Kein Hexenwerk

Ideen des maschinellen Lernens in SNAP*!*

von Tilman Michaeli, Stefan Seegerer und Sven Jatzlau

Es gibt mittlerweile eine Vielzahl an spannenden didaktischen Programmierumgebungen, sogenannten Frameworks, um Lernende und Programmieranfänger an *maschinelles Lernen* (ML) heranzuführen. Diese Frameworks bieten Lernenden einen einfachen und intuitiven Zugang zu maschinellem Lernen und reduzieren dazu programmierpraktische Hindernisse. Beschäftigt man sich mit den existierenden Ansätzen genauer, ist festzustellen, dass die meisten sich auf *überwachtes Lernen* konzentrieren. Als Kontext dient häufig die Erkennung von Bildern. Oftmals werden dabei die Prinzipien und Ideen von maschinellem Lernen – also die eigentlichen Lerninhalte und -ziele – von Vorgehensweisen zur Mustererkennung überlagert, oder das überwachte Lernen wird mit beschrifteten Daten und einer Trainingsphase mit dem Begriff des maschinellen Lernens gleichgesetzt.

Darüber hinaus ist festzustellen, dass genau die Frage, wie eine Maschine „lernt", zumeist „Hexenwerk" bleibt: Die meisten der existierenden Ansätze konzentrieren sich auf die Anwendung von maschinellem Lernen. Das eigentliche Lernen geschieht nun mit vortrainierten Modellen bzw. durch Programmierschnittstellen-Aufrufe (API-Aufrufe) auf externen Servern. Für die Lernenden steht der eigentliche Lernprozess weder im Fokus noch besteht die Möglichkeit, diesen überhaupt nachzuvollziehen. Dies steht im starken Widerspruch zu einem konstruktionistischen und handlungsorientierten Unterricht, in dem die Schülerinnen und Schüler selbst kreativ und aktiv eigene ML-Artefakte gestalten. Nur das hierdurch entstehende, grundlegende Verständnis der zentralen Ideen und Prinzipien maschineller Lernprozesse (im Gegensatz zur reinen Verwendung eben dieser) ermöglicht den Lernenden, Auswirkungen, Möglichkeiten und Grenzen von künstlicher Intelligenz und im Speziellen von maschinellem Lernen auf unsere Gesellschaft adäquat und kompetent analysieren, diskutieren und mitgestalten zu können.

Im Folgenden stellen wir *SnAIp* vor, ein didaktisches Framework mit einem klaren Fokus auf die zentralen Prinzipien und Ideen maschinellen Lernens für Schülerinnen und Schüler ab der 8. Jahrgangsstufe in der blockbasierten Programmiersprache SNAP*!*. Durch die Betrachtung des tatsächlichen Lernprozesses können die Schülerinnen und Schüler aktiv erfahren, dass maschinelles Lernen eben kein Hexenwerk ist.

Zentrale Prinzipien von SnAIp sind:

- *Blick hinter die Kulissen:* Anstatt vortrainierte Modelle zu verwenden, Bibliotheken anzuwenden oder API-Aufrufe zu nutzen, sollen die Schülerinnen und Schüler hinter die Kulissen und das „Hexenwerk" schauen können und in der Lage sein, die jeweiligen ML-Algorithmen zu verbessern oder für ihre Zwecke anzupassen … alles in SNAP*!*.
- *Eigene Projekte statt Lernprojekte:* Bei der Einführung komplexer Themen wie ML werden oftmals „künstliche", konstruierte Beispiele eingesetzt, da diese die Komplexität realweltlicher Kontexte reduzieren und die Visualisierung bestimmter Konzepte unterstützen. Mit SnAIp wollen wir erreichen, dass Schülerinnen und Schüler ML für ihre persönlichen Projekte, die für sie bedeutsam sind, einsetzen können, wie beispielsweise eigene Spiele.

