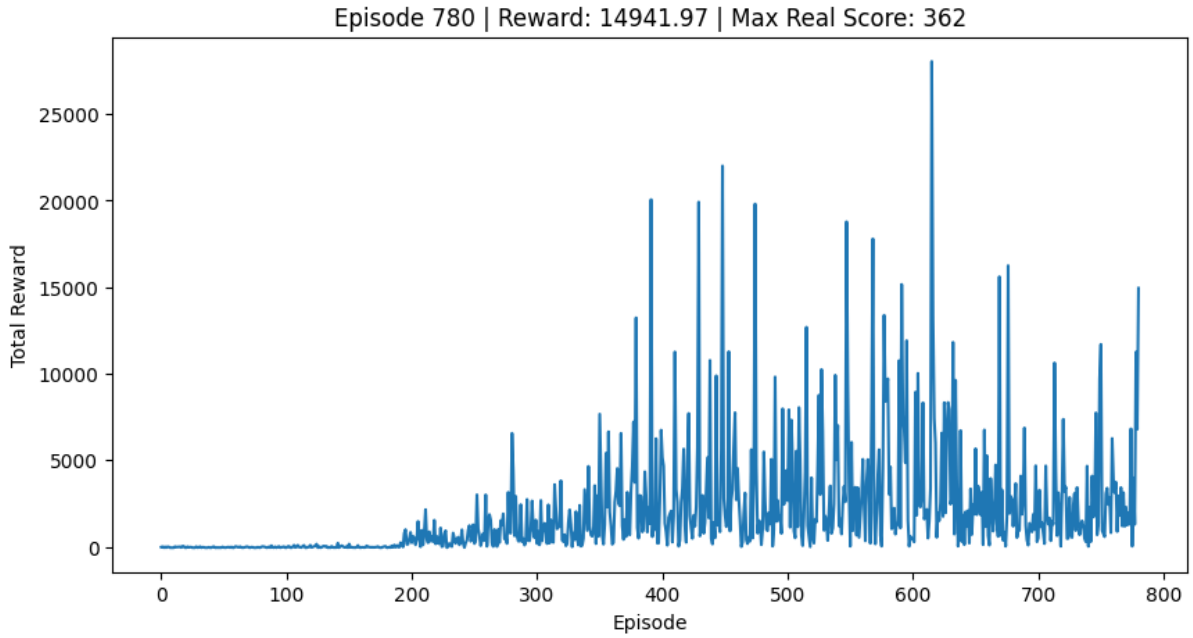

Abbildung 2: Reward-Kurve

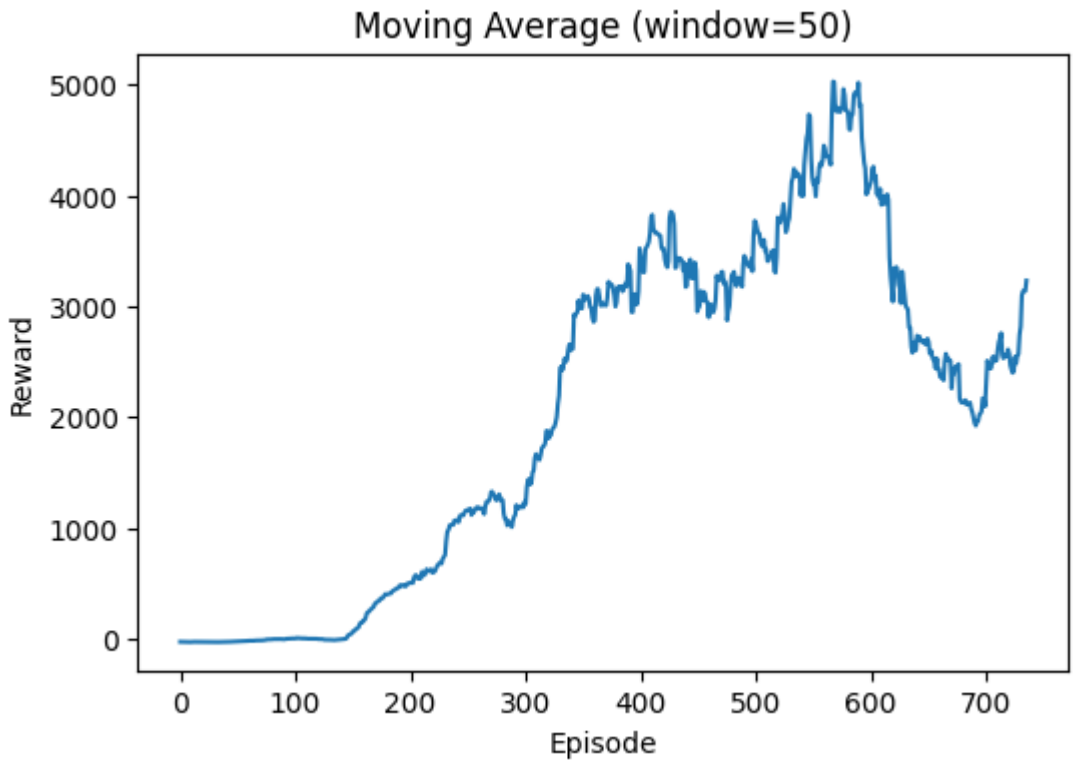

Abbildung 3: Gleitende Durchschnittskurve

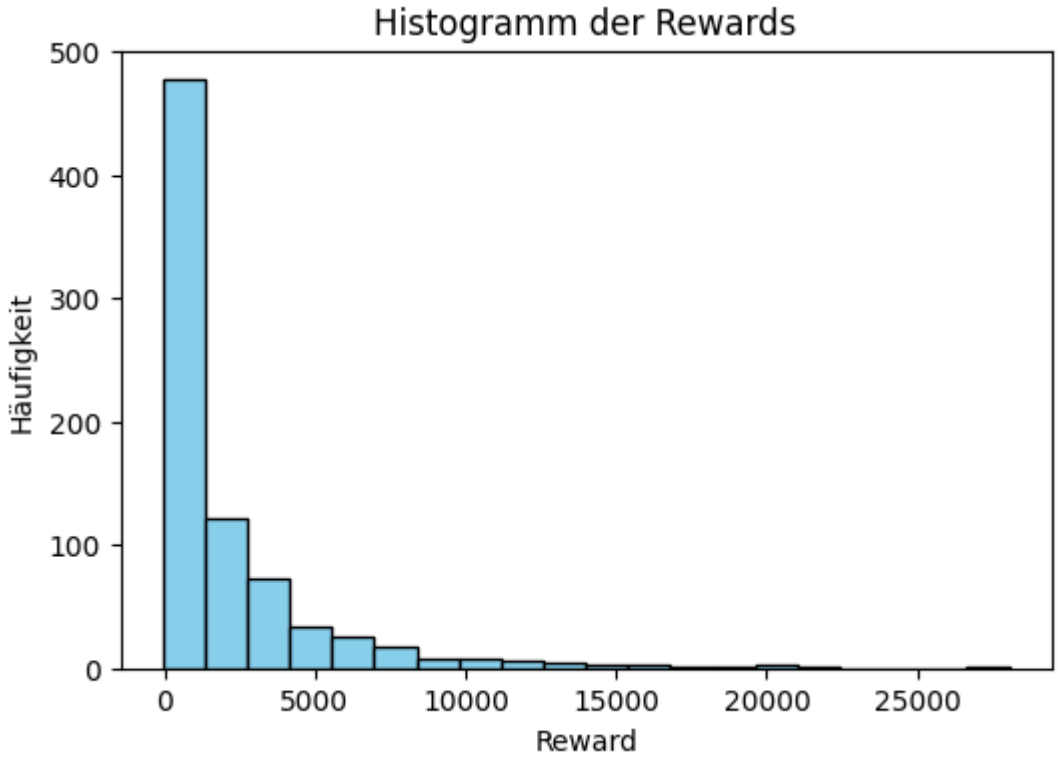

Abbildung 4: Histogramm der Rewards

Abbildung 5 zeigt die Entwicklung von Durchschnitts- und Maximalreward, wobei der maximale Reward im Verlauf deutlich ansteigt. Dies belegt, dass der Agent zunehmend bessere Strategien lernt.

