



Methodik

Kryptologie

Lerntransfer

Künstliche Intelligenz

Unterrichtsideen & Beispiele

Inklusion im Informatikunterricht

Editorial

Liebe Leserinnen und Leser,

wir freuen uns, Ihnen die dritte Ausgabe der Zeitschrift *ibis – Informatische Bildung in Schulen* vorstellen zu dürfen. Diese Ausgabe spiegelt erneut die Vielfalt und Relevanz aktueller Themen im Bereich der informatischen Bildung wider und bietet wertvolle Einblicke für die Praxis. Wir sind überzeugt, dass die vorgestellten Artikel eine inspirierende Lektüre für Lehrkräfte, Forschende und alle Interessierten darstellen. Vielen Dank an alle Autorinnen und Autoren sowie alle Reviewer, die im Hintergrund geholfen haben, die vorliegende Ausgabe zu erstellen.

In dieser Ausgabe erwartet Sie unter anderem der Artikel **„Aktivierender Unterricht von Anfang an mit dem PRIMM-Konzept“** von Peter Brichzin und Klaus Reinold, der das PRIMM-Konzept als Scaffolding Gerüst für einen aktivierenden Programmierunterricht vorstellt. Rina Ferdinand, Mareike Daeglau und Ira Diethelm beleuchten in **„Auf dem Weg zum inklusiven Informatikunterricht - Herausforderungen und Perspektiven“** die Herausforderungen, Potenziale und Perspektiven eines inklusiven Informatikunterrichts vor dem Hintergrund eines eklatanten Mangels an Studien, Konzepten und Ressourcen. Ein weiteres spannendes Thema behandelt der Artikel **„ClusterLabor: Ein Werkzeug zur interaktiven Visualisierung und Analyse von Clusteralgorithmen“** von Daniela Andres, Silvia Joachim und Martin Hennecke, der ein neues Werkzeug zur Visualisierung von Clusteralgorithmen vorstellt. Damit können verschiedene Algorithmen hinsichtlich ihrer Ergebnisse in Abhängigkeit von der gewünschten Anzahl an Clustern verglichen werden.

Besonders hervorzuheben ist auch der Beitrag **„Informatik in freier Wildbahn: Lerntransfer vom Unterricht in den Alltag“** von Michael Rücker, der untersucht, wie Informatikwissen außerhalb des Klassenzimmers angewendet werden kann. Für die Informatik müssen Schülerinnen und Schüler dazu u. a. künftige und neue Systemkategorien erschließen und Manifestationen bekannter informatischer Konzepte im Alltag erkennen können. Julian Dorn bringt uns in seinem Artikel **„Künstliche Intelligenz im Informatikunterricht“** die Integration von KI in den Unterricht anhand seines online-Buches näher. **„Symmetrische Kryptologie und ihre Veranschaulichung“** lautet der Titel des Artikels von Andreas Koch, der auch (Java-) Programme zur Verschlüsselung vorstellt. **„Das Informatikcurriculum der Hector Kinderakademien“** richtet sich an begabte und hochbegabte Kinder der ersten bis zur vierten Klassenstufe und wird von Katerina Tsarava, Katrin Kunz und Ulrich Trautwein beschrieben.

„Fachfremd unterrichten – selbstreguliert lernen“ von Susann Lenk liefert einen persönlichen Erfahrungsbericht einer Lehrkraft mit dem für sie neuen Fach Informatik. Den Abschluss bildet Katrin Grabes Beitrag **„Unterrichtsreihe zu künstlicher Intelligenz und künstlichen neuronalen Netzen für die gymnasiale Oberstufe“**, der eine sieben Doppelstunden umfassende Unterrichtsreihe zu KI für die Oberstufe präsentiert.

Die nächste Ausgabe unserer Zeitschrift ist für März 2025 geplant. Wir laden Sie herzlich ein, uns Ihre Beiträge zukommen zu lassen. Wir wünschen Ihnen viel Freude beim Lesen und erfolgreiche Anwendung der vorgestellten Ansätze in Ihrem Unterricht.

Herzliche Grüße,

Ihr Redaktionsteam der Zeitschrift *ibis – Informatische Bildung in Schulen*

Andreas
Peer
Sandra
Tilman

Impressum

Bezug der Zeitschrift

Die Zeitschrift **ibis** erscheint kostenfrei online unter www.informatischebildung.de.

Alle Beiträge stehen dort und zusätzlich in der *Digitalen Bibliothek* der *Gesellschaft für Informatik* unter dl.gi.de als Open-Educational-Ressources zur Verfügung.

Zusätzlich werden zu verschiedenen Anlässen Printausgaben produziert, z. B. zur 20. GI-Fachtagung „Informatik und Schule“ 2023, die vor Ort an der Veranstaltung bezogen werden können.

Beitragseinreichung

Beiträge zur **ibis** können jederzeit online über die Webseite www.informatischebildung.de eingereicht werden. Dort finden Sie auch den Beitragsaufruf und alle Informationen zur Einreichung.

Für Fragen steht die Redaktion gerne zur Verfügung.

Bibliographische Angaben

ISSN (Print): 2941-7538

ISSN (Online): 2941-7546

DOI-Namensraum: 0.18420/ibis-JG-NR-ART (JG: Jahrgang, NR: Heftnummer, ART: Artikelnummer)

Herausgeber

Gesellschaft für Informatik e. V.
Fachausschuss Informatische Bildung an Schulen
Fachgruppe Didaktik der Informatik

Wissenschaftszentrum, Ahrstraße 45, 53175 Bonn

Redaktion

Dr. Andreas Grillenberger (verantwortlicher Redakteur)

Prof. Dr. Tilman Michaeli

Prof. Dr. Sandra Schulz

Dr. Peer Stechert

www.informatischebildung.de

info@informatischebildung.de

Beirat

Alisch, Sven · Anthes, Jaqueline · Arnold, Peter · Barkmin, Mike · Bergner, Nadine · Best, Alexander
Bockelberg, Stefanie · Brinkmeier, Michael · Burk, Steffen · Batur, Fatma · Diethelm, Ira · Dorn, Julian
Gallenbacher, Jens · Geldreich, Katharina · Gramm, Andreas · Grillenberger, Mareen · Hellmig, Lutz
Hempel, Tino · Hennecke, Martin · Hielscher, Michael · Hildebrandt, Claudia · Kastl, Petra · Losch, Daniel
Otto, Torsten · Puhlmann, Hermann · Pöhner, Nicolai · Rau, Thomas · Reher, Jan · Reinold, Klaus
Romeike, Ralf · Rücker, Michael · Schmidt, Pascal · Spalteholz, Wolf · Strecker, Kerstin

Autorinnen und Autorinnen dieser Ausgabe

Andres, Daniela · Brichzin, Peter · Daeglau, Mareike · Diethelm, Ira · Dorn, Julian · Ferdinand, Rina · Grabe, Katrin
Hennecke, Martin · Joachim, Silvia · Koch, Andreas · Kunz, Katrin · Lenk, Susann · Reinold, Klaus · Rücker, Michael T.
Trautwein, Ulrich · Tsarava, Katerina

Gestaltung und Satz

Dr. Andreas Grillenberger

Urheberrecht / Lizenzen

Die Artikel in dieser Zeitschrift unterstehen dem Urheberrecht und der Verantwortung der jeweiligen Autorinnen und Autoren. Die Redaktion überprüft die Beiträge ebenfalls auf erkennbare (Urheber-)Rechtsverletzungen, die Verantwortung für solche liegt aber weiterhin bei den Autorinnen und Autoren.

Die Autorinnen und Autoren erteilen der **ibis** eine nicht-exklusive Veröffentlichungslizenz. Am Ende jedes Artikels wird die Lizenz genannt, unter der ein Artikel zur Verfügung steht.

Alle von der Redaktion erstellten Teile der Zeitschrift unterstehen der Lizenz CC BY-NC 4.0. Als Attribution wird die Nennung des Namens der Zeitschrift und der Webadresse www.informatischebildung.de gefordert.

Bildquellen

Titelgrafik: Erstellt mit Microsoft Designer unter Nutzung des folgenden Prompts:
„paint robots helping humanity in an abstract digital art style“

Grafiken in den Artikeln: Bildquelle gem. Angabe an den Grafiken

Inhalt

Editorial.....	1
Impressum.....	2
Ausschreibung des Unterrichtspreises der GI 2025.....	5
Veranstaltungstermine der Fachgruppen.....	5

Impulse

Dorn, J.: Künstliche Intelligenz im Informatikunterricht	7
Reinold, K. und Brichzin, P.: Aktivierender Unterricht von Anfang an mit dem PRIMM-Konzept	11

Aus der Wissenschaft für die Praxis

Ferdinand, R., Daeglau, M. und Diethelm, I.: Auf dem Weg zum inklusiven Informatikunterricht: Herausforderungen und Perspektiven	17
Rücker, M. T.: Informatik in freier Wildbahn: Lerntransfer vom Unterricht in den Alltag	25
Tsarava, K., Kunz, K. und Trautwein, U.: Das Informatikcurriculum der Hector Kinderakademien	33

Praxisbeiträge

Koch, A.: Symmetrische Kryptologie und ihre Veranschaulichung	43
Lenk, S.: Fachfremd unterrichten, selbstreguliert lernen – eine Lösung für den Informatikunterricht.....	51
Andres, D., Joachim, S. und Hennecke, M.: ClusterLabor: Ein Werkzeug zur interaktiven Visualisierung und Analyse von Clusteralgorithmen.....	55
Grabe, K.: Unterrichtsreihe zu künstlicher Intelligenz und künstlichen neuronalen Netzen für die gymnasiale Oberstufe	65

Ausschreibung für den Unterrichtspreis der GI 2025

Innovative Unterrichtsbeispiele gesucht!



Die GI schreibt einen Unterrichtspreis für Lehrkräfte an allgemeinbildenden und berufsbildenden Schulen aus.

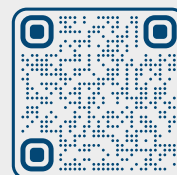
Sie bewerben sich mit einer maximal zweiseitigen deutschsprachigen Kurzbeschreibung eines Beispiels für gelungenen Informatikunterricht. Das kann eine einzelne Unterrichtsstunde oder eine Unterrichtssequenz sein. Wichtig ist, dass Sie den Unterricht tatsächlich durchgeführt haben und neben dem Verlauf auch dessen Ergebnisse und Ihre Beobachtungen beschreiben. Der eingereichte Beitrag muss bezüglich Text und Material datenschutz- und urheberrechtlich einwandfrei sein.

Ihre Kurzbeschreibung wird von einer Jury des Fachausschusses begutachtet. Wenn Ihr Unterrichtsbeispiel gute Aussichten hat, den Unterrichtspreis zu gewinnen, werden Sie nach dem Ende der Einreichungsfrist durch die Jury Ende März 2025 gebeten, bis zum 31. Mai 2025 eine 4- bis 10-seitige Darstellung Ihres Projekts zu verfassen. Die Bekanntgabe und Ehrung der Preisträger erfolgt auf der INFOS in Stoos (Schweiz) vom 22. bis zum 24. September 2025.

Alle Informationen zum Unterrichtspreis, zu den Kriterien und zur Einreichung werden ab 1. Oktober 2025 unter <https://gi.de/aktuelles/wettbewerbe/unterrichtspreis> veröffentlicht.

Terminübersicht

16.02.2025	Kurzbeschreibung einreichen
31.03.2025	Benachrichtigung der Finalisten
01.06.2025	Vollbeitrag einreichen
Juni 2025	Benachrichtigung des/der Preisträger(s)
22.-24.09.2025	Siegerehrung



Veranstaltungstermine der Fachgruppen



BIL (Bayern)

19.09.2024: Informatiklehrerinnen- und Lehrertag Bayern an der TU München

DDI (Didaktik der Informatik)

16.-18.09.2024: WiPSCE-Konferenz an der TU München

HRPI (Hessen und Rheinland-Pfalz)

19.09.2024: Landestagung an der Sportschule und Bildungsstätte des Landessportbunds

IBBB (Berlin und Brandenburg)

24.09.2024: Informatiktag Berlin-Brandenburg an der HTW Berlin

06.03.2025: GI-Tagung zur Schulinformatik in Berlin und Brandenburg an der FU Berlin

IBiSaTh (Sachsen und Thüringen)

06.11.2024: Fachkonferenz im ScaDS.AI LivingLab Leipzig

02.04.2025: Sächsischer Schulinformatiktag an der TU Dresden und online

IBMV (Mecklenburg-Vorpommern)

01.10.2024: Herbstliche Informatiklehrerfortbildung an der Universität Rostock

05.04.2025: Landestagung der Informatiklehrerinnen und -lehrer MV an der Universität Rostock

IBN (Nordrhein-Westfalen)

31.03.2025: Informatiktag NRW an der RWTH Aachen University

IBNB (Niedersachsen und Bremen)

25.03.2025: Tag der Informatiklehrerinnen und -lehrer Niedersachsen und Bremen am Landesinstitut für Schule in Bremen

ILL-BW (Baden-Württemberg)

27.09.2024: Heidelberger Informatiklehrtag im Mathematon Heidelberg

SH-HILL (Schleswig-Holstein und Hamburg)

23.11.2024: Fachtagung am Gymnasium Meiendorf, Hamburg

Künstliche Intelligenz im Informatikunterricht

Dorn, J.

DOI: 10.18420/ibis-02-02-02

Zusammenfassung

Das kostenfreie digitale Buch <https://buch.informatik.cc/ki/> bietet eine umfangreiche Grundlage, um Maschinelles Lernen als Teil der künstlichen Intelligenz im Unterricht zu behandeln. Die Themenbereiche überwacht, unüberwacht, bestärkendes und selbstüberwacht Lernen sind mit vielen Aufgaben und Beispielen beschrieben, sodass darauf aufbauend ein eigener Unterricht entwickelt werden kann. Da sich der theoretische Teil in dem verlinkten digitalen Buch nachlesen lässt, stelle ich hier die didaktischen Überlegungen und Ausgestaltungen vor.

Einleitung

Künstliche Intelligenz ist allgegenwärtig. Um diese aber nicht nur als Phänomen, sondern fundiert im Informatikunterricht behandeln zu können, habe ich ein kostenfreies digitales Buch dazu geschrieben.

<https://buch.informatik.cc/ki/> behandelt alle notwendigen Themen des sächsischen Informatiklehrplans für die neunte und elfte Klasse und erweitert sich auf generative KI wie ChatGPT. Das Thema eignet sich aber in Auszügen schon ab der siebten Klasse. In der siebten, achten und elften Klasse habe ich das Thema selbst mehrfach unterrichtet. In der neunten bis zehnten Klasse fehlen mir die Erfahrungen. Schon mit einer Doppelstunde lassen sich viele sinnvolle Einblicke gewinnen. Sechs Doppelstunden kann ich für einen fundierten Einblick empfehlen.

Seit ChatGPT ist es nicht mehr notwendig, Maschinelles Lernen im Informatikunterricht zu begründen. Eine solche Begründung soll aber dennoch kurz erfolgen: Im Gegensatz zu klassischen Algorithmen, die eine Eingabe mit bekannten Verarbeitungsschritten zu immer der gleichen Ausgabe umformen, bildet Maschinelles Lernen durch i.d.R. definierte Ein- und Ausgaben das Modell zur Verarbeitung selbst. Die Vorteile sind offensichtlich: Der schwierige Algorithmus, wie etwa das Erkennen eines Kleeblatts, muss nicht mehr selbst programmiert werden. Dafür muss man mit mangelnder Nach-

vollziehbarkeit und zwangsweise Ungenauigkeit bei der Erkennung leben.

Wer KI nicht im Lehrplan hat, sollte überlegen das Thema KI im Kontext von Data Literacy zu betrachten. Dies findet sich in Form von Datenmanagement oder Datenbanken in allen Lehrplänen. Dabei hat Andreas Grillenberger¹ den Datenlebenszyklus eingeführt, welcher sich stark vereinfacht auf

1. die Datengewinnung und -auswahl
2. Vorverarbeitung der Daten
3. Datenanalyse (Training der KI)
4. Datennachbereitung (Test und Einsatz der KI)

reduzieren lässt.

Ich habe mich bei dem Thema Künstliche Intelligenz auf das maschinelle Lernen konzentriert, erwähne aber selbstverständlich auch kurz, dass es klassische Künstliche Intelligenz gibt. Glücklicherweise kann Prolog so einfach im Leseverständnis sein, dass man hier durch Familienbeziehungen einen kleinen ersten Eindruck gewinnen kann.

Neben den bekannten überwachten, unüberwachten und bestärkenden Lernen gibt es noch das Selbstüberwachte Lernen. Da dies ist ein wichtiger Bestandteil der Generativen KI, habe ich diese Lernart mit als Hauptüberschrift aufgenommen.

Im Einklang mit dem Dagstuhldreieck umfasst jedes Unterkapitel, den theoretischen Teil, eine praktische Anwendung/Simulation und eine gesellschaftliche Betrachtung.

Der Artikel wurde im Frühjahr 2024 geschrieben. Es kann gut sein, dass neue Entwicklungen neue Schwerpunkte in der Behandlung im Unterricht notwendig machen.

Das digitale Buch beginnt mit der Frage, was Intelligenz und künstliche Intelligenz ist. Dabei wird aber nur die Frage nach der künstlichen Intelligenz beantwortet, da es keine einheitliche Definition von Intelligenz gibt.

Ganz grundsätzlich halte ich Maschinelles Lernen für ein sehr abstraktes Thema. Daher habe ich mich dazu entschlossen, in meinem Unter-

¹<http://dx.doi.org/10.17169/refubium-1932>

richt zuerst einen Überblick zu geben, wo uns das Thema bereits im Alltag begegnet und wie es sich gliedert. Als Einstieg eignet sich ein Quiz, ob eine KI das schon kann oder nicht. Legt man noch ein paar falsche Fährten, hat man die Aufmerksamkeit errungen. So sieht etwa jedes Bild von ThisPersonDoesNotExist² realistischer aus als ein retuschiertes Stock-Foto.

Überwachtes Lernen

Überwachtes Lernen ermöglicht es uns, aus vorab kategorisierten Daten automatisch Modelle zu entwickeln.

Ein idealer Einstieg in dieses Thema bietet der Entscheidungsbaum, der sich durch seine Einfachheit auszeichnet. Ein besonders zugänglicher Ansatz findet sich im „AI unplugged“-Projekt mit dem Beispiel „Welche Affen beißen?“, das einen niederschweligen Zugang bietet. Wer bereits Erfahrung mit Scratch und mehr Zeit hat, kann das Spiel „Zombie Escape“ aus dem Programm „Machine Learning for Kids“ ausprobieren, welches ins Deutsche übersetzt wurde.