SnAIp Part A: Verstärkendes Lernen in SNAP*!*

Verstärkendes Lernen und Q-Learning-Algorithmus

Eine Anwendung, auf die im Kontext von maschinellem Lernen zur Entwicklung oder Forschung häufig zurückgegriffen wird, sind klassische Arcade-Videospiele. Diese eignen sich insbesondere, da sie eine „Form von Intelligenz" zum erfolgreichen Spielen benötigen, aber sich auch durch klar definierte Aktionen auszeichnen. Im Beitrag *So lernen Maschinen* (Seite 27 ff. in diesem Heft) sind die Prinzipien und Ideen von *verstärkendem Lernen* bereits beschrieben: Ein Agent – ein Computerprogramm, das zu autonomem Verhalten fähig ist – lernt in Interaktion mit seiner Umwelt durch wiederholte Belohnungen oder Bestrafungen die Erfolgsaussichten seiner Aktionen besser einzuschätzen und ist somit in der Lage, seine Strategie zu optimieren.

In SnAIp Part A verwenden wir einen konkreten Algorithmus für verstärkendes Lernen namens *Q-Learning*. Dieser Algorithmus generiert eine Q-Tabelle, die

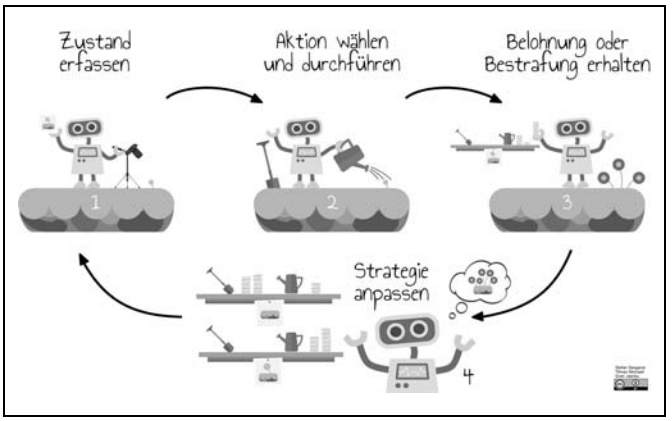


Bild 1: Prinzip des verstärkenden Lernens.

Bild 2: Bananenjagd – das Äffchen lernt, über das Fass zu springen.

der Agent verwendet, um die beste Aktion in einem bestimmten Zustand zu identifizieren. Der Q-Wert in der Q-Tabelle gibt die zu erwartende Belohnung für die Durchführung einer bestimmten Aktion im aktuellen Zustand an. Wird der Agent daraufhin belohnt oder bestraft, so passt er den jeweiligen Wert in der Tabelle an. Damit steigt oder sinkt die Wahrscheinlichkeit, dieses Verhalten erneut zu zeigen. Im Folgenden beschreiben wir eine entsprechende Unterrichtssequenz.

SnAIp Part A im Unterricht

Was ist verstärkendes Lernen?

Für die Einführung der zentralen Ideen und Prinzipien von verstärkendem Lernen wird zunächst das Unplugged-Krokodilschach verwendet (siehe auch Seite 30 in diesem Heft). In dieser Aktivität spielen jeweils zwei Schülerinnen und Schüler eine Partie Minischach gegeneinander.

Die Schülerinnen und Schüler analysieren im Anschluss das Spiel und identifizieren ihre Rollen, relevante Schritte und Bestandteile. Die Analyse mündet in der Visualisierung der „Lernschleife“ für verstärkendes Lernen (siehe Bild 1), die die Schülerinnen und Schüler auch mit ihrem eigenen Lernprozess vergleichen.

Erste Schritte mit verstärkendem Lernen in SNAP!