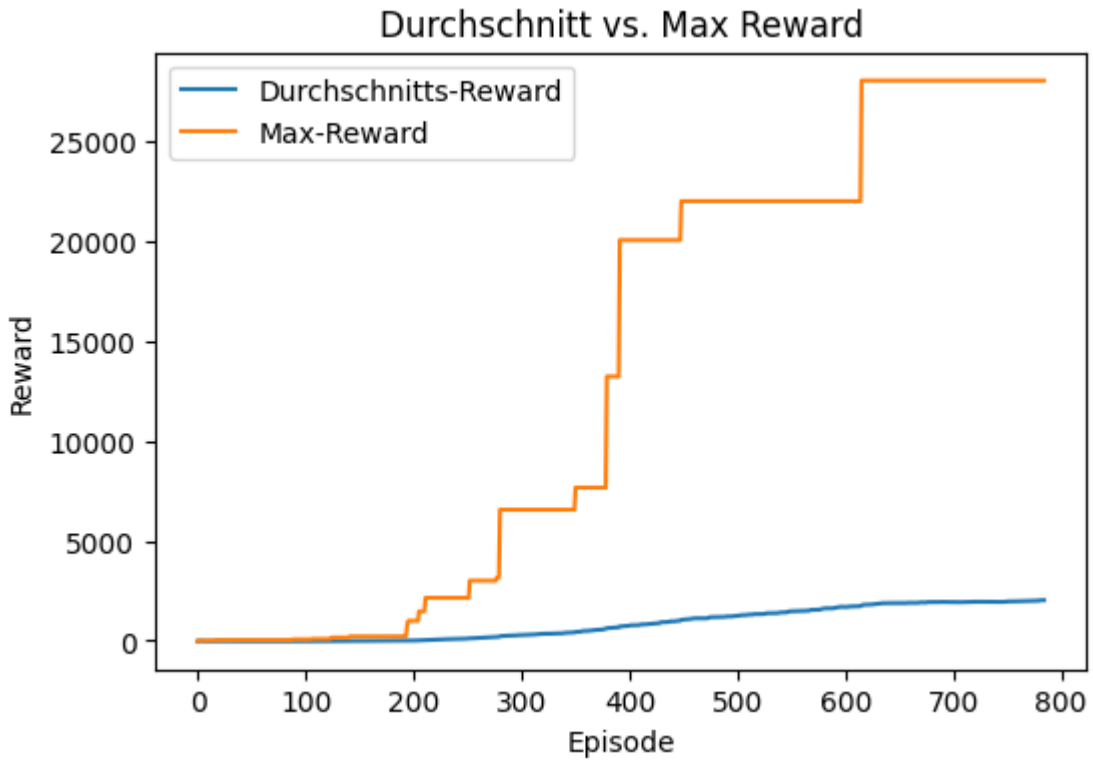

Abbildung 5: Durchschnitts- und Maximalreward

Reflexion

Stärken: Der Agent konnte erfolgreich Strategien entwickeln, um Pipes zu passieren und über mehrere Episoden hinweg zu überleben. Dies zeigt, dass der DQN-Ansatz grundsätzlich in der Lage ist, auch in einer Umgebung mit kontinuierlichen Zustandswerten sinnvolle Handlungsstrategien zu erlernen.

Limitationen: Die Ergebnisse weisen deutliche Schwankungen auf, was typisch für DQN ist. Ursache hierfür sind die hohe Abhängigkeit von Hyperparametern, die Verwendung eines einzigen Q-Netzwerks sowie die mögliche Über- oder Unterschätzung von Q-Werten. Zudem ist die Trainingszeit vergleichsweise hoch, und es besteht die Gefahr von instabilem Lernen, insbesondere bei komplexeren Umgebungen.

Verbesserungspotenzial: Eine höhere Stabilität könnte durch erweiterte Verfahren wie Double-DQN (zur Reduzierung von Q-Wert-Überoptimierungen), Dueling-DQN (zur besseren Trennung von Zustands- und Vorteilsschätzung) oder Prioritized Experience Replay erreicht werden. Außerdem könnten modernere Ansätze wie Rainbow-DQN oder PPO die Lernleistung weiter steigern.

5. Fazit

Das Projekt zeigt, dass ein DQN-Agent erfolgreich lernen kann, das Spiel Flappy Bird zu meistern. Trotz der einfachen Umgebung verdeutlicht es die wesentlichen Konzepte des Reinforcement Learnings: Zustands- und Aktionsräume, Belohnungsfunktion, Q-Learning und Exploration-Exploitation-Tradeoff.

Das Projekt bietet viele Möglichkeiten zur Weiterentwicklung, etwa durch die Nutzung fortschrittlicher RL-Algorithmen oder eine erweiterte Hyperparameter-Optimierung.

Git-Repository Link: Thanadon2001/Flappy-Bird: Reinforcement Learning

Quellen

1. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *nature*, *518*(7540), 529-533.

2. Jander, M. (2024). A Technical Introduction to Reinforcement Learning. *Available at SSRN 4974206*.

3. https://medium.com/@navneetskahlon/the-bellman-equation-decoding-optimal-paths-with-state-action-reward-and-discount-b2dad3c7c11