Ab der Sekundarstufe II lässt sich ein technischerer Einblick durch ein in Excel erstelltes künstliches neuronales Netz gewinnen. Ich habe ein starkes Interesse an der Funktionsweise dieser Netze festgestellt, jedoch auch oft eine Überforderung der Schüler:innen bemerkt. Am Beispiel der der Bilderkennung lässt sich der Datenlebenszyklus vollständig nachvollziehen. Dabei habe ich ganz bewusst auf Analogien zum Gehirn soweit es geht verzichtet, um weniger Fehlvorstellungen in diesem stark vereinfachten Modell zu vermeiden und zu betonen, dass es Wahrscheinlichkeitsrechnung und kein menschliches Denken ist.

1. Die Datenauswahl ist vorgegeben, da es sehr schwer war in Excel überhaupt ein halbwegs funktionierendes Modell zu erschaffen.
2. Die Datenaufbereitung zeigt, dass Bilder bestimmten Formaten entsprechen müssen, was Fehlerquellen birgt.
3. In der Trainingsphase habe ich versucht, das komplexe Thema des Trainings von neuronalen Netzen anzugehen, indem ich es in eine einfache Geschichte verpackte, die frei von mathematischen Konzepten ist. Aus meiner Sicht ist der mathematische Ansatz sehr herausfordernd und entspricht nicht der Idee eines grundlegenden Konzepts, da sich die Methoden mit verschiedenen KI-Iterationen weiterentwickeln.
4. Die folgende Aktivierungsfunktion verdeutlicht, dass maschinelles Lernen im Kern ein statistisches System ist, das Antworten mit einer bestimmten Wahrscheinlichkeit liefert. Dies bietet eine gute Gelegenheit, zu reflektieren, wie selten KI-Systeme die Sicherheit ihrer Antworten offenlegen und welche Konsequenzen das haben kann.

Abschließend beleuchtet der Abschnitt zur gesellschaftlichen Perspektive des überwachten Lernens, wie durch maschinelles Lernen getroffene Entscheidungen direkt unsere Gesellschaft beeinflussen können. Ein gutes Beispiel hierfür ist das selbstfahrende Auto. Mit der „Moral Machine“³ konnte ich gute Erfahrungen sammeln, um zu diskutieren, dass Entscheidungen, selbst wenn sie gut sind, nicht notwendigerweise einer Maschine überlassen werden sollten. Ein anschließender provokanter und humorvoller Auszug aus „Qualityland“⁴ beleuchtet die Vor- und Nachteile des autonomen Fahrens.

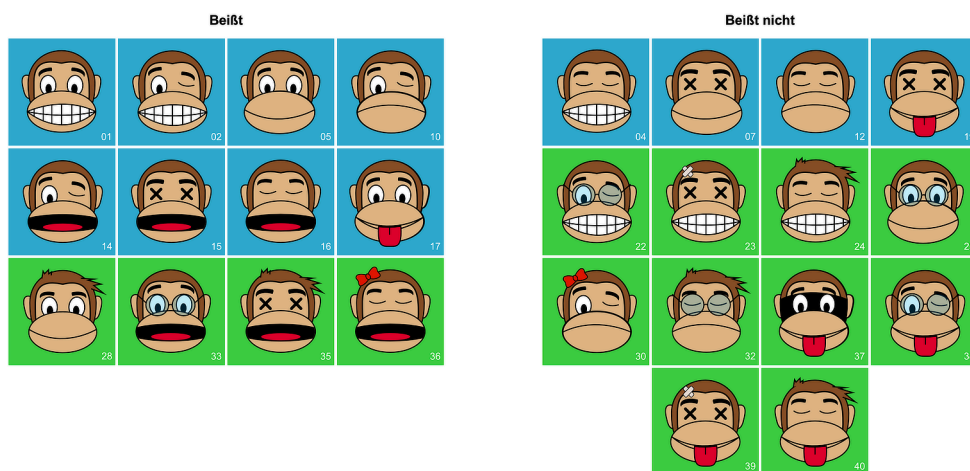


Abbildung 1: Überwachtes Lernen am Beispiel des Äffchenspiels, Grafik von Stefan Seegerer
(<https://www.stefanseegerer.de/decision-tree-monkey-game/start-advanced.html>)

² <https://thispersondoesnotexist.com>

³ <https://www.moralmachine.net/hl/de>

⁴ <https://de.wikipedia.org/wiki/Qualityland>

Unüberwachtes Lernen

Unüberwachtes Lernen ermöglicht es, Muster und Beziehungen in Daten zu erkennen, ohne dass explizite Anweisungen erforderlich sind.

Dies ist besonders nützlich für das Empfehlen von Musikstücken oder Videos auf sozialen Medien. Ein einfacher Weg, dies praktisch selbst durchzurechnen, ist der Einsatz des k -Means-Algorithmus. Leider fehlt mir hier ein motivierenderes praktisches Beispiel, weshalb ich anhand eines sozialen Netzwerkes nur eine Übung mit gesellschaftlichem Bezug durchrechnen lasse.

Ein kritischer Aspekt im Bereich des unüberwachten Lernens ist die Notwendigkeit, zwischen Korrelation und Kausalität zu unterscheiden. Dies ist zwar traditionell eine mathematische Herausforderung, die im Kontext von KI eine besondere Relevanz erhält, sodass ich diese hier mit aufgenommen habe.

Im Grunde gruppieren wir ständig jeden Tag Menschen. Wir haben feste Vorurteile, wie sensibel der bärtige Mann mit Tattoos wohl ist und wie das zierliche Mädchen wohl seine Freizeit verbringt. Ausgehend von diesen Alltagsbeobachtungen lassen sich die Auswirkung von automatisierten Vorurteilen gut reflektieren.

Bestärkendes Lernen

Beim bestärkenden Lernen muss eine Aufgabe nicht nur erfüllt, sondern möglichst gut erfüllt werden. So geht es etwa beim Eiskunstlauf nicht nur darum, nicht hinzufallen, sondern vor allem darum, die ausgewählten Figuren möglichst eindrucksvoll darzubieten.

Da dies in Spielen der Fall ist, ist es eines der spannendsten Themen für Schüler:innen. Ein gutes Beispiel hierfür ist das Schachspiel, bei dem Deep Blue 1997 den damaligen Weltmeister Garry Kasparov besiegte. Eine einfache Möglichkeit, bestärkendes Lernen praktisch zu erleben, bietet das Spiel „Schlag das Krokodil“ von AI unplugged, welches ich trotzdem als Browser-Simulation sehr empfehlen möchte. Einmal habe ich das Spiel wie vorgeschlagen mit echten Süßigkeiten durchgeführt, jedoch konnten die meisten Teams nicht zu Ende spielen, da die roten Schokolinsen auf wundersame Weise einfach alle waren. Da die KI beim Bauernschach als Zweitziehende immer gewinnt, wenn sie keine Fehler macht, führt das Spiel nach einer gewissen Dauer zu Frustration, sodass man gut den Fokus auf den eigentlichen Lernprozess umlenken kann.

Ein persönliches Highlight war für mich ein Video, in dem ein KI-Entwickler demonstrierte, wie das Gameboy-Spiel Pokémon mithilfe von bestärkendem Lernen angegangen wurde. Ich habe das Video mithilfe von KI übersetzt und mit Genehmigung veröffentlicht, um zu zeigen, wie eine Belohnungsfunktion aufgebaut wird und welche Herausforderungen dabei entstehen. Es hat sich gezeigt, dass nicht nur das Endziel belohnt werden muss, sondern auch entsprechende Zwischenziele. Bei Pokémon gab es überraschende Parallelen zum menschlichen Verhalten, wie etwa das Sammeln kurzfristiger Belohnungen, auch wenn diese für das Gesamtergebnis nicht relevant waren, und das Meiden traumatischer Ereignisse.

Bestärkendes Lernen dient auch als Grundlage für viele Science-Fiction-Szenarien. Ich habe dazu die Unterhaltungsfilme „I Am Mother“ (2019) und „I, Robot“ (2004), welche Isaac Asimovs Gesetze der Robotik besprechen, als einzige Hausaufgabe⁵ schauen lassen und anschließend besprochen. Diese Filme zeigen (ACHTUNG SPOILER), wie eine künstliche Intelligenz ihre Zielfunktion maximieren möchte und dabei der Menschheit massiv schadet.

Diese Filme verstärken die Befürchtung, dass durch bestärkendes Lernen letztendlich alles vernichtet werden würde. Als Gegenperspektive empfiehlt sich der Film „Her“ (2013), in dem (ACHTUNG SPOILER) die KI einfach das Interesse an der Menschheit verliert. Ich habe weiterführend recherchiert⁶, um fundiert darlegen zu können, dass ein Weltuntergang eher das Resultat einer sogenannten Alphamännchen-Logik ist, als ein realistisches Szenario. Warum sollte eine KI fähig sein, Asimovs Gesetze frei zu interpretieren, aber nicht erkennen, dass die Versklavung der Menschheit konträr zu allen anderen Zielen steht?

Selbstüberwachtes Lernen

Das selbstüberwachte Lernen kombiniert die Ansätze des überwachten und unüberwachten Lernen.

Es ist sicher das Themengebiet, auf das die meisten Schüler:innen gewartet haben. Da es aber viele Themengebiete vereint (und die Motivation oben hält) habe ich es ganz ans Ende gestellt.

⁵ Oft kostenfrei bei Streaming-Diensten, gebrauchte DVDs hatte ich zum Ausleihen und wer gar nicht wollte hatte eine passende Leseaufgabe erhalten: <https://waitbutwhy.com/2015/01/artificial-intelligence-revolution-2.html>

⁶ <https://schulki.de/blog/die-ki-wird-uns-alle-vernichten-wie-sag-ich-es-den-kindern>

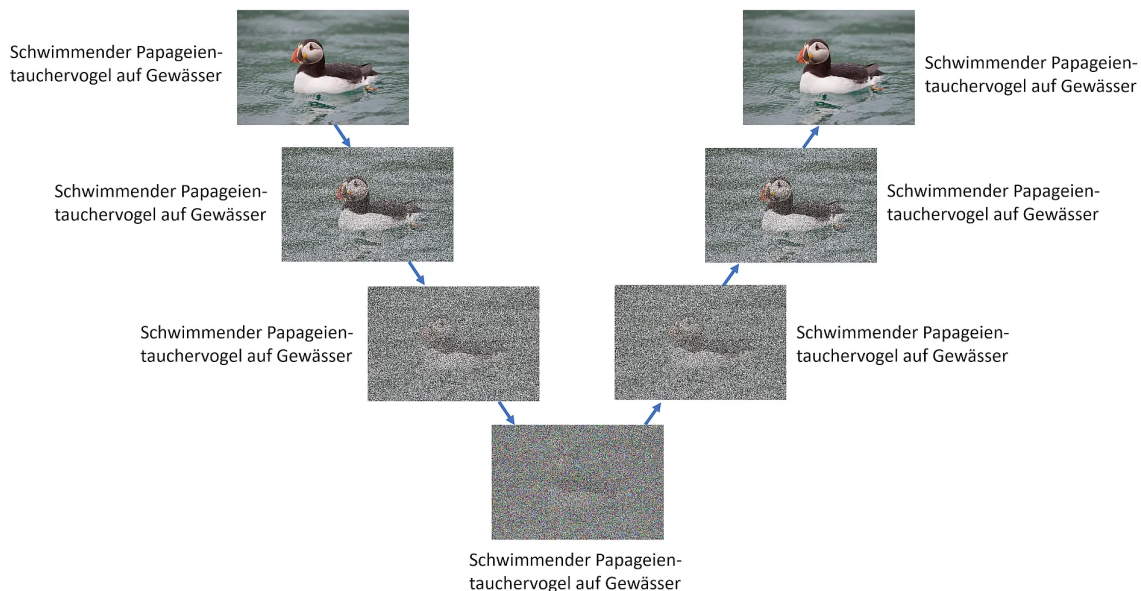


Abbildung 2: Als Beispiel für Selbstüberwachtes Lernen wird das Foto eines Vogels zuerst schrittweise verrauscht, um dann eine KI darauf zu trainieren, das Original zu rekonstruieren

Ein sehr guter Einstieg ist die Bildgenerierung, da hier das selbst überwachtes Lernen durch das Ent- und Verrauschen von Bildern sehr anschaulich gezeigt werden kann.

Gleichzeitig wird bei der Bildgenerierung das bestärkende Lernen durch menschliche Rückkopplung mit eingeführt, welches schön aufzeigt, wie weit entfernt das maschinelle Lernen noch von menschlichem Lernen ist.⁷

Beim Textgenerieren habe ich das spiralcurriculare Prinzip innerhalb eines Themenbereichs aufgegriffen und versucht, die Textgenerierung in drei Schwierigkeitsstufen zu erklären. In der siebten Klasse bleiben wir gern bei der ersten Iteration oder hören uns die zweite als Geschichte an. In der elften Klasse gehen wir in die zweite Iteration rein und mit Python-Kenntnissen hätte ich gern mit meinem Kurs meine Harry-Potter-Textanalyse auch einmal selbst nachvollzogen. Die dritte Iteration ist eher zum Selbststudium für besonders Interessierte gedacht ist.

Ab der zweiten Iteration der Erklärung zur Textgenerierung lässt sich ein Vergleich mit der Bildgenerierung erstellen, sodass hier Parallelen erkannt werden und gleichzeitig das Verständnis deutlich erleichtert wird.

Ich persönlich halte das Verständnis der Textgenerierung für sehr essenziell, um zum einen die allgegenwärtigen Chat-KIs besser zu verstehen, aber diese auch verantwortungsvoll ein-

setzen zu können. Aus diesem Grund habe ich die Prompt-Strategien mit eingeführt, obwohl diese natürlich keine fundamentalen Ideen darstellen, sondern sogar von OpenAI als Fehler und nicht als Funktion betrachtet werden⁸. Dennoch lässt sich die Time-To-Think-Methode erst verstehen und zielführend anwenden, wenn man weiß, dass die eingesetzte KI nur ein Wortvervollständigungssystem ist.

Anschließend habe ich mir in allen Jahrgängen die Zeit genommen aktuelle gesellschaftliche Probleme, wie Bias, Verlässlichkeit, Rolle der Trainingsdatenautoren und Einsatz als Hausaufgabenhilfe zu besprechen und auszuprobieren. Ich hätte das nicht machen müssen, aber selbst meine achten Klassen wussten schon, wie einzelne soziale Netze bereits eine KI für die Hausaufgaben integriert haben. Dann doch lieber mit Begleitung durch uns Lehrende als alleine.

Lizenz



Dieser Artikel steht unter der Lizenz CC BY NC 4.0 zur Verfügung.

Kontakt

Julian Dorn
E-Mail: post@wi-wissen.de

⁷ <https://www.spektrum.de/inhaltsverzeichnis/ki-und-ihr-biologisches-vorbild-spezial-bmh-1-2024/2186325> versucht zu zeigen, dass aktuelles maschinelles Lernen auf einen veralteten Lernverständnis beruht.

⁸ <https://the-decoder.de/prompt-engineering-ist-ein-fehler-kein-feature>

Aktivierender Unterricht von Anfang an mit dem PRIMM-Konzept

Reinold, K. und Brichzin, P.

DOI: 10.18420/ibis-02-02-03

Zusammenfassung

PRIMM ist ein Ansatz, der dazu dient, den Programmierunterricht so zu strukturieren, dass die Lernenden selbst von Anfang an aktiv sind. Dabei erhalten sie im Sinne des Scaffolding ein „Lerngerüst“, welches die Lernenden in passenden Schritten bei der Aneignung von Programmierkompetenzen unterstützt. PRIMM steht für Predict (Vorhersagen) – Run (Ausführen) – Investigate (Untersuchen) – Modify (Verändern) – Make (kreatives Erstellen). Die Methode fördert die Analyse von Programmen; vom Lesen und Interpretieren gelangt man über kleinere Variationen hin zum eigenständigen Schreiben eines Programms.

Einleitung

Im Informatikunterricht sind zwei grundlegende Herangehensweisen weit verbreitet: Einerseits werden häufig materialgeleitete Übungen durchgeführt, die die Lernenden Schritt für Schritt zu einem gewünschten Ergebnis führen (z. B. Stem learning: Scratch cards unter <https://kurzlinks.de/primm2>). Andererseits ist der erste Teil einer Unterrichtseinheit oft geprägt von der Instruktion durch die Lehrperson: Neue Konzepte werden häufig – methodisch vermittelt durch Vortrag oder Unterrichtsgespräch – im Plenum eingeführt, dabei gemeinsam im Programm erprobt und erst nach einer Sicherungsphase dürfen die Lernenden im zweiten Teil der Unterrichtseinheit endlich selbst an den Rechnern aktiv werden. Der Übergang zum eigenständigen Modellieren und Programmieren ist bei beiden Herangehensweisen oft mit Unsicherheit und Angst verbunden, da man unvermittelt „ins kalte Wasser geworfen wird“ und zeigen muss, dass man die präsentierten Konzepte verstanden hat und auch selbst syntaktisch und semantisch in anderem Kontext fehlerfrei umsetzen kann. Das PRIMM-Konzept stellt dem einen Ansatz gegenüber, bei dem die Lernenden von Beginn an zu Handelnden werden und ausgehend von der Codeanalyse über die Codevariation hin zu einem tieferen Verständnis sowie einem kreativen und kompetenten Entwickeln von Programmen gelangen.

Ursprung und Aufbau

Das PRIMM-Konzept wurde im Jahr 2017 von Sue Sentence und Jane Waite vorgeschlagen (Sentence/Waite, 2017)

Es fußt auf dem co-konstruktiven Ansatz von Lew Wygotski. Wesentliche Aspekte dieses Ansatzes in Hinblick auf das PRIMM-Konzept sind die folgenden (Textor, 2000):

- Die Lehrperson als Aktivitätengestalter: Sie wählt geeignete Aktivitäten aus, stellt passende Materialien bereit und interagiert nach den Erfordernissen mit den Lernenden.
- Die Lehrperson als Dialogpartner: Im Austausch mit den Lernenden unterstützt die Lehrperson den Lernprozess hin zu einem tieferen Verständnis.
- Die Lehrperson im Lernprozess: Die Lehrperson beteiligt sich am Problemlösungsprozess nur so weit, wie die Lernenden Unterstützung benötigen und unterstützt dabei den Aufbau von Verbindungen zwischen den bereits erworbenen Kompetenzen und neuen Kenntnissen.