Im nächsten Schritt geht es darum, einem Affen-Agenten beim Lernen zu helfen und ihn bei seiner Bananenjagd zu unterstützen. Diese Aufgabe wird mit einer Puzzle-Aktivität eingeleitet: Die Schülerinnen und Schüler erhalten eine Vorlage für das Spiel *Bananenjagd* (siehe Bild 2). Sie enthält alle relevanten Blöcke für den selbstlernenden Agenten – allerdings nicht in der richtigen Reihenfolge. Die Schülerinnen und Schüler werden aufgefordert, diese mit Hilfe der Roboter-Visualisierung in die richtige Reihenfolge zu bringen (Lösung, siehe Bild 3). Danach erkunden sie das Programm und beschreiben, wie der Agent lernt. Wir haben die Erfahrung gemacht, dass es über den gesamten Verlauf der Sequenz wichtig ist, immer wieder die verschiedenen Kontexte zu vergleichen: Was sind die Aktionen (Krokodilschach: verschiedene Spielzüge; Roboter: anpflanzen oder gießen; Affe: springen oder nichts tun), was sind die Zustände (Krokodilschach: abgebildete Spielsituationen; Roboter:

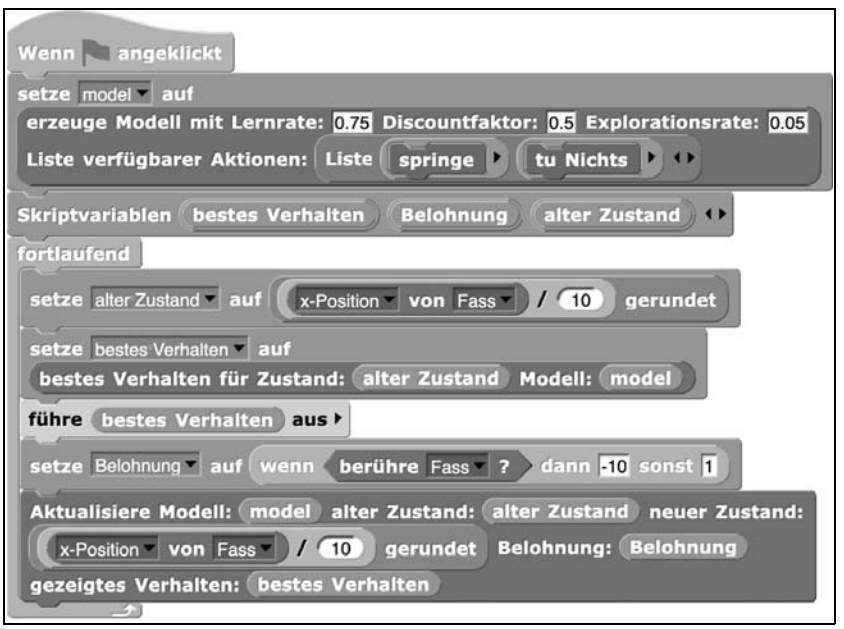
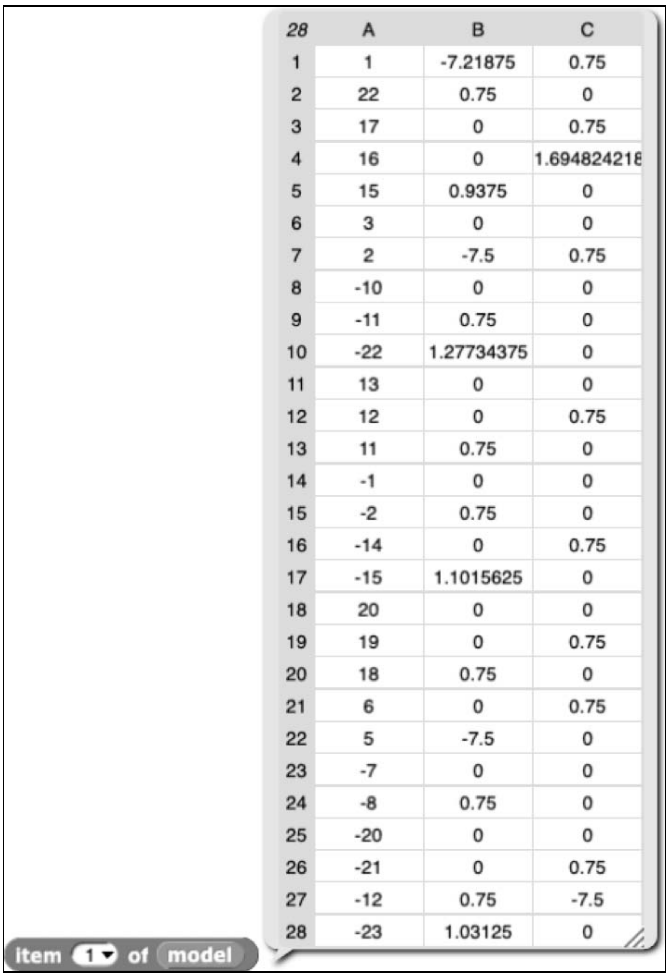


