Universität Augsburg
Institut für Informatik
Lehrstuhl für Organic Computing
Prof. Dr. Jörg Hähner

Ansprechpartner
Wenzel Pilar von Pilchau, M. Sc.
wenzel.pilar-von-pilchau@informatik.uni-augsburg.de
Eichleitnerstr. 30, Raum 507

Wintersemester 2019/2020

Praktikum Selbstlernende Systeme

Aufgabenblatt 2

Schicken Sie Ihre Lösung in der Form die in der ersten Veranstaltung festgelegt wurde bis zum **Sonntag, den 17.11.2019 um 24:00 Uhr** an obenstehende E-Mail-Adresse.

1 SARSA (3 Punkte)

1. Erweitern Sie den Agenten aus dem vorherigen Aufgabenblatt um eine SARSA-Lernkomponente.
   a) Pro Step soll der Agent soweit laufen wie er kann.
   b) Für die Modellierung des Zustandsraumes soll nur der x- und y-Abstand vom Marine zum Beacon benutzt werden. Außerdem soll der Zustandsraum in Bereiche aufgeteilt werden:

$$x_1 < s_1 < x_2$$

$$y_1 < s_1 < y_2$$

   Das dient dazu den Zustandsraum zu beschränken.
2. Recherchieren Sie eine weitere Explorationsstrategie (nicht ϵ-greedy).
   a) Welche haben Sie gewählt?
   b) Wie funktioniert Sie?
   c) Implementieren Sie diese.
3. Visualisieren Sie den Lernfortschritt genau so wie beim Q-Learning.
4. Der trainierte SARSA-Agent soll durch ein Python File (RunSARSA.py) gestartet werden können. Der Agent soll die trainierte Q-Tabelle einlesen und diese benutzen ohne weiter zu lernen.
5. Das Training des SARSA-Agenten soll durch ein Python File (TrainSARSA.py) gestartet werden können. Der Agent soll hierbei eine neue Q-Tabelle anlegen.

2 Deep Q-Network (7 Punkte)

1. Erweitern Sie ihren Agenten um eine DQN-Lernkomponente.
2. Für die Modellierung des Zustandsraumes sind keine Bereiche mehr nötig.
3. Erstellen sie mit Keras ein Neuronales Netz mit:
   a) 1 input Layer
   b) 1 hidden Layer (16 Konten + ReLu)
   c) 1 hidden Layer (32 Knoten + ReLu)
   d) 1 output Layer (8 Knoten + Linear)
4. Speichern Sie die gelernten Gewichte im HDF5-Format.
5. Benutzen Sie ein Replay Memory mit folgenden Eigenschaften:
   a) Gesamtgröße: 100.000
   b) Das Replay Memory soll mit mind. 6000 Einträgen befüllt werden bevor das lernen beginnt.
6. Benutzen Sie als Explorationsstrategie ϵ-greedy mit folgenden Eigenschaften:
   a) $\epsilon_{start} = 1$
   b) $\epsilon_{min} = 0.05$
   c) linear abnehmend bis Episode 1.000
7. Visualisieren Sie den Lernfortschritt genau so wie gehabt.
8. Der trainierte DQN-Agent soll durch ein Python File (RunDQN.py) gestartet werden können. Der Agent soll die trainierten Gewichte einlesen und diese benutzen ohne weiter zu lernen.
9. Das Training des DQN-Agenten soll durch ein Python File (TrainDQN.py) gestartet werden können. Der Agent soll hierbei neue Gewichte erlernen.

Viel Erfolg bei der Bearbeitung!