Das PRIMM-Konzept besteht nun aus den folgenden Schritten:

- *Predict*: Die Lernenden erhalten ein vorgegebenes Programm und sagen dessen Verhalten voraus. Sie zeichnen ihre Vermutung in geeigneter Form, z. B. stichwortartig, auf.
- *Run*: Die Lernenden führen das vorgegebene Programm aus und überprüfen und diskutieren dabei ihre Vermutung.
- *Investigate*: Die Lernenden beschäftigen sich genauer mit dem Code, beispielsweise durch die Diskussion geeigneter Fragen („Was würde eine bestimmte Änderung ... einer Codezeile bewirken?“, „Was ergibt sich bei veränderter Eingabe ...?“), schrittweises Nachvollziehen, Arbeiten mit dem Debugger oder Kommentieren.
- *Modify*: Die Lernenden variieren das Programm in einer Reihe von zunehmend anspruchsvolleren Übungen. Sie machen sich den Code so nach und nach mehr zueigen und gewinnen dadurch Vertrauen in ihre eigenen Fähigkeiten.
- *Make*: Die Lernenden entwerfen neue Programme unter Nutzung der neu erlernten Konzepte.

Beim PRIMM-Konzept ist die Reihenfolge und Intensität der einzelnen Phasen nicht zwingend vorgeschrieben: „*You may not be able to go through all the stages in one lesson, and you may focus on one stage more than another. Using the PRIMM framework gives you a way of labelling what you are doing when you are teaching programming.*“ (Coleman, 2021, S. 22) So kann es beispielsweise sinnvoll sein, bei komplexen Programmen zunächst mit der Run-Phase zu starten, damit ein Grundverständnis für das Ziel des Programms gewonnen wird und sich die Lernenden dem Programm annähern, um danach in der Phase Investigate die für einen bestimmten Effekt verantwortlichen Codezeilen zu finden und zu interpretieren. Durch eine Variation von Codezeilen oder Eingabewerten kann man im Anschluss in einer Predict-Phase ein tieferes Verständnis gewinnen und sich so über mehrere Zyklen an die Konzepte annähern.

Beispiele aus der eigenen Praxis

Tabellenkalkulationssysteme – Logische Funktionen

Teilaufgabe a) der untenstehenden Beispielaufgabe leitet mit *Predict* ein. Dabei ist noch kein Öffnen der Tabellenkalkulationsdokuments nötig, da die Formeln auch der Abbildung entnommen werden können – dies hilft zu fokussieren.

Bei der Vorhersage unterstützt auch die einleitende Beschreibung der Zielsetzung im Rahmen eines schülernahen Kontexts.

Run/Investigate findet in b) statt: Mit dem Tabellenkalkulationsdokument können die Lernenden ihre Antwort aus a) überprüfen und das Verhalten mit anderen Eingabewerten erkunden.

Das *Modify* ist in dieser Aufgabe nicht nur auf das Implementieren beschränkt (Teilaufgabe d), sondern über c) wird auch eine Darstellung als Modell und eine Begründung als Text gefordert. Dies steigert die Vielfalt der Kompetenzanforderungen.

In Teilaufgabe e) werden die Lernenden abschließend aufgefordert, ihre eigene Perspektive mit einzubringen und das Rechenblatt entsprechend zu erweitern. Dies ist ein erstes kleines *Make*. Das Erstellen von Rechenblättern zu anderen Kontexten mit den neu erlernten Konzepten findet dann in weiteren Aufgaben des Kapitels statt. Typischerweise wird vorher die Einstiegsaufgabe im Unterricht besprochen, um mögliche Fragen zu klären.

Unter den Eltern herrscht Frust: Viele Kinder essen täglich Pommes oder Spaghetti. Deshalb überlegt die Schulleitung des John-von-Neumann-Gymnasiums zusammen mit Mensapächter und Schülervertretung die Einführung einer Empfehlung für das Mensaessen.

Das erste Kriterium ist die biologische Erzeugung, die sichergestellt ist, wenn die Zutaten ein Biosiegel des Handels haben oder direkt von regionalen Biobauern stammen. Für den Genusswert müssen Geschmack, Aussehen und Geruch stimmen. Zum Dritten soll gesichert sein, dass die Ernährung nicht einseitig ist.

	A	B	C	D	E	F	G	H
1	Biologische Erzeugung			Genusswert			Einseitigkeit	
2	Biosiegel	WAHR		guter Geschmack	WAHR		ist einseitig?	FALSCH
3	örtlicher Anbau	FALSCH		sieht gut aus	WAHR			
4				riecht gut	WAHR			
5	empfehlenswert: =ODER(B2;B3)			empfehlenswert: =UND(E2;E3;E4)			empfehlenswert: =NICHT(H2)	

a. Das Ergebnis der Funktionen in den Zellen B5, E5 und H5 sind Wahrheitswerte. Notiere die Werte, die deiner Meinung nach berechnet werden.

b. Experimentiere mit dem vorliegenden Tabellendokument, indem du die Wahrheitswerte in den Spalten B, E, H veränderst. Untersuche die Auswirkungen in den Berechnungen von Zeile 5.

c. Die Schülervertretung wünscht sich eine Auszeichnung „rundum empfehlenswert“, wenn alle drei Kriterien gleichzeitig erfüllt sind. Begründe, welche der Funktionen aus Zeile 5 du dafür benötigst. Erstelle für die Präsentation im Schulforum ein anschauliches Datenflussdiagramm, das auch Schulleitung und Mensapächter überzeugt.

d. Ergänze das Rechenblatt um die Auszeichnung „rundum empfehlenswert“.

e. Welche Kriterien sind für dich beim Mensa-Essen wichtig? Ergänze mindestens ein zusätzliches Kriterium im Rechenblatt und passe die Formeln, soweit nötig, an.

(Übernahme von Brichzin/Jetzing/Neumeyer/Reinold/Wiedemann, 2021, S. 44 mit geringer Adaption)

Datenmodellierung und Datenbanksysteme – Einfache Datenbankabfragen

Sicher hast du schon einmal in einem Online-Shop nach einem Kleidungsstück, einem Buch oder etwas Ähnlichem gesucht. Nach der Eingabe des Suchbegriffes bekommst du sofort die passenden Artikel angezeigt. Aber was steckt eigentlich hinter der Webseite des Shops? Viele Anwendungen, die große Mengen an Daten bereitstellen und verarbeiten, verwenden zur Datenverwaltung eine sogenannte Datenbank. Unter <https://www.dbiu.de/shop/> kannst du einen Blick hinter einen Webshop werfen.

a Öffne die oben angegebene Seite und betrachte die Ansicht im Reiter DB-Backend. Beschreibe, wie die Artikel hier dargestellt sind und vergleiche mit der Darstellung im User-Frontend.

b Stelle im User-Frontend die Filter so ein, dass dir nur die Farbe und der Preis aller T-Shirts angezeigt werden. Wende die Filter an und wechsle in das DB-Backend. Analysiere die hier angegebene Datenbankabfrage.

c Formuliere im DB-Backend eine weitere Datenbankabfrage, welche die Art und den Preis aller Artikel in der Farbe Blau zurückgibt! Überprüfe deine Eingabe, indem du nach dem Senden der Abfrage wieder zurück in das User-Frontend wechselst.

(Brichzin/Jetzinger/Neumeyer/Reinold/Wiedemann, 2021, S. 60)

Diese Aufgabe, die in einem schülernahen Kontext sehr praktisch das Kapitel Datenbank einleitet, startet direkt mit Run/Investigate. Durch den Wechsel zwischen Backend und Frontend ist sehr einfach der Blick hinter die Kulissen von Webseiten mit Datenbankabfragen möglich. Durch den Wechsel zwischen den Ansichten sind Elemente des Predict auch enthalten.

Über den Modify-Auftrag in Teilaufgabe b) können sich die Schüler*innen den Aufbau und Bestandteile einfacher SQL-Abfragen erschließen. Auf dieser Basis sind in c) neue Datenbankabfragen (Make) möglich.

User-Frontend

DB-Backend

Datenbankmodell:

```

E, art:ZEICHENKETTE, typ:ZEICHENKETTE, geschlecht:ZEICHENKETTE, farbe:ZEICHENKETTE, pre
  
```

Klassendiagramm

Datenbankabfrage:

```

1 SELECT *
2 FROM Kleidung
  
```

Abfrage senden

Schriftgröße: + -

Ergebnistabelle (148 Datensätze):

artikelnummer	art	typ	geschlecht	farbe	preis	bildverweis
2110532	Hose	Jeans	m	blau-washed	89.95	bilder/2110532.jpg
2112044	Hose	Jogginghose	m	schwarz	17.95	bilder/2112044.jpg
2112497	Hose	Jogginghose	m	rot	24.95	bilder/2112497.jpg
2112860	Hose	Jeans	w	grau	89.95	bilder/2112860.jpg

Abbildung 1: Abfragemöglichkeit im Datenbankbackend

User-Frontend

DB-Backend

Filter:

Filter anwenden

Art

☐ Hose

☐ Rock

☐ T-Shirt

Typ

Geschlecht

Farbe


Preis

Anzeigen:

☒ Bild ☒ Artikelnummer ☒ Art ☒ Typ ☒ Geschlecht ☒ Farbe ☒ Preis

anwenden

Produkte (148):



Artikelnummer: 2110532


Art: Hose

Typ: Jeans

Geschlecht: m

Farbe: blau-washed

Preis: 89.95



Artikelnummer: 3248249

Art: T-Shirt

Typ: Langarm

Geschlecht: w

Farbe: orange

Preis: 24.95

Abbildung 2: Ergebnis der Abfrage im Frontend

Objektorientierte Modellierung und Programmierung – Überschreiben von Methoden

Beim Spiel „Don't touch“ muss der rote Ball durch die Pfeiltasten gesteuert werden, so dass er nicht gegen die verschiedenartigen Hindernisse stößt.

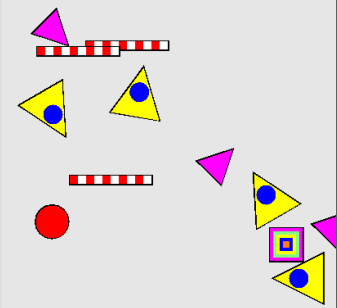
a. Erprobe das Spiel und beobachte das unterschiedliche Aussehen und Bewegungsverhalten insbesondere von Schlagbaum und Quadrat. Vergleiche deine Beobachtung mit dem Code der Methoden *Bewegen* und *Zeichnen* in den Klassen *SCHLAGBAUM* und *QUADRAT*. Ergänze als Ergebnis deiner Beobachtungen einen kurzen Kommentar im Quelltext vor die Methode *Bewegen*, der die Bewegung beschreibt.

b. Erstelle ein Klassendiagramm zu den Klassen *HINDERNIS*, *QUADRAT* und *SCHLAGBAUM* mit den jeweiligen Methoden.

c. Der Kopf der Methoden *Bewegen* und *Zeichnen* in der Klasse *QUADRAT* bzw. *SCHLAGBAUM* hat in den meisten Programmiersprachen ein neues Syntaxelement wie `@override` bzw. `@Override`. Erkunde die Bedeutung des englischen Begriffs und versuche dann zu erklären, was bei den betrachteten Methoden gemeint ist.

Tipp: Wirf zum Vergleich auch einen Blick in die Klasse *HINDERNIS*.

d. Erstelle selbst eine neue Unterklasse von *HINDERNIS* mit neuem Aussehen und Bewegungsverhalten. (Übernahme von Brichzin/Jetzinger/Neumeyer/Reinold/Wiedemann, 2021, S. 130 mit geringer Adaption)



In Teilaufgabe a) ist *Predict* mit *Run/Investigate* eng verzahnt. *Predict* ergibt sich aus der Betrachtung des statischen Quelltexts; sobald das Programm ausgeführt wird, ist man bereits in der *Run/Investigate-Phase*.

Herausfordernd bei Quelltextanalysen für die Phasen *Predict/Investigate* ist häufig, dass das Programm einen gewissen Umfang benötigt, damit es Jugendlichen als motivierendes, interessantes Betätigungsfeld wahrnehmen. Der Umfang mag eine Gefahr sein, dass sich die Schülerinnen und Schüler im Quelltext verlieren bzw. überfordert fühlen. Deshalb ist es

wichtig, den Quelltext gut zu strukturieren und bei den Schüler*innen die Kompetenz aufzubauen, zwischen Blackbox-Sicht und Whitebox-Sicht zu unterscheiden. Betrachtet man beispielsweise die Methode *Bewegen* (Whitebox-Sicht), ist diese in den Unterklassen sehr übersichtlich. Die aufgerufene Methode *EntfernenWennAußerhalb* muss dabei nicht näher betrachtet werden (Blackbox-Sicht).

Klasse SCHLAGBAUM	Klasse QUADRAT
<pre>/** * Lässt das Hindernis um 5 Einheiten * gehen */ @Override void Bewegen() { Gehen(5); EntfernenWennAußerhalb(); }</pre>	<pre>/** * Lässt das Hindernis um 20 * Einheiten gehen */ @Override void Bewegen() { Gehen(20); EntfernenWennAußerhalb(); }</pre>

Quelltext 1: Quelltexte der Klassen für die Aufgabe

Codierung – Binärsystem

Bei einer Show des Wahlkurses Zaubertricks sollen Zahlen per „Gedankenübertragung“ übermittelt werden. In Wirklichkeit aber ver-raten kleine Strahler in der Bühnenbeleuchtung dem „Zauberer“ die jeweiligen Werte. Das System ist in der Vorlage dargestellt: Durch Klicken auf die einzelnen Leuchten kann man sie an- bzw. ausschalten. Die dadurch heimlich übermittelte Zahl wird darunter angezeigt.



13

a. Finden Sie heraus, nach welchem System die Zahlen dargestellt werden und wie viele Werte mit den vier Leuchten insgesamt be-schrieben werden können. Geben Sie die Darstellung der Zahl 10 an.

b. Folgern Sie, wie die Zahlen 32 und 47 nach diesem System dargestellt werden müssten, und geben Sie beide Darstellungen an. (Hinweis: Der Zahlenbereich wird erweitert, es sind mehr als vier Lampen nötig.)

c. Für Schnelle: Die Leuchtmittel werden durch LEDs ausgetauscht, die neben gelb auch noch in rot leuchten können. Erschließen Sie sich das veränderte System und geben Sie eine Darstellung der Zahlen 17 und 64 an. Wie viele Werte könnte man mit den vier vorhandenen Leuchten nun insgesamt darstellen?

(Übernahme von Brichzin/Janus/Jetzinger/Neumeyer/Reinold/Seegerer/Wiedemann 2023, S. 78 mit gerin-ger Adaption)

Vorlage zur Aufgabe siehe <https://kurzlinks.de/primm3>

Das letzte Beispiel soll zeigen, dass Intensionen von PRIMM, insbesondere die kognitive Aktivie-rung, auch bei Lerngerüsten ohne Quelltexte für den (Informatik-)Unterricht nutzbar sind: Teilaufgabe a bringt die Lernenden sofort in eine aktive, explorierende Rolle (*Run/Investi-gate*), denn sie müssen sich einen neuen Inhalt (hier das Binärsystem) erschließen. Bewusst wurde hier ein Kontext gewählt, der zumindest auf den ersten Blick nichts mit Mathematik zu tun hat, um eventuell vorhandene Barrieren zu umgehen. Auf den Erkenntnissen aus Teilaufga-be a aufbauend muss zur Darstellung der Zah-len 32 und 47 das System erweitert werden (*Mo-dify*). Bei beiden Teilaufgaben empfiehlt es sich, von den Schüler*innen eine schriftliche Zusammenfassung einzufordern, die Basis für eine ge-meinsame Besprechung und Ergebnissicherung im Plenum ist. Geht man regelmäßig so vor, kann dabei der Mehrwert von Skizzen mit Le-genden gegenüber rein textlichen Zusammen-fassungen deutlich gemacht werden. Da beim beschriebenen konstruktivistischen Prozess die Lerngeschwindigkeit recht unterschiedlich ist, ist die binnendifferenzierende Teilaufgabe c sehr wichtig. Diese bietet für die Leistungsstär-keren eine Herausforderung, indem über das

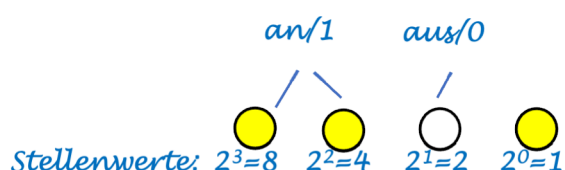


Abbildung 3: Beispielnotiz zur Aufgabe

Binärsystem hinaus auf das Ternärsystem er-weitert wird; das Ergebnis der Bearbeitung kann an späterer Stelle im Unterricht wert-schätzend eingebracht werden, beispielsweise wenn das Hexadezimalsystem angesprochen wird.

Fazit

Dieser Artikel zeigt Einsatzmöglichkeiten von PRIMM, auch über Programmieraufgaben hin-aus, und dessen Potenzial in einem konsequent konstruktivistisch geführten Informatikunter-richt. Der Ansatz unterstützt die Lehrperson in der Strukturierung von Lernaufgaben hin zu ei-nem konsequent schüleraktivierenden Unter-richt, der durch die Abfolge der Phasen den Ler-nenden der verschiedenen Leistungsniveaus passend entgegenkommt. Wie bei allen Unter-richtsmethoden ist auch PRIMM kein Selbstläu-fer bei einem vereinzelt Einsatz. Eigenverant-wortliches Arbeiten und die Fähigkeit der Ler-nenden, zwischen Blackbox- und Whitebox-Sicht wechseln zu können, sind ebenso wichtige Voraussetzungen für einen erfolgreichen Ein-satz, wie die passende Reduzierung der Aufga-benkomplexität durch die Lehrenden.

Material

Vorlagen zu den vorgestellten Beispielen (und vielen anderen Einstiegsaufgaben) können un-ter informatikschulbuch.de heruntergeladen werden. Das Material für die Beispiele 1, 3 und 4 ist unter informatikschulbuch.de/primm zu-sammengefasst.

Quellen

Alle Webseiten/Links wurden zuletzt geprüft am 16.06.2024.