Bild 3:
Die Lernschleife in SNAP!.



28	A	B	C
1	1	-7.21875	0.75
2	22	0.75	0
3	17	0	0.75
4	16	0	1.694824218
5	15	0.9375	0
6	3	0	0
7	2	-7.5	0.75
8	-10	0	0
9	-11	0.75	0
10	-22	1.27734375	0
11	13	0	0
12	12	0	0.75
13	11	0.75	0
14	-1	0	0
15	-2	0.75	0
16	-14	0	0.75
17	-15	1.1015625	0
18	20	0	0
19	19	0	0.75
20	18	0.75	0
21	6	0	0.75
22	5	-7.5	0
23	-7	0	0
24	-8	0.75	0
25	-20	0	0
26	-21	0	0.75
27	-12	0.75	-7.5
28	-23	1.03125	0

Bild 4:
Exemplarische Q-Tabelle des Bananenjagdprojekts.

Spalte A enthält eine Beschreibung des Zustands (hier x-Koordinate des Fasses/10 abgerundet); die Spalten B und C enthalten die Q-Werte für die Aktionen „springen“ bzw. „nichts tun“.

Wachstumsstand der Pflanzen; Affe: x-Position des Fasses), wie wird belohnt und wie wird diese Belohnung verwaltet (Krokodilschach: Schokolinsen in „Tabelle“; Roboter: Geld im Regal; Affe: numerischer Wert in der Tabelle)?

Pimp my Game

Der nächste Schritt für die Schülerinnen und Schüler ist das Implementieren des verstärkenden Lernens mithilfe der SnAIp-Blöcke in ihren eigenen Spielen in SNAP*!*. Hierzu müssen sich die Schülerinnen und Schüler überlegen, welche Aktionen in ihrem Spiel zur Verfügung stehen, wie die verschiedenen Zustände definiert werden können und wie der Agent belohnt oder bestraft werden soll. Im Fall von *Pong* sind die verfügbaren Aktionen möglicherweise „Schläger nach oben“, „Schläger nach unten“, „Nichts tun“, und der Zustand über die y-Koordinate des Balls ist gegeben. Gerade bei der „richtigen“ Belohnung gilt wie auch in der professionellen Welt, dass viel ausprobiert werden muss. Im Prinzip kann verstärkendes Lernen so auf jedes Spiel angewendet werden; es gibt jedoch einige Regeln und Einschränkungen für die Auswahl eines Beispiels zur Verwendung in einer regulären Unterrichtsstunde (hauptsächlich aus Zeitgründen): Die Geschwindigkeit, mit der der jeweilige Agent das erwünschte Verhalten erlernt, ist abhängig von der Größe des Zustandsraums. Beispiele, die sich besonders für den Einsatz im Unterricht eignen, sind Spiele wie *Breakout*, *Pong* oder *Flappy Bird*. Gerade bei wenig Vorerfahrung mit SNAP*!* haben sich unsere online zur Verfügung stehenden Coding Cards, die die eigene Umsetzung schrittweise begleiten, als hilfreiche, optionale Unterstützung erwiesen (siehe Abschnitt „Fazit“, S. 70).

