TICTACTOE AI

TicTacToe AI Documentation

Inhaltsverzeichnis

**Es wurden keine Einträge für das Inhaltsverzeichnis gefunden.**

Ansatz

As we got the assignment I started thinking about possible answers to the question: How to train a Neural Network to learn a deterministic game like tic-tac-toe? While I though about this I remembered some of the researches I did for my matura work. I also consulted ChatGPT as I had an idea in mind and wanted to know what other possible solutions there are. On one hand I knew that the idea was to write an algorithm that estimates good moves or take another automated approach on creating the dataset or even finding one online and adjust it so our programs could read it. On the other hand I knew that reinforcement learning could do the job even if it would be a little overkill so after asking ChatGPT and receiving the term “q-learning” I got curious and started my research on that topic. My findings were interesting enough that I decided to take this approach, partially because I tried this before and gave up and partially because it seemed to be a very interesting topic. As I was reading through blog posts and articles, looked at some github code and read some forums I concluded that this would be my approach.

test\_boards = (

np.zeros((3, 3), dtype=int),

np.array(([0, 0, 0], [0, 0, 0], [0, 0, 0]), dtype=int),

np.array(([0, 0, 0], [0, 1, 0], [0, 0, 0]), dtype=int),

np.array(([1, 1, 0], [-1, 0, -1], [0, 0, 0]), dtype=int),

np.array(([1, 0, 0], [0, -1, 0], [-1, 1, 0]), dtype=int),

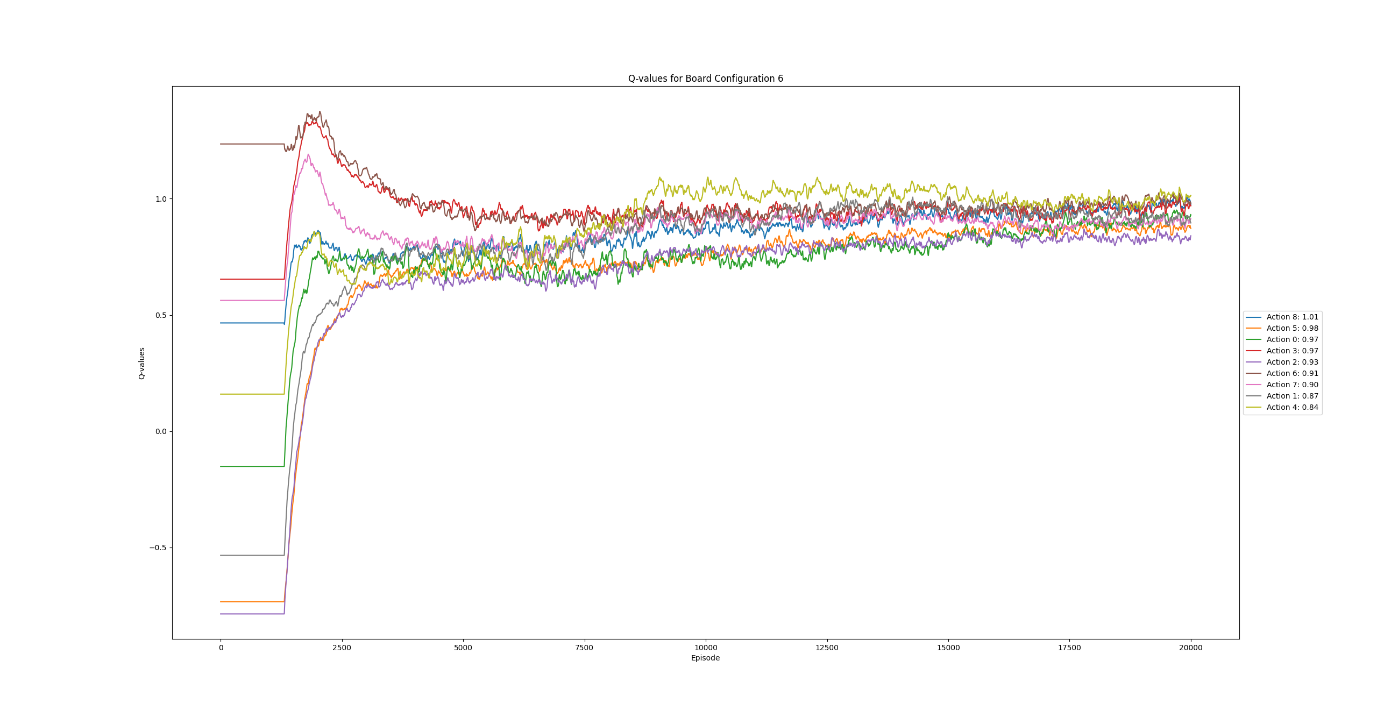
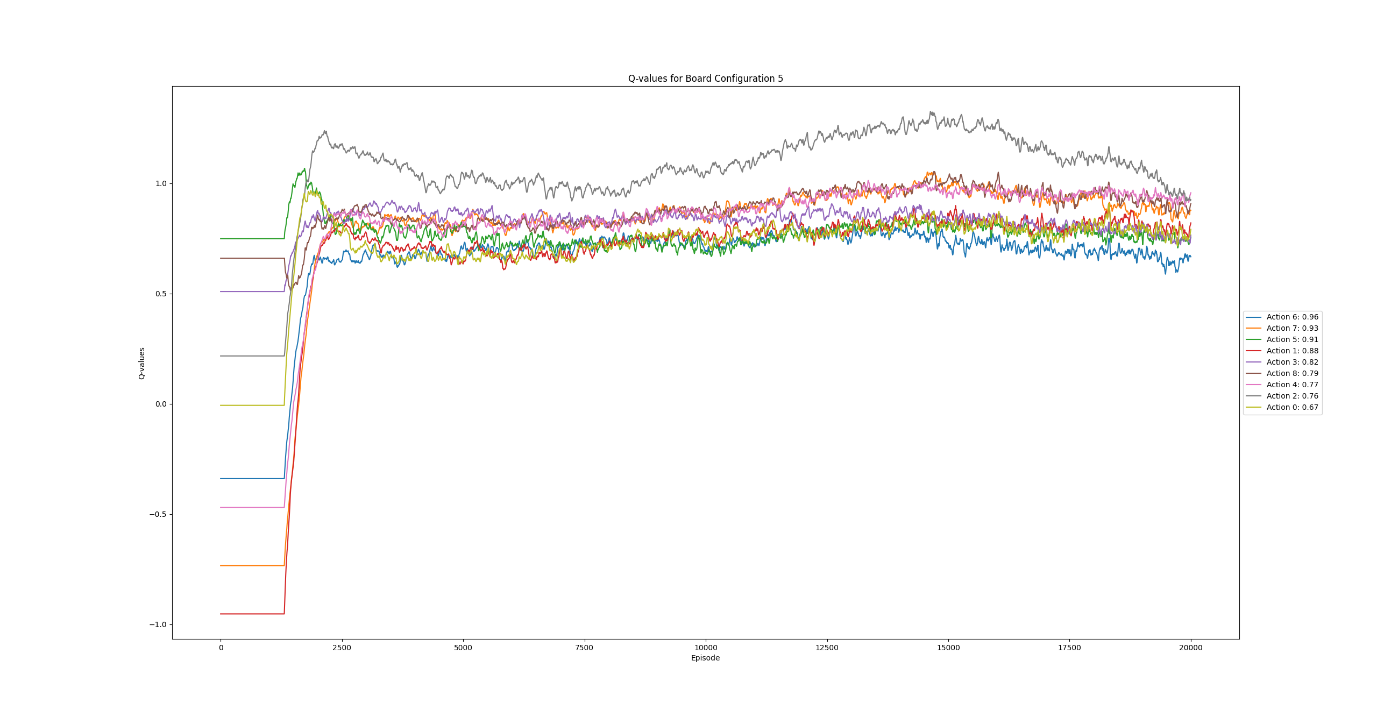
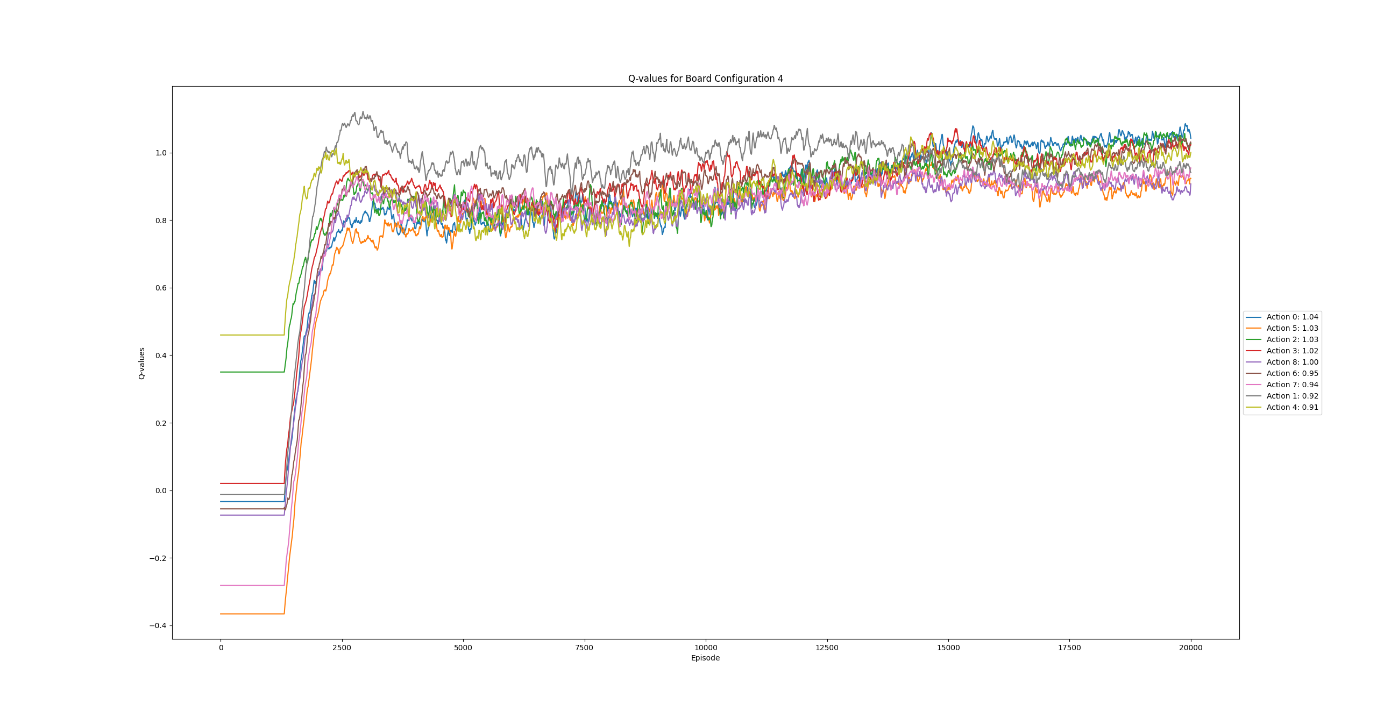
np.array(([0, -1, 0], [0, 1, -1], [0, 0, 1]), dtype=int),