Brichzin, Peter; Jetzinger, Franz; Neumeyer, Johannes; Reinold, Klaus; Wiedemann, Albert (2021): Informatik 3: Funktionale Modellierung, Datenmodellierung und objektorientierte Modellierung, Cornelsen Verlag, Berlin

Brichzin, Peter; Janus, Florian; Jetzinger, Franz; Neumeyer, Johannes; Reinold, Klaus; Seegerer, Stefan; Wiedemann, Albert (2023): Informatik Gymnasium Oberstufe 1: Algorithmen, Codierung, Kommunikation in Netzwerken, Künstliche Intelligenz, Cornelsen Verlag, Berlin

Coleman, Gemma (Hg.): The PRIMM Approach. In: Raspberry Pi Foundation (2021): The Big Book of Computing Pedagogy. <https://www.raspberrypi.org/hello-world/issues/the-big-book-of-computing-pedagogy>

Sentance, Sue; Waite, Jane (2017): PRIMM: Exploring pedagogical approaches for teaching text-based programming in school. in Proceedings of the 12th Workshop on Primary and Secondary Computing Education, 113 – 114. Nijmegen, The Netherlands: ACM.

Sentance, Sue; Waite, Jane; Kallia, Maria (2019): Teaching computer programming with PRIMM: a sociocultural perspective, Computer Science Education, DOI: 10.1080/08993408.2019.1608781

Stem Learning: Scratch cards: <https://www.stem.org.uk/resources/elibrary/resource/35110/scratch-cards>

Textor, Martin: Lew Wygotski – der ko-konstruktive Ansatz. In: Fthenakis, Wassilios E., Textor, Martin R. (Hg.): Pädagogische Ansätze im Kindergarten. Weinheim, Basel: Beltz 2000, S. 71-83

Lizenz



Dieser Artikel steht unter der Lizenz CC BY NC 4.0 zur Verfügung.

Kontakt

Klaus Reinold
Rupprecht-Gymnasium, München
E-Mail: reinold@rupprecht-gymnasium.de

Peter Brichzin
Erasmus-Grasser-Gymnasium, München
E-Mail: schule@brichzin.de

Auf dem Weg zum inklusiven Informatikunterricht Herausforderungen und Perspektiven

Ferdinand, R., Daeglau, M. und Diethelm, I.

DOI: 10.18420/ibis-02-02-04

Zusammenfassung

Der Beitrag untersucht die zunehmende Bedeutung inklusiver Ansätze im Informatikunterricht angesichts der wachsenden Diversität der Schülerschaft. Basierend auf einer Betrachtung des aktuellen Forschungsstands werden zentrale Herausforderungen und Potenziale beleuchtet. Es zeigt sich ein Mangel an Studien, Konzepten und Ressourcen für die Umsetzung inklusiver Bildung in der Informatik. Als vielversprechender Ansatz wird das Universal Design for Learning (UDL) vorgestellt. Der Artikel schließt mit Empfehlungen für Praxis, Aus- und Fortbildung sowie Forschung, um die Inklusion im Informatikunterricht in Deutschland voranzubringen.

Einleitung

In den letzten Jahrzehnten wurde das Thema Inklusion im Bildungsbereich immer wichtiger. Ein bedeutender Schritt war die „*Empfehlung zur sonderpädagogischen Förderung in den Schulen in der Bundesrepublik Deutschland*“ vom 6. Mai 1994. Mit der Ratifizierung der *UN-Behindertenrechtskonvention (CRPD)* am 24. Januar 2009 hat sich Deutschland verpflichtet, inklusive Bildungssysteme umzusetzen. Seit 2013 ist der Anteil der Schüler*innen mit sonderpädagogischem Förderbedarf von 6,6% auf 7,5% im Jahr 2022 gestiegen. Auch der Anteil jener, die Regelschulen besuchen, hat sich von 31,4% auf 44,1% erhöht (KMK 2022).

Inklusiver Unterricht kann viele Vorteile bringen. Schüler*innen mit sonderpädagogischer Förderung finden oft innovative Lösungen für komplexe Probleme, was besonders in der Informatik hilfreich ist (Wille et al. 2017). Die Informatik bietet zudem Technologien, die den Alltag von Menschen mit Behinderungen erleichtern. Mehr Chancengerechtigkeit und die Einbindung unterrepräsentierter Gruppen erhöhen die Vielfalt und Leistungsfähigkeit der Branche (Miesenberger 2015). Beispiele hierfür sind barrierefreie Programmierumgebungen und spezielle Eingabegeräte (Dirks 2022). Die Informatikbranche bietet gute Berufschancen für Menschen mit Behinderungen und sonderpädagogischem Förderbedarf. Es ist wichtig, dass diese Menschen aktiv an der Entwicklung

von Technologien teilnehmen und ihre Perspektiven einbringen können. Ein inklusiver Bildungsansatz gibt allen Lernenden gleiche Chancen und fördert die Vielfalt, indem er auch bisher unterrepräsentierte Gruppen einbezieht. Trotzdem bekommt die Entwicklung inklusiver Lernumgebungen in der Informatik noch nicht genug Aufmerksamkeit (Stefik 2019).

Was ist Inklusion?

Inklusion bedeutet, dass alle Menschen, unabhängig von ihren Fähigkeiten oder Hintergründen, gleichberechtigt an der Gesellschaft teilhaben können. Dazu gehören auch Menschen mit Behinderungen, die notwendige Unterstützung erhalten müssen (Textor 2015). Betroffene Gruppen sind unter anderem Kinder und Jugendliche mit Behinderungen, aus schwierigen Verhältnissen, mit Migrationshintergrund, besonderen Begabungen oder unterschiedlicher sexueller Orientierung (Saalfrank/Zierer 2017). Zu den Förderschwerpunkten bei Behinderungen gehören beispielsweise Lernen (3%), geistige Entwicklung (1,38%), emotionale und soziale Entwicklung (1,35%), Sprache (0,77%), körperliche und motorische Entwicklung (0,51%), Hören (0,27%) und Sehen (0,12%) (KMK 2022).

Im Informatikunterricht gibt es viele Herausforderungen, wie zum Beispiel komplexe Inhalte, spezielle Software, abstrakte Konzepte und sprachliche Barrieren. Traditionelle Lehrmethoden können dabei ganze Gruppen von Schüler*innen ausschließen (Stefik et al. 2019). Daher sind flexible und individuell angepasste Lernumgebungen notwendig, um allen Lernenden die Möglichkeit zu geben, kompetente Anwender*innen und vor allem Mitgestalter*innen von Informatiksystemen zu werden. Das Selbstvertrauen und die Wahrnehmung der eigenen Fähigkeiten in Mathematik und Problemlösung beeinflussen die Entscheidung für ein Informatikstudium (Hong et al. 2015). Besonders Mädchen haben oft ein geringeres Selbstvertrauen, was sie davon abhalten kann, sich in der Informatik zu engagieren (Cheryan et al. 2017). Deshalb muss Inklusion auch Kinder und Jugendliche mit unterrepräsentiertem Geschlecht berücksichtigen. Traditioneller Unterricht kann somit Barrieren für alle Schüler*innen schaffen, nicht nur für solche mit sonder-

pädagogischem Förderbedarf (Stefik et al. 2019). Die Einstellungen der Lehrkräfte, strukturelle Bedingungen und Schulausstattung sind dabei entscheidend für den Abbau von Barrieren (Amrhein 2011). So benötigen Schulen unterstützende Technologien wie Text-to-Speech oder Screenreader, um barrierefreien Zugang zu gewährleisten (Capovilla 2019). Besonders beim Programmieren stoßen sehgeschädigte Personen oft auf Barrieren, die durch geeignete Hilfsmittel überwunden werden müssen (Moun-tapmbeme et al. 2022). Darüber hinaus ist das Verständnis der Fachsprache ein weiterer wichtiger Faktor für den Erfolg (Lampe et al. 2019) und muss so die Berücksichtigung finden.

Es ist wichtig, diese Barrieren zu berücksichtigen, um einen inklusiven Informatikunterricht zu ermöglichen. Daraus lässt sich ableiten, dass im Gegensatz zur Heterogenität die Inklusion nicht bei der Bildung aufhört, sondern auch soziale, kulturelle und ökonomische Teilhabe miteinschließt. Es betont die Notwendigkeit, von Anfang an gleiche Chancen für alle zu schaffen, was eine grundlegende Veränderung der Einstellungen und Strukturen in der Gesellschaft und im Bildungssystem erfordert. Ein inklusiver Ansatz muss die Überschneidung verschiedener Identitätsmerkmale wie Behinderung, Geschlecht oder Herkunft berücksichtigen und mehrfache Diskriminierungen abbauen.

Forschung zur Inklusion in der deutschen Didaktik der Informatik

Die Analyse des aktuellen Forschungsstands zur Inklusion in der deutschen Informatikdidaktik zeigt eine eklatante Lücke: In einer Google Scholar-Recherche zur Inklusion in der Informatik fanden sich unter den 50 relevantesten Treffern gerade einmal fünf direkt einschlägige Beiträge von vier verschiedenen Autor*innen. Bei der Untersuchung des Themas "Inklusion"¹ im Zusammenhang mit "Informatikunterricht" oder "informatische Bildung" wurden über Google Scholar insgesamt 306 Treffer erzielt. Aufgrund der Relevanz wurde die Anzahl der berücksichtigten Treffer auf die ersten 50 begrenzt. In der ersten Iteration wurden alle Workshop-Beiträge, Komplettbände, Beiträge zu Kompetenzen und Bildungsstandards, sowie solche, in welchen wichtige Information fehlte, oder doppelte Beiträge entfernt. Dadurch redu-

zierte sich die Anzahl auf 38 Treffer. In der zweiten Iteration wurden alle Beiträge, die tatsächlich Inklusion behandelten und interessant erschienen, weiter analysiert. Letztendlich blieben vier relevante Treffer übrig: Capovilla (2015), Capovilla (2019), Akao und Fischer (2021) sowie Hilbig (2022). Zusätzlich wurde die Literatur um den Beitrag von Akao und Fischer (2020) ergänzt, obwohl dieser nur gedruckt erhältlich ist. Dies zeigt bereits deutlich, dass es nur wenige relevante² Beiträge zur Inklusion im Informatikunterricht in Deutschland gibt. Das Forschungsfeld in der Didaktik der Informatik ist somit klein und speziell.

Capovilla (2015) stellt heraus, dass inklusive Bildung in der Informatik für sehgeschädigte Menschen durch spezialisierte Hard- und Software sowie angepasste Unterrichtsmethoden möglich ist. Evaluationsstudien zeigen, dass sehgeschädigte Personen mit Standardausrüstung Aufgaben in vergleichbarer Zeit wie Nicht-Sehgeschädigte lösen können. Zwei entwickelte Unterrichtskonzepte, Individualisierung und sensorische Parallelisierung³ mit haptischen Modellen, unterstützen den inklusiven Unterricht und machen abstrakte Konzepte greifbar.

Capovilla (2019) betont die Notwendigkeit, die allgemeine Didaktik der Informatik an die Lernvoraussetzungen an Menschen mit Behinderung anzupassen. Auch wird verdeutlicht, dass dafür das Universal Design for Learning (UDL) geeignet sein kann. Darüber hinaus ermöglichen assistive Technologien wertvolle alternative Bedienungskonzepte. Es wird keine spezielle Didaktik für Lernende mit Behinderung vorgeschlagen, sondern eine Anpassung der allgemeinen Informatikdidaktik. Informatiksysteme können als Mittel zur Förderung von Teilhabe dienen, wobei sensorische und kognitive Parallelisierung als inklusiver Ansatz und nicht als sonderpädagogische Maßnahme betrachtet wird.

Akao und Fischer (2020) führten 2019 eine Umfrage an 2123 Schulen in Nordrhein-Westfalen durch. Die Umfrage ergab, dass nur 37% Schulen inklusiven Informatikunterricht anbieten. Über die Hälfte der Lehrkräfte erwarb ihre Kenntnisse zur Inklusion nicht durch die formelle Lehramtsausbildung und viele hatten gar keine Kenntnisse. Lediglich 14% lernten im Lehramtsstudium etwas über Inklusion, und nur 35%

¹ Es wurde sich bewusst an dieser Stelle auf Inklusion beschränkt und Begriffe wie Heterogenität ausgeschlossen von der Suche, da dieser nicht weitreichend genug ist. Auch wurden bewusst nur deutsche Artikel ausgewählt, um den Forschungsstand in der Didaktik der Informatik Deutschland darzustellen.

² Zum Vergleich: Für den Sachunterricht liegen deutlich mehr als 100 Publikationen vor (Simon 2020)

³ Dies bedeutet, dass ein Unterrichtsinhalt beispielsweise sowohl durch Text und Bilder (visuell) als auch durch mündliche Erklärungen (auditiv) und taktile Modelle (haptisch) vermittelt wird.

nahmen an einer Fortbildung zu diesem Thema teil. Viele Lehrkräfte betreuen Schüler*innen mit besonderem Förderbedarf, ohne zu wissen, wie inklusiver Unterricht umgesetzt werden soll. Zudem stimmten 75% der Lehrkräfte nicht zu, dass aktuelle Lehrbücher oder Hilfsmittel für inklusiven Unterricht geeignet sind, und 62% fanden, dass es nicht genügend Fortbildungsangebote gibt. Diese Ergebnisse verdeutlichen den erheblichen Bedarf an verbesserten Schulungs- und Ausbildungsangeboten für Lehrkräfte im Bereich der inklusiven Bildung.

Akao und Fischer (2021) führten eine weitere Umfrage zur Umsetzung inklusionsorientierter Informatiklehramtsausbildung an Hochschulen durch, welche zeigt, dass das Thema „Inklusiver Informatikunterricht“ schrittweise in der Didaktik der Informatik (DDI) umgesetzt wird. Es fehlt jedoch an Dozierenden mit sonderpädagogischem Wissen. Von 34 Befragten gaben 19 an, dass das Thema zumindest teilweise umgesetzt wird und dass Inklusion in den letzten Jahren in DDI-Veranstaltungen an den meisten Hochschulen behandelt wird. Diversitätsaspekte wie Geschlecht, Lernschwierigkeiten, Hochbegabung, kultureller Hintergrund und sprachliche Kompetenzen werden in diesen Veranstaltungen thematisiert, während motorische/sensorische Beeinträchtigungen und die emotionale/soziale Entwicklung weniger behandelt werden. Die Hauptgründe für die Nicht-Umsetzung inklusiven Informatikunterrichts sind laut Befragten das Fehlen von DDI-Dozierenden mit sonderpädagogischem Wissen (71,43%), mangelnde wissenschaftliche Ideen (42,86 %) und zeitliche Einschränkungen in Vorlesungen (35,71 %). Nur drei von 34 Befragten gaben an, dass in ihrer Arbeitsgruppe eine Person mit sonderpädagogischem Wissen vorhanden ist, um das Thema ausreichend zu behandeln. Die Schlussfolgerung der Studie zeigt, dass ein großer Bedarf an der Weiterentwicklung der inklusionsorientierten Informatiklehramtsausbildung und der Fortbildung der DDI-Dozierenden besteht, um die bildungspolitischen Ziele zu erreichen.

André Hilbig (2022) stellt in seiner Arbeit dar, dass Barrieren sowohl in der Gestaltung des Unterrichts als auch in den Bildungsdokumenten entstehen können. In diesen Dokumenten gibt es Kompetenzen, die Barrieren darstellen können⁴. Mit Blick auf die Fachwissenschaft Informatik stellt sich die Frage, ob grafische Darstellungsformen ein essenzieller Teil des Fachs oder lediglich eine Möglichkeit zur Kommunikation über Modelle sind. Es gibt nur wenige Un-

tersuchungen und Bestrebungen, die sich mit der Inklusion im Informatikunterricht befassen und über einzelne spezielle Aspekte von Diversität hinausgehen. Nach Hilbig (2022) ist die Aufgabe der Fachdidaktik zwischen den fachlichen Ansprüchen, die durch Bildungsdokumente prüfbar vorgegeben werden, und der Teilhabe an Bildung zu vermitteln. Durch die Anpassung und teilweise Reduzierung von fachlichen Vorgaben sowie durch kompetenzorientierte Prüfungen kann die Teilnahme am Fachunterricht ermöglicht werden. Teilhabe an Bildung sollte jedoch mehr als nur Teilnahme bedeuten. Die Informatikdidaktik hat die Aufgabe, sowohl auf der Leitungsebene als auch auf der Ebene der konkreten Umsetzung einheitliche Konzepte zu entwickeln, gegebenenfalls in Anlehnung an das Universal Design for Learning. Dazu gehört die Überarbeitung und Prüfung der Bildungsdokumente, insbesondere wie informatische Modellierung im Kontext von Inklusion stärker expliziert werden kann.

Wie kann Inklusion im Informatikunterricht umgesetzt werden?

Die Umsetzung von Inklusion im Informatikunterricht erfordert spezifische Annahmen und Strategien, die auf die Bedürfnisse aller Lernenden eingehen. Laut Israel (2021) lauten die grundlegenden Annahmen für inklusiven Informatikunterricht wie folgt:

- Alle Lernenden verdienen es, sinnvoll in den Informatikunterricht einbezogen zu werden.
- Alle Lernenden können im Informatikunterricht erfolgreich sein.
- Die Diversität der Lernenden ist ein Gewinn für den Informatikunterricht.
- Der Informatikunterricht muss alle Lernenden ansprechen.

Diese Grundsätze bilden die Basis für eine inklusive Bildung im Bereich Informatik, die darauf abzielt, allen Schüler*innen unabhängig von ihren individuellen Voraussetzungen eine gleichberechtigte Teilnahme zu ermöglichen. Das Universal Design for Learning (UDL) ist ein didaktisches Rahmenkonzept, das auf diese Grundsätze abzielt.