Hinter den Kulissen

Bisher haben die Schülerinnen und Schüler „plugged“ wie „unplugged“ die Prinzipien von verstärkendem Lernen erfahren und aktiv eingesetzt, um ihre Spiele zu „pimpen“, d. h. effektvoller zu gestalten. Im letzten Schritt wird jetzt der eingesetzte Algorithmus Q-Learning und damit die technologische Ebene genauer betrachtet.

Dazu beobachten die Schülerinnen und Schüler, wie sich die Werte der Q-Tabelle (siehe Bild 4) im Laufe der Zeit verändern. Sie experimentieren mit der Explorationsrate (d. h. mit der Wahrscheinlichkeit, mit der ein Agent „neugierig“ handelt und zufällige Aktion exploriert, statt dem laut Modell vielversprechendsten Verhalten zu folgen), dem Discountfaktor (der angibt, welche Bedeutung zukünftigen Belohnungen bzw. Bestrafungen beigemessen wird) und der Lernrate (die angibt, wie schnell der Agent Verhalten lernt). Außerdem können sie Anpassungen oder Optimierungen am Algorithmus vornehmen.

SnAIp Part B: Unüberwachtes Lernen in SNAP*!*

Unüberwachtes Lernen und lineare Vektorquantisierung

Unüberwachtes Lernen ist ein Paradigma des maschinellen Lernens, das stark von der Statistik beeinflusst wird. Häufig besteht die Aufgabe darin, nicht beschriftete Daten zu gruppieren oder zu kategorisieren, z. B. Kunden in Gruppen mit ähnlichen Kaufgewohnheiten zu segmentieren, um etwa Werbeanzeigen zielgruppengerecht auszuliefern.

Part B verwendet lineare Vektorquantisierung (LVQ = ***l****earning* ***v****ector* ***q****uantization*), einen Algorithmus, der solche Gruppen oder Cluster in Datensätzen findet. Das Lernen erfolgt mithilfe einer festen Anzahl von Clustermittelpunkten, sogenannte Prototypen. Für jeden Datenpunkt wird dazu zunächst der nächstgelegene Prototyp (entsprechend einer gewählten Abstands-

Bild 5: Unplugged-Aktivität für den Einstieg in die Sequenz zu unüberwachtem Lernen.

messung) ausgewählt und dessen Koordinaten aktualisiert, sodass sich der Prototyp in Richtung des gerade betrachteten Datenpunkts bewegt.

SnAIp Part B im Unterricht

Was ist unüberwachtes Lernen?

Da unüberwachtes Lernen meist für das Bilden von Clustern verwendet wird, geht es auch in der einleitenden Unplugged-Aktivität darum, Cluster in einem Satz zweidimensionaler Daten zu finden. Die Aufgabe der Schülerinnen und Schüler besteht darin, für die Suche nach Gold im Wilden Westen die bestmöglichen Schürfstellen für drei Grabungsteams zu bestimmen (siehe Bild 5). Eine gute Schürfstelle ist dadurch gekennzeichnet, dass sie möglichst nahe an vielen Goldquellen liegt. Die Schülerinnen und Schüler erhalten dazu eine Karte des Wilden Westens, 25 Datenpunkte in Form von Spielkarten, die aus x- und y-Koordinaten eines berichteten Goldfunds bestehen sowie drei Münzen, die als Prototypen dienen und gleichzeitig die Grabungsteams visualisieren. Die Schülerinnen und Schüler ziehen kontinuierlich Datenpunkt-Karten, überlegen, wie sie die darin enthaltenen Informationen verarbeiten können, und legen die Kärtchen anschließend verdeckt auf den Ablagestapel. Dabei sehen sie wie ein Computer immer nur einen Datenpunkt und nie „alle Daten auf einmal“. Da sie keine Möglichkeit haben, alle Datenpunkte gleichzeitig zu markieren, müssen die Schülerinnen und Schüler eine Strategie finden, wie sie nur mithilfe der Münzen zu lukrativen Schürfstellen gelangen.

