Reinforcement Learning mit einer veränderten Rewardfunktion

Von: Stefan Manthey
Matrikelnummer: 558040

Inhaltsverzeichnis

1 Verwendete Konzepte

In dem von mir bearbeitetem Projekt für die Veränderung eines Environments für das Spiel „Pong“, um einen veränderten Reward zu nutzen, für einen möglichen schnelleren Lerneffekt, wird der Ansatz des Reinforcment Learnings verwendet. Das neuronale Netz nutzt keine highlevel API, wie z.B. Tensorflow, sondern nutzt hierfür lediglich die Bibliothek „numpy“. Für die Analyse der Situation und die daraus resultierende Aktion der KI, nutzt die Software nicht wie in der wissenschaftlichen Veröffentlichung zum Atari Game „Breakout“, vier Frames für eine Entscheidung, sondern der vergangene Frame wird von dem aktuellen Frame subtrahiert, wodurch nur die Veränderung sichtbar ist. Die Differenz aus den beiden Frames ergibt die Bewegungen im Bild, welche daraufhin an das neuronale Netz weitergegeben wird. Dieses nutzt Forward Propagation, um eine Aktion (Steuerungsaktion) auf die aktuelle Situation zugeben bzw. zu schätzen, welche dann an das Environment weitergegeben wird.

Sowohl das alte Environment, als auch das neue vergeben Ihren Reward bei Änderungen des Spielstandes, das heißt wenn ein Punkt erzielt wird ändert sich der Punktestand, diese Änderung wird daraufhin als Reward an die KI weitergegeben. Beim alten Environment wurde beim erzielen eines Punktes ein Reward von 1, beim verlieren einer Runde -1 und bei keiner Änderung des Spielstandes (laufende Runde) ein Reward von 0 vergeben. Im neuen Pong Environment wird nun ein veränderter Reward verteilt der auf den gleichen Grundsätzen basiert. Für eine gewonnene Runde erhält die KI 10, beim verlieren -5 und wie gehabt, bei unverändertem Spielstand 0 als Reward. Der Lernverlauf des neuronalen Netzes wird mithilfe der Rewardsumme, die bei jeder Episode gebildet wird, in einem Diagramm dargestellt. Eine Episode endet, wenn ein Spieler zuerst 21 Punkte erreicht.

2 Implementierung des neuen Environments

Zuerst wurde eine „setup.py“ im Root-Verzeichnisses erstellt. In diesem wird das Package „atari_games“, welches das neue Environment enthält, angegeben und die Bedingung, dass für die Benutzung des Packages, „gym“ von Open AI installiert sein muss.

In der „_init_.py“ im Package „atari_games“ wird das neue Environment im „gym“ registriert, damit dieses es später aufrufen kann. Auch der Einstiegspunkt, der zu ladenden Klasse, wird hier angegeben.

Im Unterpackage wird in der „_init_py“ die Klasse „PongEnv“ importiert. Dies ist eine Vorgabe von gym und wird benötigt um die Klasse, welches das neue Environment beinhaltet in gym zu importieren.

Für die Funktionsweise des neuen Environments „htw_pong-v0“ möchte ich hier auf die wichtigsten Funktionen der Klasse „PongEnv“ eingehen:

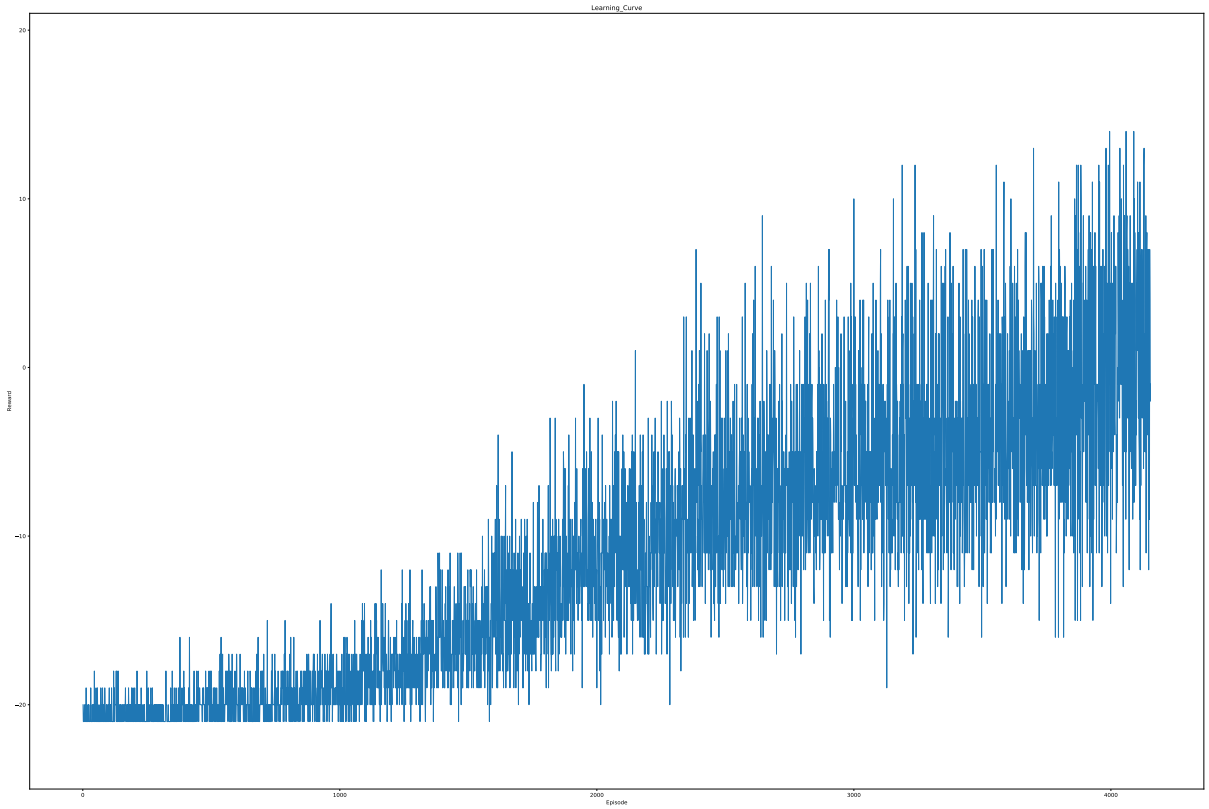
Funktion	Beschreibung
_manipulate_reward(action)	Ist eine interne Funktion die, die eigentliche Aktion (die Bewegung des Paddels) ausführt. Sie bekommt den Reward, der aus dem Spielstand resultiert, und verändert diesen. Anschließend wird der neue Reward zurückgegeben.
_get_obs()	Es handelt sich hierbei, um eine interne Funktion, die den aktuellen Zustand(Frame) aus dem RAM oder dem Bild läd und ihn zurück gibt.
_n_actions()	Funktion bzw. Attribut, dass die Anzahl aller möglichen Aktionen, die im Spiel durchgeführt werden können, angibt.
step(action)	Ruft die internen Funktionen „_get_obs()“ und „_manipulation_reward(action)“ auf, um die Steuerungsaktion auszuführen, den daraus resultierenden Reward und den neuen Zustand (Frame) zu erhalten. Die Funktion liefert den Reward, neuen Zusand, Information ob die Episode zu Ende ist und ob das Spiel noch läuft zurück.
reset()	Diese Funktion setzt den Anfangszustand des Spiels wieder her und gibt den nun aktuellen Frame, mithilfe der Funktion „_get_obs()“, wieder zurück.
render(mode)	Rendert im Normalfall (default) ein Fenster in dem das Spiel dargestellt wird. Alternativ kann der aktuelle Frame auch als Array mit RGB-Werten zurückgegeben werden.