Das Universal Design for Learning (UDL)⁵ wurde von David Rose und Anne Meyer vom Center for Applied Special Technology (CAST) entwickelt

⁴ Dazu gehört beispielsweise die Auseinandersetzung mit der Entwicklung eines grafischen Datenmodells (ER-Modell)

⁵ Weitere Information mit Anwendungsmöglichkeiten sind auf der Webseite des UDLs zu finden: <https://udlguidelines.cast.org/>

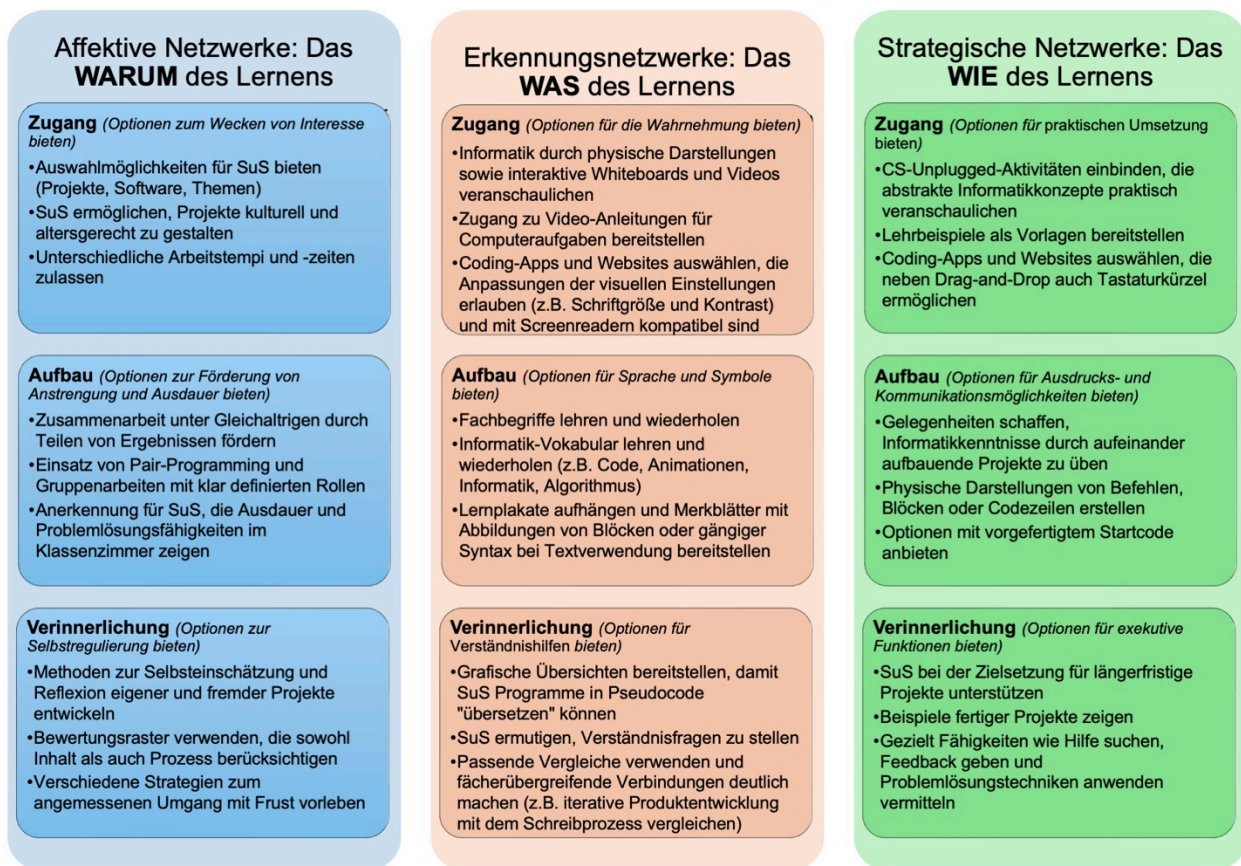


Abbildung 1: Übersicht vom Universal Design for Learning angepasst durch Israel (2017), übersetzt und tlw. gekürzt

und soll im Folgenden detailliert besprochen werden (CAST 2018). UDL basiert auf einer Vielzahl von Lerntheorien und betont Flexibilität in Unterrichtsmethoden und Lernumgebungen, um Barrieren zu beseitigen und auf unterschiedliche Bedürfnisse einzugehen. Besonders im Bereich der Inklusion und Diversität könnte sich UDL als geeignet für die Erreichung dieser Ziele erweisen (Capovilla 2019).

UDL dient als Blaupause und Reflexionsmöglichkeit für Lehrende. Ziel ist es, die Lernenden zu Expert*innen ihres eigenen Lernens zu machen, die planvoll und motiviert vorgehen, einfallsreich und kenntnisreich sind und strategisch und zielgerichtet lernen. Das didaktische Rahmenkonzept von UDL basiert auf Erkenntnissen der Neurowissenschaft über Lernprozesse und unterscheidet drei Arten von Lernbedürfnissen: Wahrnehmung und Erkenntnis (das "Was" des Lernens), Handlung und Ausdruck (das "Wie") sowie Motivation und Verhalten (das "Warum"). Im Folgenden wird das UDL durch die Anmerkungen von Israel (2019) und zusätzlichen Details aus Wille et al. (2017) ergänzt.

Die affektiven Netzwerke konzentrieren sich auf die Motivation und das Engagement der Ler-

nenden. UDL bietet Optionen, um das Interesse der Schüler*innen zu wecken, etwa durch die Auswahl von Projekten, Software oder Themen, die kulturell und altersgerecht gestaltet sind. Schüler*innen mit Aufmerksamkeitsstörungen können Schwierigkeiten haben, Aufgaben zu beginnen oder motiviert zu bleiben. Anpassungen wie die Wahl interessanter Aufgaben oder der Einsatz von Motivationsstrategien helfen, ihre Aufmerksamkeit aufrechtzuerhalten. Die Zusammenarbeit unter Gleichaltrigen, etwa durch Pair-Programming, ist besonders effektiv für Schüler*innen mit Lernschwierigkeiten. Diese Methode fördert Verantwortungsbewusstsein, soziale und technische Fähigkeiten und motiviert die Schüler*innen durch gegenseitige Unterstützung und gemeinsames Problemlösen. Ein unterstützendes Umfeld zu schaffen, in dem jede*r die Möglichkeit hat, sich einzubringen, ist somit wichtig. Schüler*innen, die Schwierigkeiten beim Erkennen sozialer Hinweise haben, profitieren davon, wenn Lehrkräfte während der Gruppenarbeiten aktiv durch den Raum gehen und ihnen helfen, passende Phrasen oder Ausdrucksweisen zu finden. Dies ist besonders wichtig, da Schüler*innen mit sozialen Schwierigkeiten oft Probleme haben, soziale Signale zu erkennen, was sich negativ auf

die Zusammenarbeit auswirken kann. Beispielsweise könnten sie Schwierigkeiten haben zu erkennen, wenn sie zu viel reden, andere unterbrechen oder unbewusst scharfe Kommentare machen.

Die Erkennungsnetzwerke befassen sich mit der Wahrnehmung und dem Verständnis von Informationen. UDL empfiehlt, verschiedene Präsentationsmethoden einzusetzen, um Lerninhalte zugänglich zu machen, etwa physische Darstellungen und interaktive Whiteboards. Vor allem Schüler*innen mit Lesestörungen profitieren von diesen Ansätzen, da sie oft Schwierigkeiten haben, die Lautstruktur der Sprache zu erkennen oder Wörter schnell abzurufen. Wichtig ist dabei, Schlüsselwörter und Phrasen hervorzuheben und wichtige Informationen in mehreren Formaten bereitzustellen, nicht nur in Textform. Ein zugängliches Glossar mit den relevanten Wörtern für jede neue Lektion sowie die gezielte Überprüfung dieser Wörter in neuen Kontexten, wie z.B. in einer Programmierumgebung, sind ebenfalls hilfreich. Vielseitige Präsentationsmöglichkeiten wie Videoanleitungen für Programmieraufgaben ermöglichen es ihnen, in ihrem eigenen Tempo zu lernen und komplexe Aufgaben besser zu verstehen. Anpassbare visuelle Einstellungen in Coding-Apps und Webseiten sowie Lernplakate mit Abbildungen von Programmierblöcken unterstützen zusätzlich. Schüler*innen, besonders jene mit Sprachstörungen, hilft dies beim Verständnis und bei der Anwendung komplexer Konzepte, da sie möglicherweise über einen eingeschränkten Wortschatz verfügen oder Schwierigkeiten haben, die Bedeutung von Wörtern zu verstehen.

Die strategischen Netzwerke konzentrieren sich auf die praktische Umsetzung und die Ausdrucksmöglichkeiten der Lernenden. UDL schlägt vor, abstrakte Konzepte durch praktische Aktivitäten, wie CS-Unplugged, zu veranschaulichen. Physische Darstellungen von Befehlen und Blöcken erleichtern es den Schüler*innen, abstrakte Konzepte besser zu verarbeiten. Schüler*innen mit Rechenstörung profitieren von diesen praktischen, visuellen Darstellungen, da sie ihnen helfen, abstrakte informatische Konzepte greifbarer zu machen. Vorbereitete Code-Optionen ermöglichen es den Schüler*innen, ihre Programmierfähigkeiten schrittweise zu entwickeln, ohne von komplexen Aufgaben überfordert zu werden. Gezieltes Feedback und strukturierte Projekte unterstützen Schüler*innen mit Exekutivfunktionsstörungen bei Zielsetzung, Planung und Zeitmanagement. Schüler*innen mit Schreibstörungen, die oft Schwierigkeiten haben, ihre Gedanken klar zu Papier zu bringen, profitieren von struk-

turiertem Feedback und klaren Diskussionsregeln. Dies hilft ihnen, ihre Gedanken auszudrücken und aktiv am Lernprozess teilzunehmen. Klare, strukturierte Projekte und regelmäßiges Feedback tragen somit dazu bei, dass die Schüler*innen ihre Lernprozesse erfolgreich steuern und abschließen.

Mögliche Ansätze zur Umsetzung von inklusivem Unterricht lassen sich bspw. bei der Erweiterung des Spioncamps⁶ der Universität Wuppertal finden, wo im Rahmen einer Masterarbeit Stationen zur Kryptologie und Kryptographie entwickelt wurden, die speziell für den Einsatz im inklusiven Informatikunterricht konzipiert sind.

Herausforderungen und Handlungsempfehlungen

Inklusion im Bereich der Informatikdidaktik ist bisher wenig erforscht. Dies wird deutlich an der geringen Anzahl von Veröffentlichungen und Projekten, die sich speziell mit inklusivem Informatikunterricht beschäftigen. Im Vergleich dazu ist die Inklusionsforschung in anderen Bereichen, wie dem Sachunterricht, deutlich weiterentwickelt und besser dokumentiert. Ein Problem sind fehlende Ressourcen und die unzureichende Ausbildung der Lehrkräfte. Viele Lehrkräfte im Informatikunterricht haben keine spezielle Schulung in inklusiver Pädagogik erhalten. Sie stehen zudem unter hohem Zeitdruck, was es ihnen schwer macht, geeignete Materialien zu entwickeln und im Unterricht einzusetzen. Hinzu kommt, dass viele Lehrkräfte das Fach Informatik fachfremd unterrichten müssen. Es mangelt sowohl an Fortbildungen als auch an inklusiven Lehrmaterialien, die den Lehrkräften zur Verfügung gestellt werden können. Die wenigen vorhandenen Studien, wie die von Akao und Fischer (2020), unterstreichen den dringenden Bedarf an gezielten Schulungs- und Ausbildungsangeboten für Lehrkräfte, um den Anforderungen eines inklusiven Informatikunterrichts gerecht zu werden.

In anderen Fächern, wie dem Sachunterricht, wurde bereits früh mit dem Thema Inklusion gearbeitet. Vertreter*innen aus der Sachunterrichtsdidaktik haben sowohl in der Theorie als auch in der Praxis bedeutende Beiträge zur Inklusion geleistet. Zudem gibt es eine spezielle Arbeitsgruppe im Sachunterricht, die sich intensiv mit Inklusion beschäftigt und bereits drei Fachtagungen zu diesem Thema organisiert

⁶ Weitere Information unter <https://ddi.uni-wuppertal.de/web/website/materialien/spioncamp.html>

hat (Simon, 2020). Pech und Kollegen (2018) betonen, dass theoretische Diskussionen durch empirische Forschung weiter untermauert und entwickelt werden müssen. Es gibt daher zahlreiche Forschungsprojekte und Veröffentlichungen, die sich mit der Umsetzung inklusiver Bildung im Sachunterricht befassen. Dadurch wird der Sachunterricht zu einem Vorreiter in der Inklusionsforschung (Kahlert, 2015).

Erfolgreiche Ansätze aus anderen Fachbereichen können auch in der Informatikdidaktik Anwendung finden. Ein Beispiel dafür ist das Universal Design for Learning (UDL), das flexible Lernumgebungen und differenzierte Unterrichtsmethoden fördert, um den unterschiedlichen Bedürfnissen aller Lernenden gerecht zu werden (Capovilla, 2019). Um diese Prinzipien effektiv umzusetzen, ist die Entwicklung umfassender Konzepte und Materialien notwendig, die alle Lernenden ansprechen. Gleichzeitig sind weitere Forschungsprojekte erforderlich, um diese Ansätze zu testen und anzupassen. Die Informatikdidaktik kann hierbei von der Sachunterrichtsdidaktik lernen und ähnliche Forschungsprojekte starten, um die theoretischen Grundlagen und praktischen Möglichkeiten inklusiver Bildung zu erweitern. Um die Inklusion in der Informatik erfolgreich voranzubringen, sind gezielte Maßnahmen auf verschiedenen Ebenen notwendig:

Auf der Praxisebene sollten inklusive Unterrichtsmaterialien und Aufgaben entwickelt und verbreitet werden. Assistive und adaptive Technologien müssen verstärkt genutzt werden, und es sollten fortlaufende Fortbildungsangebote für Lehrkräfte im Bereich der inklusiven Didaktik geschaffen werden. In der Aus- und Fortbildung von Lehrkräften ist es wichtig, Themen wie Inklusion und Diversität dauerhaft in der Lehramtsausbildung zu verankern. Eine stärkere interdisziplinäre Zusammenarbeit zwischen den verschiedenen Fachdidaktiken ist hierbei entscheidend. Dozententeams sollten sonderpädagogische Expertise einbinden, und es sollten Kooperationen zwischen der Informatikdidaktik und anderen Fachdidaktiken gefördert werden. Auf Forschungsebene sollten Projekte initiiert werden, die inklusionsorientierte didaktische Konzepte erproben und weiterentwickeln. Es gilt, diagnostische Instrumente zu entwickeln und zu validieren sowie die Bildungsstandards kritisch auf Inklusion hin zu analysieren. Bildungspolitisch müssen die Bildungsstandards überprüft und gegebenenfalls angepasst werden, um potenzielle Barrieren abzubauen. Finanzielle Förderprogramme für Inklusionsprojekte in der Informatik sollten bereitgestellt werden, ebenso wie die notwendigen

personellen und infrastrukturellen Ressourcen. Eine erfolgreiche inklusive Bildung in der Informatik kann nur durch abgestimmte Maßnahmen auf allen Ebenen gelingen. Inklusion ist eine gesamtgesellschaftliche Aufgabe, die das Engagement aller Beteiligten erfordert – von der Politik über Bildungseinrichtungen und Lehrkräfte bis hin zu Forschenden und nicht zuletzt den Lernenden selbst.

Fazit und Ausblick

Ein inklusiver Informatikunterricht, der den wachsenden Anforderungen durch Diversität gerecht wird, hat für alle Seiten Potenziale: Alle Schüler*innen erhalten bessere Bildungschancen und Perspektiven. Für die Informatikbranche bedeutet mehr Inklusion auch mehr Leistungsfähigkeit durch neue Ideen und Sichtweisen. Und für die Gesellschaft insgesamt ist Inklusion ein Schritt zu mehr Chancengerechtigkeit und Teilhabe. Allerdings zeigen die Recherchen einen erheblichen Mangel an Ressourcen, Qualifizierung und Forschung für die Umsetzung inklusiver Bildung in der Informatik. Es bedarf entschlossener Maßnahmen in den Bereichen Lehramtsausbildung, Unterrichtsmaterialien, Fortbildungsangebote sowie interdisziplinärer Forschung und Austausch. Inklusion muss in allen bildungspolitischen und strukturellen Vorgaben verankert werden. Nur so können wir das Potenzial eines inklusiven, diversitätsorientierten und chancengerechten Informatikunterrichts für alle Lernenden erschließen.

Quellen

Alle Webseiten/Links wurden zuletzt geprüft am 31.05.2024.

Akao, K. und Fischer, J. (2020). Wie läuft die Umsetzung inklusiven Informatikunterrichts tatsächlich? – Eine Lehrerumfrage zum inklusionsorientierten Unterricht. In Thomas, M. und Weigend, M. (Hrsg.), *Mobil mit Informatik*, pages 9–18. BoD, Norderstedt.

Akao, K. und Fischer, J. (2021). Zum Stand der Lehramtsausbildung für einen inklusiven Informatikunterricht. INFOS 2021 – 19. GI-Fachtagung Informatik und Schule.

Amrhein, B. (2011). *Inklusion in der Sekundarstufe: eine empirische Analyse*. Klinkhardt Forschung. Verlag Julius Klinkhardt.

Capovilla, D. (2015). *Inklusion in der Informatischen Bildung am Beispiel von Menschen mit Sehschädigung*. München: TUM School of Education. PhD thesis.

Capovilla, D. (2019). Informatische Bildung und inklusive Pädagogik. In A. Pasternak (Hrsg.) Informatik für alle - 18. GI-Fachtagung Informatik und Schule (S. 35-48). Bonn: Köllen Druck + Verlag GmbH.

CAST (2018). Universal Design for Learning Guidelines version 2.2. Retrieved from <http://udlguidelines.cast.org>.

Cheryan, S., Ziegler, S. A., Montoya, A. K. und Jiang, L. (2017). Why are some stem fields more gender balanced than others? *Psychological Bulletin*, 143:1–35.

Dirks, S. (2022). Inklusion im informatikunterricht. In Thomas, M. und Weigend, M. (Hrsg.), *Inklusion mit Informatik*. 10. Münsteraner Workshop zur Schul informatik, pages 7–8. Universität Münster.

Hilbig, A. (2022). Diversität im informatikunterricht als Gestaltungsaufgabe der Fachdidaktik. In Thomas, M. und Weigend, M. (Hrsg.), *Inklusion mit Informatik*. 10. Münsteraner Workshop zur Schul informatik, pages 11–20. Universität Münster.

Hong, H., Wang, J., Ravitz, J. und Fong, M. L. (2015). Gender differences in high school students' decisions to study computer science und related fields. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, page 689, Kansas City, MI. ACM.

Israel, Maya; Last, Todd; Jeong, Gakyung; Wherfel, Quentin M. (2017): *Project TACTIC: Teaching All Computational Thinking through Inclusion and Collaboration. Helpful Strategies for Utilizing the Universal Design For Learning Framework in Computer Science Education*. TACTICal Teaching Brief.