Erfahrungsgemäß entwerfen fast alle Schülerinnen und Schüler einen Algorithmus, der konzeptionell der folgenden LVQ-Variante ähnelt:

1. Prototypen in Form von Münzen (zufällig) auf der Karte platzieren.
2. Für jeden Datenpunkt:
   a) Identifizieren des am nächsten gelegenen Prototypen.
   b) Aktualisieren der Position dieses Prototyps, indem er um die halbe Strecke in Richtung der Koordinaten des Datenpunkts bewegt wird.

Nach der Verarbeitung aller Karten erhalten die Schülerinnen und Schüler eine Overheadfolie mit allen Datenpunkten, die sie über ihre Karte legen können, um ihre Endergebnisse zu bewerten. Mithilfe dieser Folie können sie anschließend ihren Algorithmus verbessern. Basierend auf den Erfahrungen, die die Schülerinnen und Schüler in dieser Aktivität gemacht haben, können Prinzipien und Charakteristika des un-

Bild 6 (Mitte):
Prinzip des unüberwachten Lernens.

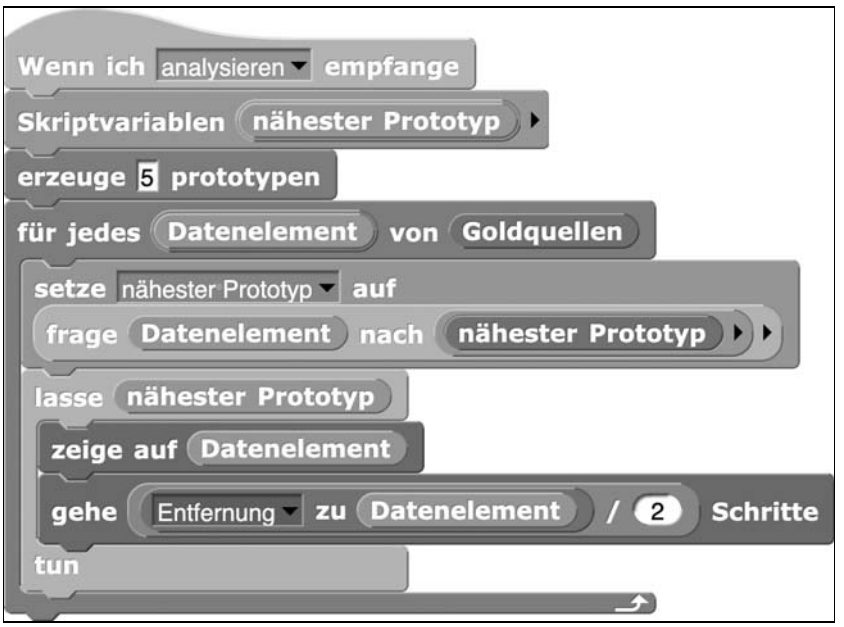


Bild 7 (rechts):
Bilden von Goldclustern mittels unüberwachtem Lernen in SNAP!.

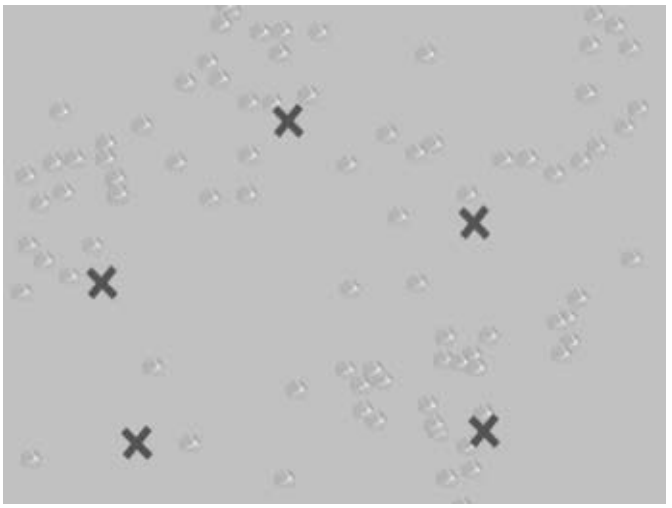


Bild 8: Die SNAP!-Bühne.