3 Fazit

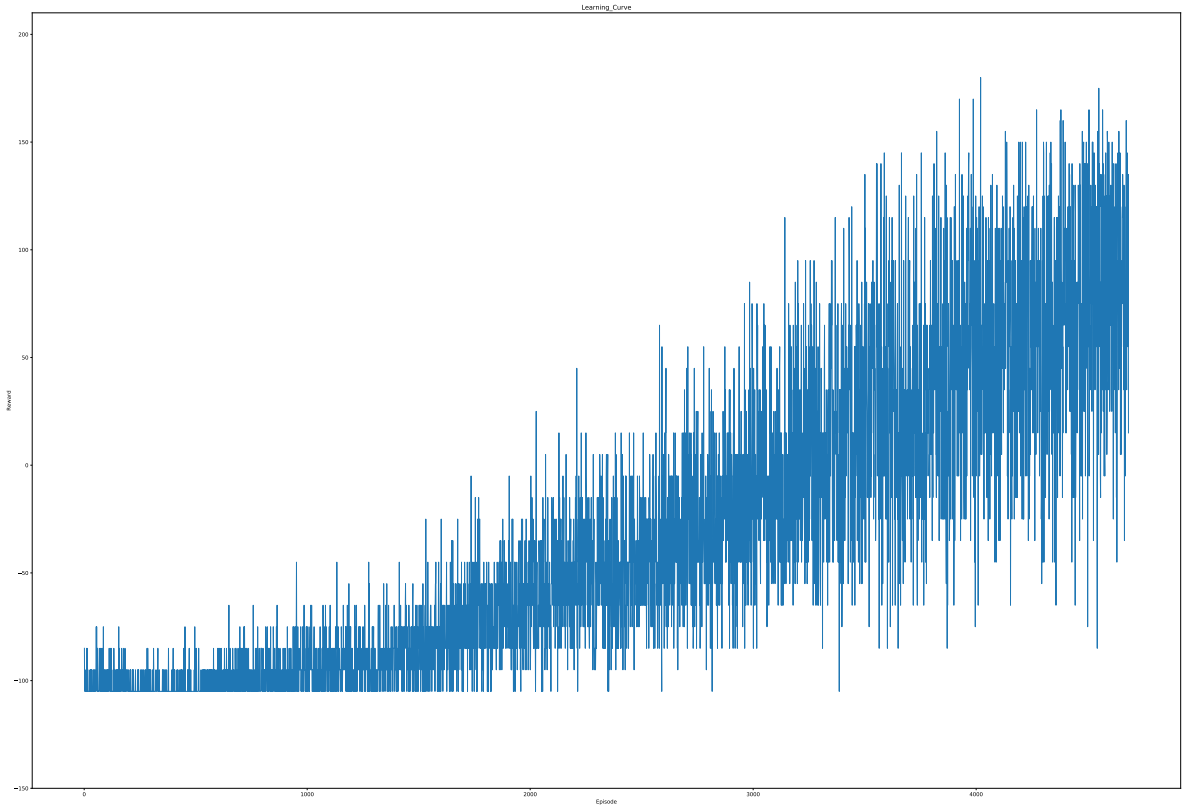
Es wurden zwei neuronale Netze trainiert, eins mit dem normalen Reward und eins mit dem veränderten Reward, über eine Zeitspanne von jeweils 24 Stunden. Anhand der Diagramme lässt sich erkennen, dass die KI mit dem normalen Reward am Anfang besser lernt, allerdings über einen längeren Zeitraum kaum fortschritte macht. Hingegen lernt das neuronale Netz kontinuierlich und dadurch etwas effizienter auf lange Sicht gesehen.

Der veränderte Reward lässt das neuronale Netz in den ersten 24 Stunden effizienter Lernen, jedoch könnte sich dieses Verhalten auch ändern, wenn es über einen längeren Zeitraum trainiert. Dies wurde nicht getestet.

Normaler Reward:



Veränderter Reward:



4 Quellen

Pong-Code für Reinforcement Learning:

https://github.com/schinger/pong_actor-critic [Letzter Stand: 06.02.2019]

Gym Environment erstellen:

https://github.com/openai/gym/tree/master/gym/envs [Letzter Stand: 06.02.2019]