Israel, M. (2021). Equity principles for including learners with disabilities in k-12 cs education. In *Understanding Computing Education (Vol 2): Equity, Diversity and Inclusion*. *Proceedings of the Raspberry Pi Foundation Research Seminars*. Raspberry Pi Foundation.

Kahlert, J. (2016). *Der Sachunterricht und seine Didaktik*. Klinkhardt, Bad Heilbrunn.

KMK (2022), *Ständige Konferenz der Kultusminister der Länder in der Bundesrepublik Deutschland. Sonderpädagogische Förderung in Schulen*.

Lampe, Timo & Diethelm, Ira. (2019). *Transkriptanalyse einer Informatik-Unterrichtsstunde*. 10.1007/978-3-658-27168-8_8.

Miesenberger, K. (2015). Advanced and Emerging Solutions: ICT and AT in Education of Low Vision and Blind Students. In *Proceedings of ICEAPVI'15*, pages 17–26, Athen.

Mountapmbeme, A., Okafor, O. und Ludi, S. (2022). Addressing Accessibility Barriers in Programming for People with Visual Impairments: A Literature Review. *ACM Trans. Access. Comput.*, 15(1).

Pech, D., Schomaker, C. und Simon, T. (2018). Literaturübersicht zum Zusammenhang Inklusion und Aachunterricht. In Pech, D., Schomaker, C. und Simon, T. (Hrsg.), *Sachunterrichtsdidaktik & Inklusion*. Ein Beitrag zur Entwicklung, pages 124–133. Schneider, Baltmannsweiler.

Saalfrank, W.-T. und Zierer, K. (2017). *Inklusion*. Number 4541 in *UTB Pädagogik, Erziehungswissenschaft, Bildungswissenschaft*. Ferdinand Schöningh, Paderborn.

Simon, T. (2020). Sachunterricht(sidaktik) auf dem Weg zur Inklusion? Rück-, Ein- und Ausblicke. *k:ON - Kölner Online Journal für Lehrer*innenbildung*, 2(2, 2/2020), 70–93. <https://doi.org/10.18716/ojs/kON/2020.2.04>

Stefik, A., Ladner, R. E., Allee, W. und Mealin, S. (2019). *Computer Science Principles for Teachers of Blind and Visually Impaired Students*. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education, SIGCSE '19*, page 766–772, New York, NY, USA. Association for Computing Machinery.

Textor, A. (2015). *Einführung in die Inklusionspädagogik*. UTB. 4340. Schulpädagogik, Sonderpädagogik. Klinkhardt, Bad Heilbrunn.

Wille, S., Century, J. und Pike, M. (2017). Exploratory Research to Expand Opportunities in Computer Science for Students with Learning Differences. *Computing in Science Engineering*, 19(3):40–50.

Lizenz



Dieser Artikel steht unter der Lizenz CC BY NC 4.0 zur Verfügung.

Kontakt

Rina Ferdinand, Mareike Daeglau, Ira Diethelm
Carl von Ossietzky Universität Oldenburg
{rina.martina.ferdinand, mareike.daeglau, ira.diethelm@uni-oldenburg.de}

Informatik in freier Wildbahn: Lerntransfer vom Unterricht in den Alltag

Rücker, M. T.

DOI: 10.18420/ibis-02-02-05

Zusammenfassung

Nicht für die Schule, sondern für das Leben soll gelernt werden. Um diesem Anspruch gerecht zu werden, müssen Lernende in die Lage versetzt werden, ihr schulisches Wissen und Können auch außerhalb und nach der Schule – in der freien Wildbahn – anzuwenden. Diese Herausforderung ist kaum zu unterschätzen. Für die Informatik müssen sie dazu u.a. künftige und neue Systemkategorien erschließen und Manifestationen bekannter informatischer Konzepte im Alltag erkennen können. Wenn im Unterricht bspw. eingebettete Systeme, neuronale Netze oder Datenbanken behandelt werden, gilt es, diese im Alltag auch wiederzuerkennen. Andernfalls wäre im Ergebnis eher „totes Wissen“ entstanden. Auf Basis kognitionspsychologischer Grundlagen sowie drei fachdidaktischer Studien werden im vorliegenden Artikel Ansätze vorgestellt, entsprechende Transferprozesse im Informatikunterricht zu fördern.

Einleitung

„Nicht für das Leben, sondern für die Schule lernen wir!“ Diese praktisch zeitlose Kritik des Philosophen Seneca haben die meisten wohl schon einmal gehört. Seine Forderung, das in der Schule erworbene Wissen und Können müsse auch im Leben nach der Schule anwendbar sein, ist insbesondere für den allgemeinbildenden (Informatik)Unterricht unmittelbar einleuchtend. Ein sehr naheliegender Ansatz ist, im Unterricht einen Bezug zu eben dieser außerschulischen und alltäglichen Lebenswelt der Schüler:innen herzustellen, indem Inhalte in geeignete Kontexte eingebettet sowie mit Beispielen und Phänomenen aus dem Alltag in Bezug gesetzt werden (Diethelm & Dörge, 2011). Ziel ist es einerseits, die Lernenden zu motivieren, indem an ihre unmittelbaren Alltagserfahrungen und Interessen angeknüpft wird. Andererseits soll die Fähigkeit gefördert werden, sich ebensolche alltäglichen Kontexte und Phänomene später eigenständig erschließen zu können (vgl. Humbert & Puhlmann, 2004). Bei dem dabei angestrebten Lerntransfer von der Schule in den Alltag handelt es sich jedoch um eine kognitive Herausforderung, deren Härte kaum zu unterschätzen ist.

Stellen Sie sich vor, es ist ein typischer Schultag (sofern es so etwas gibt) und der Informatikunterricht beginnt in wenigen Minuten. Schon aufgrund des Stundenplans ist allen Beteiligten klar, dass das, was jetzt und hier in den nächsten Minuten passiert, Informatik ist. Informatik ist jetzt *dran*. Die Lernenden sind also von vornherein dafür sensibilisiert, ihr informatisches Wissen und Können im Unterricht zur Anwendung zu bringen. Wer jetzt leise in sich hinein lacht und denkt: „Schön wär’s!“, hat recht. In der Regel braucht es dazu etwas mehr. Es bedarf einer gezielten Impulssetzung, eines geeigneten Stundeneinstiegs, der es vermag, das nötige Wissen und Können zunächst zu reaktivieren und in der Situation wieder verfügbar zu machen – und sei es nur das der letzten Stunde. Kognitionspsychologisch handelt es sich dabei bereits um die Herbeiführung von Lerntransfer. Zuvor Gelerntes soll reaktiviert und in der aktuellen, neuen Situation verwendet werden. Dass diese Aktivierung hier mit einer gewissen Hilfestellung abläuft und dass die neue Situation nun wieder eine Lernsituation ist, ist dabei zunächst nicht weiter relevant (Bransford & Schwartz, 1999; Lobato, 2012).

Machen wir nun den gedanklichen Sprung hinaus in den Alltag – in die freie Wildbahn – dann müssen wir im Vergleich feststellen, dass all die oben beschriebenen Hilfsmittel dort ersatzlos wegfallen. Dort gibt es keinen Stundenplan, der klar vorgibt, wann welche Fachdisziplin *dran* ist. Es gibt sich auch niemand Mühe, einen sinnvollen Impuls zu setzen, der helfen soll, das konkret nötige Wissen und Können zu reaktivieren. Diese Prozesse müssen im Alltag nicht nur selbstständig, sondern auch gänzlich spontan ablaufen. Führt man sich nun vor Augen, was für eine enorme Herausforderung dies in der Regel bereits im Unterricht darstellt (wie schnell geht selbst der am besten geplante Einstieg mal daneben!), bekommt man ein Gefühl dafür, was für eine immense kognitive Leistung solch ein Transferakt im Alltag eigentlich darstellt. Welche Ansätze gibt es also, Alltagstransfer im Informatikunterricht gezielt zu fördern?

Induktive Schlüsse und Kontextuelle Vielfalt

Menschliches Denken erfolgt in großen Teilen entlang von Kategorien, welche kognitionspsychologisch kaum von „Konzepten“ zu trennen sind (Goldstone et al., 2018). Kategorisierung ist essenziell für den Lerntransfer, denn sie „erlaubt es, bereits vorhandenes Wissen auf neue Erfahrungen anzuwenden“ (Waldmann, 2008, S. 378). Wenn Sie bspw. ein neues Smartphone sehen, müssen Sie es nicht erst im Detail untersuchen, um davon überzeugt zu sein, dass es über ein Betriebssystem, einen Touch-Screen oder ein Funkmodul verfügt. Die Kategorisierung des Objektes *als Smartphone* sowie entsprechendes Vorwissen über diese Gerätekategorie reichen aus, um solch *deduktive Schlüsse* zu ziehen. Lerntransfer basiert dabei auf dem kognitiven Akt, eine Sache einer Kategorie zuzuordnen (Ist das hier ein Smartphone?), wodurch Dinge, die zuvor über die Kategorie gelernt wurden, darauf anwendbar werden.

Bedauerlicherweise ist unser alltägliches Leben nicht in eine endliche, geschweige denn überschaubare und wohldefinierte Menge von Kategorien zerlegbar. Ab wann gilt etwas eigentlich als Smartphone und wann hört es auf, eins zu sein? Gerade für die Informatik und ihre stetigen Weiter- und Neuentwicklungen ist dies problematisch und eine Fokussierung auf aktuelle Technik-Kategorien daher wenig nachhaltig (vgl. Schwill, 1997). Ein möglicher Ausweg wäre, die Kategorien abstrakter zu machen und statt über *Smartphones* z.B. über *Informatiksysteme* nachzudenken. Dies stellt uns – bzw. die Lernenden – dann aber vor die analoge Frage: Wann ist etwas denn ein Informatiksystem und woran erkennt man das im Alltag? Dazu später mehr.

Tatsächlich erfolgt Transfer im Alltag oft gar nicht deduktiv, sondern *induktiv*, vor allem dann, wenn Wissen von bekannten auf neue Dinge und Kategorien transferiert werden soll. Induktion beschreibt hier den kognitiven Prozess,¹ generalisierende Schlüsse aus einer Menge bekannter Beispiele zu ziehen (Osherson et al., 1990). Wenn Sie z.B. bereits wissen, dass Smartphones und Tablets über Funkmodule verfügen, liegt die Vermutung nahe, dass dies auch für Smart Watches oder sogar allgemein für Handhelds oder Wearables gilt. Auch ein Kleinkind, das versucht, einen „normalen“ Monitor über Touch-Gesten zu bedienen, tut dies sehr wahrscheinlich auf Basis eines (hier leider

falschen) induktiven Schlusses: Alle anderen bisher bekannten Bildschirme reagierten auf Berührungen, also wird es dieser hier wohl auch tun. Induktive Schlüsse sind *empirisch* und somit immer mit einer gewissen Unsicherheit verbunden.

Wie überzeugt eine Person von einem induktiven Schluss ist, hängt vor allem von den bereits bekannten Beispielen ab. Zunächst ist eine möglichst große Menge bekannter Beispiele natürlich vorteilhaft. Doch auch ihre konkrete Beschaffenheit und Zusammensetzung ist relevant. Vergleichen Sie einmal die beiden fiktiven (bewusst nicht informatischen) Szenarien in Tabelle 1 und stellen Sie sich folgende Frage: In welchem der beiden Szenarien erscheint es Ihnen wahrscheinlicher, dass auch Pinguine über einen Pepsin-Stoffwechsel verfügen?

Szenario A	Szenario B
Es ist bekannt, dass Spatzen, Tauben und Krähen über einen Pepsin-Stoffwechsel verfügen.	Es ist bekannt, dass Schwäne, Strauße und Kolibris über einen Pepsin-Stoffwechsel verfügen.
Induktiver Schluss: Auch Pinguine verfügen über einen Pepsin-Stoffwechsel.	

Tabelle 1: Zwei Gedankenspiele.

Wenn Sie die Schlussfolgerung in Tabelle 1 in Szenario B für wahrscheinlicher halten, sind Sie in guter Gesellschaft. Ein Faktor dafür sind sogenannte Diversitätseffekte (Butler et al., 2017; Feeney & Heit, 2011; Hahn et al., 2005). Je verschiedener, je diverser, die Prämissen, desto wahrscheinlicher erscheint im Allgemeinen der induktive Schluss. Spatzen, Tauben und Krähen sind sich im Vergleich zu Pinguinen relativ ähnlich. Schwäne, Strauße und Kolibris hingegen unterscheiden sich nicht nur optisch stärker, sondern leben auch in sehr verschiedenen Gegenden, ernähren sich unterschiedlich etc. Dies macht sie zu einer deutlichen besseren – weil diverseren – Ausgangslage für Transfer.

Für die Didaktik der Informatik ergibt sich hier eine Herausforderung, denn die Diversität der Beispiele speist sich aus sogenannten *Oberflächenmerkmalen*. Spatzen, Tauben, Kolibris, Strauße und Pinguine sind alle Vögel. Sie unterscheiden sich lediglich in Merkmalen, die für diese Kategorie nicht definierend sind: Lebensraum, Körperform oder Fressverhalten. Analog dazu unterscheiden sich Unterrichtsbeispiele für Von-Neumann-Rechner gerade nicht in den definierenden Merkmalen der Von-Neumann-Architektur. Sonst wären sie ja keine Beispiele dafür! Stattdessen unterscheiden sie sich viel-

¹ Nicht zu verwechseln mit dem mathematischen Beweisverfahren der vollständigen Induktion, bei dem es sich logisch gesehen eigentlich um ein deduktives Schlussverfahren handelt.

leicht in Merkmalen wie Anwendungskontext, Peripherie, Rechenleistung oder Speicherkapazität. Genau solche zeitlich kurzlebigen Merkmale sind es jedoch, die im Sinne der fundamentalen Ideen der Informatik (Schwill, 1997) oft als nicht nachhaltig und nicht transferfähig abgetan werden und für die Unterrichtsgestaltung daher als weitgehend beliebig austauschbar oder sogar potenziell hinderlich für Transfer angesehen werden (vgl. Guzdial, 2010). Kognitionspsychologisch ist jedoch davon auszugehen, dass die Präsenz und die Beschaffenheit der Beispielkontexte eine essenzielle Rolle sowohl für den Erfolg von Lern- (Belenky & Schalk, 2014; diSessa, 2018; Knobelsdorf & Tenenbergs, 2013) als auch von Transferprozessen (Alfieri et al., 2013; Hammer et al., 2005; Lobato, 2012; Wagner, 2006) spielen.

Die vielleicht etwas unbequeme Erkenntnis hier ist, dass es für eine Förderung des Transfers allgemeiner Konzepte und Ideen eben *nicht* egal ist, anhand welcher konkreten und potenziell kurzlebigen Kontexte und Beispiele sie erlernt werden. Denn durch sie erhalten die Lernenden überhaupt erst einen Eindruck davon, wie transferfähig das Konzept eigentlich ist. Ihre Auswahl sollte daher bewusst unter Berücksichtigung relevanter diversitätsstiftender Oberflächenmerkmale erfolgen.

Was aber sind *relevante* Oberflächenmerkmale von technischen Systemen? Die Wahrnehmung ist an dieser Stelle subjektiv und kann für die Fachdidaktik daher nur empirisch durch die Untersuchung von Schülerwahrnehmungen erörtert werden. Genau das haben wir in einer Interviewstudie getan, in der wir Schüler:innen mit Bildern technischer Geräte konfrontiert haben und sie gebeten haben, „Dinge, die zusam-

men gehören“ in Gruppen zu sortieren, wobei es ihnen frei stand, wie viele Gruppen sie erstellen (Rücker & Pinkwart, 2018). Abbildung 1 zeigt die Gruppen eines Schülers der 11. Klasse, der einen Informatik-Leistungskurs besuchte. Die Bezeichnungen stammen ebenfalls von ihm.

In der Auswertung der insgesamt 163 Gruppen, die die 21 Teilnehmer:innen erstellten, entstand die in Abbildung 2 gezeigte Taxonomie. Sie unterteilt „Technik“ in vier Domänen: Unterhaltung, Haushalt, Öffentlichkeit und „echte“ Technik. Auffallend ist, dass die Hierarchie sehr einseitig aufgefächert ist. Tatsächlich bewegen sich die vier Domänen entlang eines Spektrums von der unmittelbaren Alltagsrelevanz der Schüler:innen (links) zu Bereichen, mit denen sie im Alltag eher weniger bis nie direkten Kontakt haben (rechts). Daher ist es nur plausibel, dass die Bereiche zur linken Seite weiter ausdifferenziert sind.

Wie erwähnt handelt es sich hier um eine Rekonstruktion subjektiver Wahrnehmungen. Entsprechend erhebt Abb. 2 keinen Anspruch auf Objektivität oder Vollständigkeit. Das Modell bietet aber eine empirisch gestützte Reflexionsgrundlage für die Auswahl möglichst heterogener Unterrichtsbeispiele und -kontexte. Natürlich ist es unrealistisch in jeder Stunde alle hier gezeigten Kategorien abzudecken. Aber wenn Sie das nächste Mal eine längere Reihe oder ein Halbjahr planen, wenn Sie mögliche Szenarien für ein Arduino-Projekt oder Beispielanwendungen für das EVA-Prinzip, den Von-Neumann-Rechner oder für maschinelles Lernen formulieren: Achten Sie auf Heterogenität in den Kontexten! Je vielfältiger diese gewählt werden, desto besser die Grundlage für einen späteren Lerntransfer.