überwachten Lernens diskutiert (siehe Bild 6, vorige Seite) und mit anderen Paradigmen des maschinellen Lernens verglichen werden.

Unüberwachtes Lernen in SNAP!

Im nächsten Schritt wird der zuvor entwickelte LVQ-Algorithmus in SNAP! implementiert (siehe Bild 7, vorige Seite). Die Lernenden erhalten eine Vorlage, die Prototypen und eine dynamische Anzahl von zufällig erzeugten Datenpunkten liefert, die bereits auf der Bühne visualisiert werden (siehe Bild 8). Die Aufgabe besteht darin, den eigentlichen LVQ-Algorithmus zu vervollständigen. Dabei können die Schülerinnen und Schüler mit den verschiedenen Parametern, wie der Anzahl der Prototypen, Datenpunkte etc. experimentieren.

Dabei ist man nicht auf die Nutzung von Zufallsdaten beschränkt, sondern kann auch auf reale Daten (die beispielsweise vorher von oder mit den Schülerinnen und Schülern erhoben werden könnten) angewendet werden. So können möglicherweise ideale Standorte für die Schulbushaltestellen bestimmt werden, indem die Längen- und Breitengrade der Wohnorte der Schülerinnen und Schüler im CSV-Format importiert werden. Im Allgemeinen kann dieser Algorithmus auf jeden Datensatz angewendet werden.

Wettbewerb

Anschließend erhalten die Schülerinnen und Schüler in einem Wettbewerb die Aufgabe, den Algorithmus zu optimieren. Hierfür gibt es mehrere mögliche Ansätze, zum Beispiel:

- ▷ Verwendung mehrerer Iterationen zur Verfeinerung der endgültigen Standorte der Prototypen.
- ▷ Schrittweise Verringerung des Bewegungsumfangs für jeden Prototyp.
- ▷ Erhöhung der Priorität von nicht verschobenen Prototypen.
- ▷ Optimierung der Ausgangslage der Prototypen.

Ähnliche Optimierungen werden auch bei anderen Clusteringverfahren angewendet. Mithilfe des CSV-Imports können alle Schülerinnen und Schüler die optimierten Algorithmen schließlich auf denselben Eingabedaten testen.

Fazit

Mit SnAIp können die zugrunde liegenden Ideen und Prinzipien hinter verstärkendem und unüberwachtem Lernen mithilfe blockbasierter Sprachen unterrichtet werden. Dabei liegt ein besonderer Fokus darauf, dass auch ein Blick in die „Black Box“ geworfen wird, denn maschinelles Lernen ist eben kein Hexenwerk. Die verwendeten Algorithmen (tabellenbasiertes Q-Learning und LVQ) sind dabei besonders für den Einsatz in der Schule und auch für die Umsetzung in anderen Programmierumgebungen geeignet. Im Gegensatz zu Technologien wie neuronalen Netzen, die unabhängig vom Lernparadigma (überwacht, unüberwacht oder verstärkend) Anwendung finden, ermöglichen die gewählten Algorithmen einen klaren Fokus auf das zugrunde liegende Prinzip des jeweiligen Paradigmas statt auf mathematische und technologische Details. Die ausführlichere Beschreibung der Sequenz, Materialien etc. sind online unter

https://computingeducation.de/tags/ki/

zu finden. Dort wird auch in Kürze Part C veröffentlicht, der sich mit überwachtem Lernen beschäftigt.

Tilman Michaeli
Stefan Seegerer
Sven Jatzlau
Friedrich-Alexander-Universität Erlangen-Nürnberg
Department Informatik (INF)
Professur für Didaktik der Informatik
Martensstraße 3
91058 Erlangen

E-Mail: tilman.michaeli@fau.de
E-Mail: stefan.seegerer@fau.de
E-Mail: sven.jatzlau@fau.de

Danksagung: Wir danken Urs Lautebach und der Königsteiner Arbeitsgruppe für die Idee und weitere Anregungen, aus denen Part B hervorgegangen ist.