np.array(([1, -1, 1], [-1, -1, 1], [0, 0, -1]), dtype=int)

) Ein Bild, das Text enthält.

Automatisch generierte Beschreibung mit mittlerer ZuverlässigkeitEin Bild, das Text enthält.

Automatisch generierte Beschreibung mit mittlerer ZuverlässigkeitEin Bild, das Diagramm, Text, Karte enthält.

Automatisch generierte Beschreibung

# Quellen

<https://developer.nvidia.com/cudnn>

<https://developer.nvidia.com/cuda-toolkit>

<https://www.tensorflow.org/install/pip#windows-native_1>

<https://www.tensorflow.org/install/source#gpu>

<https://www.tensorflow.org/api_docs/python/tf/keras/layers/Dense$>

<https://www.tensorflow.org/api_docs/python/tf/keras/Model>

https://www.tensorflow.org/api\_docs/python/tf/keras/activations

https://docs.python.org/3/library/multiprocessing.html

<https://moodle.ksasz.ch/pluginfile.php/195842/mod_resource/content/0/Script_NN_Kapitel_4-4_Tensorflow.pdf>

<https://www.simplilearn.com/tutorials/machine-learning-tutorial/what-is-q-learning#:~:text=Q%2Dlearning%20is%20a%20reinforcement,that%20environment%20is%20not%20known>.

https://github.com/tensorflow/agents/blob/master/docs/tutorials/1\_dqn\_tutorial.ipynb

<https://medium.com/@atharvashekatkar1.2/using-q-learning-to-play-the-chrome-dinosaur-game-e836da2bc74a>

<https://pythonprogramming.net/training-deep-q-learning-dqn-reinforcement-learning-python-tutorial/?completed=%2Fdeep-q-learning-dqn-reinforcement-learning-python-tutorial%2F>

<https://github.com/AtharvaShekatkar/DinoGameAI_V2>

<https://www.geeksforgeeks.org/deque-in-python/>

<https://numpy.org/doc/stable/index.html>

<https://medium.com/@yauheniya.ai/visualizing-the-q-learning-algorithm-bd283600773f>

<https://en.wikipedia.org/wiki/Q-learning>