Abbildung 1: Frei gelegte Gruppierungen eines Schülers (11. Kl., Informatik-LK)

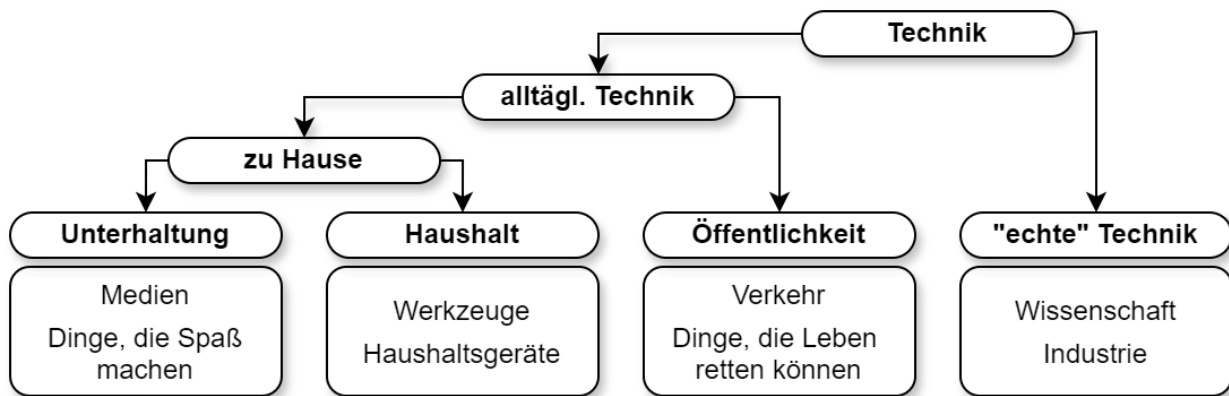


Abbildung 2: Techniktaxonomie basierend auf den von den Schüler:innen gelegten Gruppierungen

Deduktive Schlüsse und Identifikationsprozesse

Kommen wir noch einmal zurück zu den deduktiven Schlüssen. Wenn wir den Lernenden im Unterricht also etwas über abstrakte Kategorien wie *Computer*, *Programmierung* oder *Informatik* beibringen, dann stellt sich für einen anschließenden Transfer die Frage, wie und woran man diese denn im Alltag eigentlich erkennt. Auch hierzu haben wir Schüler:innen dieselben Bildkarten von technischen Geräten präsentiert und sie diesmal explizit darum gebeten, Dinge zusammen zu legen, die programmiert sind, die eine Art von Computer enthalten oder etwas mit Informatik zu tun haben (Rücker & Pinkwart, 2019). Die drei Gruppen wurden nacheinander und in variabler Reihenfolge gebildet. Jedes Gerät konnte also keiner, einer, zwei oder allen drei Gruppen zugeordnet werden. Abbildung 3 zeigt die drei Gruppen einer Schülerin der 9. Klasse, die einen Informatik-Wahlpflichtkurs besuchte.

Die Gruppe der programmierten Dinge ist klar die größte und enthält mehrere Geräte (Ampel, Waschmaschine, Radio, Digitalkamera, Raspberry Pi), die in den Augen der Schülerin trotz dieser Programmierung weder einen Computer noch Informatik enthielten. Dies war kein

Einzelfall. Im Schnitt enthielten die Programmiergruppen drei Elemente mehr als die anderen beiden. Am häufigsten betroffen waren dabei die Waschmaschine, die Ampel, der Fernseher und der Industrieroboter.

Doch wie haben die Schüler:innen dies begründet? In der Analyse der Argumentationen ergab sich, dass die große Mehrheit zwischen drei Stufen funktionaler Mächtigkeit zu unterscheiden schien, welche wir *Informatik*, *Elektronik* und *(Elektro)Mechanik* genannt haben. Tabelle 2 erläutert die drei Stufen sowie die Fähigkeiten, technischen Elemente und Geräte, die damit typischerweise assoziiert wurden. Es sei nochmal darauf hingewiesen, dass es sich auch hier um eine Rekonstruktion von Schülervorstellungen und nicht notwendigerweise um fachlich korrekte Kategorisierungen handelt.

Die Ebene der *(Elektro)Mechanik* enthält Technik, die als sehr limitiert in ihren Fähigkeiten wahrgenommen wurden. Entsprechende Geräte wurden in der Regel aus allen drei Gruppen ausgeschlossen. Die zweite Ebene der *Elektronik* erlaubt bereits rudimentäre Logik wie Zeitmessung, Zählen, bedingtes Verhalten oder die Ausführung „vorprogrammierter“ Einstellungen. Für viele Schüler:innen erforderten solche Funktionen zwar eine Programmierung, waren aber dennoch nicht ausreichend, um dahinter



Abbildung 3: Computer-, Programmierung- und Informatik-Gruppen einer Schülerin der 9. Klasse (Inf-WP)

	Was es kann	Was „drin steckt“	typische Vertreter
Informatik	Intelligenz, Vielseitigkeit, Adaptivität, Vernetzung	Software, Computer, System	Smartphone, Spielkonsole, Laptop, PC
Elektronik	Zeitmessung, bedingtes Verhalten, fixe Einstellungen	Programm, Platine, Schaltkreis	Ampel, Waschmaschine, Radio, Industrieroboter
(Elektro-) Mechanik	„nichts“, an und aus, simple Bewegung	„nichts“, Kabel, Motor	Bohrmaschine, Wecker, Taschenmesser, Rasenmäher

Tabelle 2: Schülervorstellungen zu drei konzeptionellen Komplexitätsstufen von Technik

auch einen „echten“ Computer oder Informatik zu vermuten. Eine Schülerin der 11. und ein Schüler der 13. Klasse, die beide einen Informatik-Leistungskurs besuchten, gaben z.B. folgende Begründungen, aufgrund dessen sie den Mars-Rover aus der Computergruppe bzw. den Industrieroboter aus der Informatikgruppe ausschlossen:

Also mein kleiner Bruder hat das. So einen kleinen Hund. Der ist auch so ein Roboterhund. Der kann wirklich vieles. Der kann lernen und alles und irgendwelche Kunststücke machen, wenn man ihm nur was sagt. Und der hört dann auf den Befehl und so. Den würde ich als Computer einordnen. Der [Mars-Rover] sieht für mich jetzt nicht so aus. (Schülerin, Kl. 11, Info-LK)

Der [Industrieroboter] macht stupide seinen Job. [...] Er programmiert sich nicht selber um bzw. greift auf andere Programmierungen zu, wenn er was anderes als Umgebung hat. (Schüler, Kl. 13, Info-LK)

Beide Begründungen referenzieren intelligentes Verhalten, Vielseitigkeit und Adaptivität – und sprechen sie dem jeweils betrachteten System ab. Entsprechende Merkmale werden damit zu Identifikationskriterien für „echte“ Computer und Informatik. Im Ergebnis stellte sich heraus, dass viele Schüler:innen die potenziellen Fähigkeiten und die Verbreitung informatischer Systeme im Alltag zu unterschätzen schienen, insbesondere bei eingebetteten Systemen. Kurz gesagt: Informatik war in ihren Augen vielseitig, vernetzt, adaptiv und komplex, und Dinge, die nicht vielseitig, vernetzt, adaptiv und komplex sind, waren im Umkehrschluss keine Informatik.

In einer anschließenden Unterrichtsstudie zu eingebetteten Systemen mit drei Wahlpflichtkursen der 10. Klasse zeigten sich zunächst dieselben Denkmuster (Rücker et al., 2020). Eine Untersuchung der Lernprozesse ergab zudem, dass mehrere Schüler:innen sich offenbar schwer damit taten, zu akzeptieren, dass Informationstechnik sowohl niederschwellig (klein, preiswert, sparsam) als auch funktional mächtig (vielseitig, adaptiv, vernetzt, s.o.) sein kann. Kleine und billige „Platinen“ sind zwar weit verbreitet, die Funktionalität ihrer „Programme“ dafür aber stark limitiert. Mächtigere „Systeme“ mit komplexeren Fähigkeiten erfordern „echte Computer“ und anspruchsvolle „Software“, deren Einsatz jedoch nur dort gerechtfertigt ist, wo die kleinen „Platinenprogramme“ nicht mehr ausreichen.

Wenngleich diese Logik nicht gänzlich abwegig erscheint, so widerspricht sie doch dem grundlegenden fachdidaktischen Anliegen, allgemeine Konzepte der Informatik zu vermitteln, die gleichermaßen auf alle Arten von Informatiksystem anwendbar sein sollen. Die Vorstellung, dass bspw. eine „Kleinanwendung“ in der Waschmaschine keine Informatik erfordere (Zitat Schüler, 9. Kl., Info-WP), steht diesem Ziel offensichtlich entgegen.

Wie lässt sich darauf nun im Unterricht reagieren? Auch dazu konnten wir in den beiden Studien Ansätze identifizieren. Einer besteht darin, die Vielseitigkeit und Mächtigkeit auch kleiner Einplatinenrechner im Unterricht explizit erfahrbar zu machen. In der Unterrichtsstudie haben die Schüler:innen z.B. mithilfe eines Raspberry Pi Zero W eigene prototypische eingebettete Systeme implementiert und anschließend über das WLAN angesteuert. Die Netzwerkfähigkeit der kleinen Rechner hatte einige sichtlich überrascht, ebenso wie die Fähigkeit des Raspberry Pi autark vom Desktop-PC zu arbeiten. Die folgende Aussage eines Schülers am Ende der Reihe bringt den gewünschten Lerneffekt auf den Punkt:

Ich fand sehr überraschend, dass so kleine Computer sozusagen genauso viele Funktionen haben [...] wie so ein großer Computer.

Ein weiterer Ansatz ist Sprache. Wie Tabelle 1 und die obigen Formulierungen bereits erkennen lassen, können Begriffe wie „Platine“, „Chip“ oder „Programm“ für Schüler:innen eine andere – weniger mächtige – Konnotation haben als „System“, „Computer“ oder „Software.“ Sprache und mentale Kategorien sind eng miteinander verbunden (Johansen et al., 2015), so-

dass die konkrete Wortwahl einen Einfluss auf mentale Kategorien haben kann. Was genau ist eigentlich der Unterschied zwischen „Programm“ und „Software“? Lässt sich dieser sinnvoll kommunizieren oder sind die beiden Begriffe synonym? Der Unterschied zwischen „Computer“, „Platine“ und „Mikrokontroller“ ist fachlich relativ klar. Didaktisch sollte aber bedacht werden, dass begriffliche Abgrenzungen immer auch als Grenzen zwischen Kategorien verstanden werden können. Und Grenzen zwischen Kategorien sind immer auch potenzielle Grenzen für Lerntransfer. Unsön wäre sicherlich, wenn sich Ihre Schüler:innen am Ende des Unterrichts unsicher sind, welche der Dinge, die sie bereits über „Computer“ gelernt haben, nun eigentlich auch auf diese „Mikrokontroller“ zutreffen.

Bilden Sie sich hierzu zumindest eine bewusste Meinung und vielleicht wollen Sie in Zukunft dann im Unterricht konsequent davon sprechen, dass bspw. der Arduino ein Computer ist und mit einer Software bespielt wird. Und wenn das bei Ihren Schüler:innen Fragen aufwirft, dann hatte es womöglich genau den gewünschten Effekt.

Fazit

Nicht für die Schule, sondern für das Leben soll gelernt werden. Um diesem Anspruch gerecht zu werden, müssen Lernende in die Lage versetzt werden, ihr schulisches Wissen und Können auch außerhalb und nach der Schule anzuwenden. Diese Herausforderung ist kaum zu unterschätzen. Für die Informatik müssen sie dazu u.a. künftige und neue Systemkategorien erschließen und Manifestationen bekannter informatischer Konzepte im Alltag identifizieren können. Auf Basis kognitionspsychologischer Grundlagen sowie drei fachdidaktischer Studien wurden im vorliegenden Artikel Ansätze vorgestellt, dies im Unterricht zu fördern.

Wählen Sie bewusst vielfältige Unterrichtsbeispiele aus möglichst verschiedenen Anwendungskontexten. Die in Abb. 2 gezeigte Taxonomie kann hier als Orientierungshilfe dienen. Scheuen Sie sich dabei auch nicht, die unmittelbare Erfahrungswelt der Schüler:innen zu verlassen. Einerseits ist ein direkter Lebensweltbezug natürlich ein richtiges und wichtiges Prinzip der Unterrichtsgestaltung. Andererseits ist auch klar, dass Unterricht dort nicht stehen bleiben darf. Das Ziel, den Lernenden ein möglichst breites und vielfältiges Bild der Informatik aufzuzeigen und ihren Erfahrungshorizont in dieser Hinsicht zu erweitern, erfordert, dass wir

genau diesen Horizont überschreiten. Wenn wir im Unterricht vor allem über die Beispiele und Kontexte reden, bei denen den Kindern ohnehin klar ist, dass dort irgendwie Informatik drinsteckt (Online-Medien, Smartphones, Videospiele, Roboter, ...), dann wird an dieser Front auch kein nennenswerter Lernzuwachs erfolgen. Wenn Sie es aber schaffen, dass jemand in ihrem Unterricht murmelt: „Ach, da steckt das auch drin? Hätte ich nicht gedacht.“ Dann haben Sie in diesem Moment eine Kategorien-grenze verschoben und Raum für potenziellen Lerntransfer geschaffen. Achten Sie daher auf kontextuelle Vielfalt im Informatikunterricht!

Versuchen Sie darüber hinaus, den Schüler:innen die funktionale Mächtigkeit, die Autonomie, Vielseitigkeit und Vernetzung, gerade auch kleiner und billiger Einplatinenrechner aufzuzeigen, indem Sie diese z.B. mit einem Netzwerk verbinden oder auch bewusst vom „großen“ Rechner trennen. Auch Sprache kann hier wichtige Signale setzen. Entscheiden Sie bewusst, ob und wann Sie Begriffe wie „Programm“ und „Software“ bzw. „Platine“, „Mikrocontroller“ und „Computer“ unterscheiden und verwenden wollen. Diskutieren Sie deren Bedeutung doch einmal mit Ihren Schüler:innen oder lassen Sie sie Bildkarten in „informatisch“ und „nicht informatisch“ sortieren. Vielleicht werden Sie überrascht sein, was der einer oder die andere darunter eigentlich versteht.

Die in den Studien verwendeten Bilddateien liegen dem Artikel als Download bei (vgl. Artikel auf [ibis-Webseite](#)).

Quellen

- Alfieri, L., Nokes-Malach, T. J., & Schunn, C. D. (2013). Learning Through Case Comparisons: A Meta-Analytic Review. *Educational Psychologist*, 48(2), 87–113. <https://doi.org/10.1080/00461520.2013.775712>
- Belenky, D. M., & Schalk, L. (2014). The Effects of Idealized and Grounded Materials on Learning, Transfer, and Interest: An Organizing Framework for Categorizing External Knowledge Representations. *Educational Psychology Review*, 26(1), 27–50. <https://doi.org/10.1007/s10648-014-9251-9>
- Bransford, J. D., & Schwartz, D. L. (1999). Rethinking Transfer: A Simple Proposal with Multiple Implications. *Review of Research in Education*, 24(1999), 61–100.
- Butler, A. C., Black-Maier, A. C., Raley, N. D., & Marsh, E. J. (2017). Retrieving and applying knowledge to different examples promotes transfer of learning. *Journal of Experimental Psychology: Applied*, 23(4), 433–446. <https://doi.org/10.1037/xap0000142>

Diethelm, I., & Dörge, C. (2011). Zur Diskussion von Kontexten und Phänomenen in der Informatikdidaktik. In M. Thomas (Ed.), *Informatik in Bildung und Beruf—14. GI-Fachtagung "Informatik und Schule"*—INFOS 2011.

diSessa, A. A. (2018). A Friendly Introduction to "Knowledge in Pieces": Modeling Types of Knowledge and Their Roles in Learning. In G. Kaiser, H. Forgasz, M. Graven, A. Kuzniak, E. Simmt, & B. Xu (Eds.), *Invited Lectures from the 13th International Congress on Mathematical Education* (pp. 65–84). Springer International Publishing. https://doi.org/10.1007/978-3-319-72170-5_5

Feeney, A., & Heit, E. (2011). Properties of the diversity effect in category-based inductive reasoning. *Thinking & Reasoning*, 17(2), 156–181. <https://doi.org/10.1080/13546783.2011.566703>

Goldstone, R. L., Kersten, A., & Carvalho, P. F. (2018). Categorization and Concepts. In *Stevens' Handbook of Experimental Psychology and Cognitive Neuroscience* (pp. 275–317). John Wiley & Sons, Inc. <https://doi.org/10.1002/9781119170174.epcn308>

Guzdial, M. (2010). Does contextualized computing education help? *ACM Inroads*, 1(4), 4. <https://doi.org/10.1145/1869746.1869747>

Hahn, U., Bailey, T. M., & Elvin, L. B. C. (2005). Effects of category diversity on learning, memory, and generalization. *Memory & Cognition*, 33(2), 289–302. <https://doi.org/10.3758/BF03195318>

Hammer, D., Elby, A., Scherr, R. E., & Redish, E. F. (2005). Resources, framing, and transfer. In J. P. Mestre (Ed.), *Transfer of Learning from a Modern Multidisciplinary Perspective* (pp. 89–120). Information Age Publishing.

Humbert, L., & Puhlmann, H. (2004). Essential Ingredients of Literacy in Informatics. In J. Magenheimer & S. Schubert (Eds.), *Informatics and Student Assessment: Concepts of Empirical Research and Standardisation of Measurement in the Area of Didactics of Informatics* (pp. 65–76). Gesellschaft für Informatik.

Johansen, M. K., Savage, J., Fouquet, N., & Shanks, D. R. (2015). Salience Not Status: How Category Labels Influence Feature Inference. *Cognitive Science*, 39(7), 1594–1621. <https://doi.org/10.1111/cogs.12206>

Knobelsdorf, M., & Tenenbergh, J. (2013). The Context-Based Approach InIK in Light of Situated and Constructive Learning Theories. In I. Diethelm & R. T. Mittermeir (Eds.), *Informatics in Schools. Sustainable Informatics Education for Pupils of all Ages. ISSEP 2013. Lecture Notes in Computer Science* (Vol. 7780, pp. 103–114). Springer. https://doi.org/10.1007/978-3-642-36617-8_9

Lobato, J. (2012). The Actor-Oriented Transfer Perspective and Its Contributions to Educational Research and Practice. *Educational Psychologist*, 47(3), 232–247. <https://doi.org/10.1080/00461520.2012.693353>

Osherson, D. N., Smith, E. E., Wilkie, O., López, A., & Shafir, E. (1990). Category-based induction. *Psychological Review*, 97(2), 185–200. <https://doi.org/10.1037/0033-295X.97.2.185>

Rücker, M. T., & Pinkwart, N. (2018). The things that belong: A grounded theory study of student categorizations of complex technical artifacts. *International Journal of Technology and Design Education*, 28(3). <https://doi.org/10.1007/s10798-017-9419-y>

Rücker, M. T., & Pinkwart, N. (2019). "How Else Should It Work?" A Grounded Theory of Pre-College Students' Understanding of Computing Devices. *ACM Transactions on Computing Education*, 19(1), 1–23. <https://doi.org/10.1145/3226592>

Rücker, M. T., van Joolingen, W. R., & Pinkwart, N. (2020). Small but Powerful: A Learning Study to Address Secondary Students' Conceptions of Everyday Computing Technology. *ACM Transactions on Computing Education*, 20(2), 1–27. <https://doi.org/10.1145/3377880>

Schwill, A. (1997). Computer science education based on fundamental ideas. In D. Passey & B. Samways (Eds.), *Information technology* (pp. 285–291). Springer. https://doi.org/10.1007/978-0-387-35081-3_36

Wagner, J. F. (2006). Transfer in Pieces. *Cognition and Instruction*, 24(1), 1–71. https://doi.org/10.1207/s1532690xci2401_1

Waldmann, M. R. (2008). Kategorisierung und Wissenserwerb. In J. Müsseler (Ed.), *Allgemeine Psychologie* (pp. 377–427). Spektrum.

Lizenz



Dieser Artikel steht unter der Lizenz CC BY 4.0 zur Verfügung.

Kontakt

Prof. Dr. Michael T. Rücker
Juniorprofessur für Didaktik der Informatik
Friedrich-Schiller-Universität Jena
E-Mail: michael.ruecker@uni-jena.de

Das Informatikcurriculum der Hector Kinderakademien

Tsarava, K.* , Kunz, K.* und Trautwein, U.

DOI: 10.18420/ibis-02-02-06

Zusammenfassung

In den Bildungsplänen für Grundschulen der Bundesländer bleibt Informatik und informatisches Denken immer noch außen vor. Dies wird jedoch teilweise durch außerschulische Angebote kompensiert. In Baden-Württemberg und Hessen erhalten beispielsweise Grundschulkin- der die Möglichkeit, an Kursen der Hector Kinderakademien teilzunehmen. In diesem Artikel möchten wir daher die Kurse des Hector Informatikcurriculums Lehrkräften und Fachleuten im Bereich der Informatikbildung vorstellen. Das Informatikcurriculum der Hector Kinderakademien besteht aus drei Kursen und richtet sich an begabte und hochbegabte Kinder der ersten bis vierten Klassenstufe, die an Informatik interessiert sind. Bei der Entwicklung und Etablierung der Kurse wird ein evidenzbasierter Ansatz verfolgt, das heißt, dass die Kurse von der wissenschaftlichen Begleitung der Hector Kinderakademien empirisch hinsichtlich ihrer Förderwirksamkeit untersucht werden. Erste Studien belegen die Förderwirkung der Kurse.

Einleitung

Obwohl heute oft schon die Kleinsten ein eigenes Tablet haben oder mit dem Smartphone spielen dürfen, beginnt der formale Unterricht in Informatik und die Förderung des informatischen Denkens in Deutschland relativ spät, wenn man bedenkt, dass informatisches Denken (engl. *computational thinking*) als Teil der „21st century skills“ betrachtet wird. Tatsächlich hat informatisches Denken mittlerweile weltweit Einzug in die Bildungspläne gehalten, vermehrt auch in Grundschulen. In Europa waren Estland und Großbritannien 2012 Vorreiter, und seitdem haben 15 EU-Länder informatisches Denken in ihren Bildungsplänen für Grundschulen verankert (Bocconi et al., 2022). Obwohl auch die Gesellschaft für Informatik, die größte deutschsprachige Organisation für Informatik, bereits 2019 Empfehlungen für Informatikunterricht und die Förderung von informatischem Denken an Grundschulen veröffentlicht hat (Best et al., 2019), haben in Deutschland immer noch längst nicht alle Bundesländer Informatik

oder informatisches Denken in ihren Bildungsplänen für den Primarbereich aufgenommen. Dies wird teilweise mit fehlenden Lehrkräften begründet, teilweise aber auch mit dem Verweis, dass die Grundschulen zunächst die Grundlagen in Kernbereichen wie Lesen, Schreiben und Mathematik legen müssten und bereits dies bei einem hohen Prozentsatz von Schülerinnen und Schülern misslinge (vgl. Stanat et al., 2022). Der Präsident des Deutschen Lehrerverbandes, Heinz-Peter Meidinger, hat als Reaktion auf Ergebnisse der IGLU-Studie 2022 sogar gefordert, Informatik grundsätzlich aus Grundschulen zu verbannen (Epp, 2023).

Allerdings bedeutet ein Verzicht auf die Förderung informatischen Denkens in der Grundschule auch, dass vielen leistungsstarken Kindern, die großes Interesse an entsprechenden Inhalten aufweisen, eine systematische frühe Förderung in diesem Bereich verwehrt bleibt. Es ist deshalb zu überlegen, ob und wie außerunterrichtliche Angebote diese Lücke schließen können. Für ein solches außerunterrichtliches Angebot haben wir als Teil der wissenschaftlichen Begleitung der Hector Kinderakademien mehrere Kurse entwickelt, die informatisches Denken an Grundschulkindern vermitteln.

Im Jahr 2010 wurde die Hector Kinderakademie (HKA; Golle, J., Herbein, E., Hasselhorn, M., & Trautwein, 2017) in Baden-Württemberg ins Leben gerufen. Bis August 2024 sind 69 lokale Standorte der Akademie¹, die in der Regel an einer oder mehreren Grundschulen angesiedelt sind, Teil der HKA. Das Ziel der HKA ist es, Enrichment-Kurse für die 10% begabtesten Grundschulinnen in Baden-Württemberg anzubieten. In Hessen befinden sich die Hector Kinderakademien derzeit im Aufbau; die ersten Hector Kinderakademien werden dort voraussichtlich 2025 mit ihrer Arbeit beginnen. Die Hector Kinderakademie wird von der Hector Stiftung II finanziert und vom Ministerium für Kultus, Jugend und Sport Baden-Württemberg sowie dem Hessischen Kultusministerium unterstützt.

Rund 23.000 Kinder nehmen jedes Jahr am Kursangebot teil. Die HKA bietet Kurse in allen akademischen Bereichen an, hat aber einen klaren Schwerpunkt für Mathematik, Informatik, Naturwissenschaften und Technik, den sogenann-

* Geteilte Erstautorenschaft der beiden erstgenannten Autoren.

¹ Vgl. die Webseite der Hector Kinderakademie: <https://hector-kinderakademie.de>.

ten MINT-Fächern. Die Teilnahme an den HKA-Kursen ist freiwillig und findet außerhalb der regulären Schulzeit statt, beispielsweise im Ganztagsprogramm, nach der Schule oder am Wochenende. Alle Grundschulen in Baden-Württemberg dürfen Kinder für die HKA nominieren. Die nominierten Kinder können dann jedes Semester selbst entscheiden, an welchem bzw. welchen der angebotenen Kurse in einer nahegelegenen Akademie sie teilnehmen möchten.

Wissenschaftlich begleitet wird die Hector Kinderakademie vom Hector-Institut für Empirische Bildungsforschung an der Universität Tübingen und dem Leibniz-Institut für Bildungsforschung und Bildungsinformation (DIPF) in Frankfurt. Die wissenschaftliche Begleitung koordiniert Qualifizierungsveranstaltungen, unterstützt die Öffentlichkeitsarbeit und verantwortet das Zertifikatsstudium „Begabtenförderung und Potenzialentwicklung“ sowie die akademieübergreifenden Onlinekurse auf der Moodle-Lernplattform. Darüber hinaus umfasst die wissenschaftliche Begleitung die Evaluation der Förderwirkung des Angebots sowie die Federführung bei der Entwicklung von sogenannten Hector Core Courses. Diese werden auf der Basis des aktuellen Forschungsstandes entwickelt und ihre Wirkung mithilfe eines ambitionierten Untersuchungsdesigns überprüft.

Die drei Hector Core Courses des Hector-Informatikcurriculums² (siehe Abbildung 3) richten sich an Kinder von Klassenstufe 1 bis Klassenstufe 4. Kinder der Klassenstufen 1 und 2 können den Kurs „Planeten der Informatik“ besuchen und spielerisch und unplugged verschiedene Gebiete der Informatik kennenlernen. In Klasse 3 und 4 lernen die Kinder zu „Verstehen wie Computer denken“ und erarbeiten mithilfe von lebensgroßen Spielen und blockbasiertem Programmieren Konzepte wie Variablen, Schleifen und bedingte Verzweigungen. Dieses Wissen können Kinder der Klassenstufe 4 dann „Kreativ am Computer“ einsetzen und auf die Programmiersprache Python übertragen. Die beiden Kurse, die sich an Kinder der Klasse 3 und 4 richten, werden dabei auch in einer Online-Version angeboten. Das Hector-Informatikcurriculum ist dabei besonders auf die Förderung von informatischem Denken ausgerichtet. Um dieses Ziel zu erreichen, verwenden die Kurse unplugged-Methoden sowie block- und textbasiertes Programmieren und basieren auf den Prinzipien des spielebasierten und verkörperlichten Lernens.

² Vgl. die Webseite der Hector Core Courses des Hector-Informatikcurriculums: <https://uni-tuebingen.de/de/123280>.

Informatisches Denken als didaktischer Ansatz

Unter informatischem Denken werden zum einen die kognitiven Prozesse und Fähigkeiten verstanden, die eine wesentliche Rolle bei der Lösung informatischer Problemstellungen und bei der Entwicklung von Programmierfähigkeiten spielen (Garcia-Peñalvo, 2016). Zum anderen werden damit erlernbare Denkstrategien bezeichnet, die für das systematische Problemlösen in Disziplinen über die Informatik hinaus einsetzbar sind. Informatisches Denken hebt sich dabei von der allgemeinen Problemlösefähigkeit ab, indem algorithmische Lösungen gefunden werden, die strategisch, systematisch, abstrakt, reproduzierbar und berechenbar sind (Barr & Stephenson, 2011).

Wenn Informatisches Denken unterrichtet werden soll, heißt das also nicht, den Kindern primär oder ausschließlich die Syntax einer Programmiersprache zu zeigen oder den Aufbau von Datenbanken zu erklären. Die Kinder sollen vielmehr lernen, komplexe Probleme zu verstehen, präzise zu formulieren, systematisch zu lösen und die entwickelte Lösung daraufhin zu implementieren, zu bewerten und zu verbessern (Wing, 2006). Um Lösungsalgorithmen für Probleme zu finden, können die Kinder dabei auf informatische Konzepte wie Sequenzen, Operatoren, Variablen, Schleifen, bedingte Verzweigungen, Ereignisse und Funktionen zurückgreifen.

Bei informatischem Denken als fachdidaktischem Ansatz stehen also diese informatischen Konzepte selbst im Vordergrund und nicht etwa ihre Umsetzung in unterschiedlichen Programmiersprachen. Damit kann informatisches Denken anfänglich sogar völlig unplugged, also ohne jegliche digitale Unterstützung, eingeübt werden, bevor es später auf praktische (Programmier-)aufgaben und schlussendlich auf fächerübergreifende Kontexte übertragen wird.

Informatisches Denken (un)plugged

Für die frühe Informatikbildung werden weitgehend „unplugged“-Übungen empfohlen (z.B. Prottsman, 2014). Unplugged (= „ausgesteckt“) meint den Verzicht auf Geräte, die Strom benötigen, also insbesondere Computer oder Roboter. Beim unplugged-Ansatz wird auf greifbare Materialien wie Stifte, Spielkarten und Spielfiguren zurückgegriffen, um den Kindern den Einstieg in die Informatik zu erleichtern und sie nicht mit Technologien zu überfordern.

Der unplugged-Ansatz hat dabei den Vorteil, dass nicht zuerst Programmieren gelernt wer-



Abbildung 1: Ein Beispiel für eine "unplugged"-Übung im Spiel "Schildkröten und Krabben".

den muss, bevor tiefere Einblicke in Bereiche der Informatik möglich werden. Er implementiert zudem den didaktischen Ansatz des Konstruktivismus, indem die Kinder die Möglichkeit bekommen sich selbstständig mit kinästhetischen Übungen zu beschäftigen (Tsarava et al., 2017, 2018). Dadurch kann zum einen bereits sehr früh mit der Förderung von informatischem Denken begonnen werden, zum anderen kann eine große Bandbreite von Kindern, insbesondere auch Mädchen, für die Informatik begeistert werden, die hier sonst unterrepräsentiert sind (Bell & Vahrenhold, 2018).

Allerdings besteht bei ausschließlicher Verwendung von unplugged-Übungen die Gefahr, dass künstliche und kontextlose Lernszenarien dominieren, weshalb die erlernten Fähigkeiten auch auf „plugged-in“-Übungen übertragen werden sollen. Eine Möglichkeit, Kinder schon

früh an informatische Geräte heranzuführen, sind zum Beispiel Lernroboter wie Beebots (Seckel et al., 2023) oder Ozobots (Körper et al., 2021), die mit einfacher Programmierung von den Kindern gesteuert werden können.

Blockbasiertes, hybrides und textbasiertes Programmieren

Der Einstieg ins eigentliche Programmieren geschieht mittlerweile fast ausschließlich in blockbasierten Programmierungsumgebungen wie Scratch oder Code.org, bzw. mit den spezifischen Programmierungsumgebungen der verwendeten Lernroboter. Beim blockbasierten Programmieren werden Programmierblöcke wie Puzzleteile in- und aneinandergeschoben. Dadurch wird Programmieren möglich, ohne dass den Kindern die Tastatur vertraut sein muss oder sie die genaue Syntax der Programmiersprache kennen müssen. Die Kinder können zudem aus einem Angebot an vorgefertigten Blöcken geeignete Anweisungen auswählen und mithilfe von „Drag- und Drop“ an die gewünschten Stellen ziehen. Der Vorteil dabei ist, dass die Kinder Anweisungen nicht im Vorfeld erlernen und sich merken müssen. Als ein Nachteil von blockbasiertem Programmieren wird allerdings gesehen, dass es trotz seiner Möglichkeiten für sehr komplexe Programme von den Kindern oft nicht als „richtiges Programmieren“ angesehen wird (Weintrop & Wilensky, 2017).

Zudem hat sich gezeigt, dass blockbasiertes Programmieren einen späteren Übergang zu

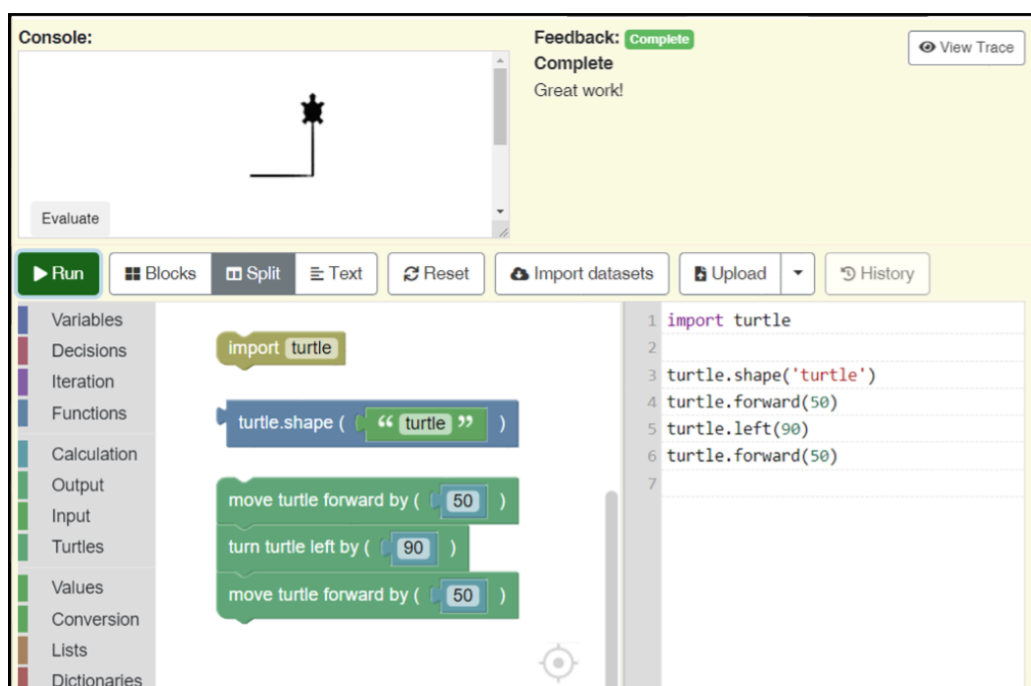


Abbildung 2: Die dual hybride Programmierungsumgebung BlockPy mit der Möglichkeit zum parallelen block- und textbasierten Programmieren.

textbasiertem Programmieren nicht immer erleichtert. Untersuchungen geben ein gemischtes Bild, ob Kinder die erlernten Konzepte einfach übertragen können. Der Übergang kann außerdem dazu führen, dass die Motivation und positive Einstellung der Kinder fürs Programmieren sinken, wenn zusätzliche Schwierigkeiten wie Syntax, Programmiersprachen in Englisch oder auch das Tastaturschreiben dazu kommen (Alrubaye et al., 2020).

Eine Möglichkeit, diesen Übergang weniger abrupt zu gestalten, sind sogenannte hybride Programmierumgebungen (z.B. Weintrop & Wilensky, 2017). Dabei kann blockbasiertes Programmieren zum einen in einer textbasierten Umgebung integriert werden, indem die Programmbearbeitung blockbasiert stattfindet und der bearbeitete Abschnitt anschließend automatisch in Text übersetzt wird. Zum anderen können blockbasiertes Programmieren und textbasiertes Programmieren dual nebeneinander stehen und Veränderungen in einer Modalität werden automatisch auf die andere übertragen.

Spielebasiertes und verkörperlichtes Lernen

Wenn Inhalte und Kompetenzen anhand von Spielen oder spielerischen Aufgaben vermittelt werden, wird dies als spielebasiertes Lernen bezeichnet. In der Informatik gibt es dabei erstens die Möglichkeit Aufgaben zu gamifizieren, indem mithilfe von gelösten Aufgaben Punkte oder Levels erreicht werden können. Dieser Ansatz wird zum Beispiel mit Code.org verfolgt. Zweitens können informatische Aufgaben direkt Teil des Spiels sein und müssen als Teil der Geschichte des Spiels gelöst werden. Drittens

können Kinder mit ihren Programmierkenntnissen auch schon früh eigene Spiele entwickeln, was zum Beispiel bei Scratch ermöglicht wird.

Verkörperlichtes Lernen (auf Englisch: embodied learning) basiert auf der Theorie, dass viele Aspekte des Denkens stark vom physischen Körper oder von physischen Gegenständen abhängig sind (Barsalou, 2008). Der unplugged-Ansatz, aber auch die Verwendung von Robotern basieren auf diesen Grundlagen.

Die Hector-Informatikkurse und Evidenz für ihre Wirksamkeit

Um Kinder der Hector Kinderakademien in ihrem informatischen Denken zu fördern, haben wir drei Kurse entwickelt, die die Methoden des spielebasierten und verkörperlichten Lernen einsetzen: „Planeten der Informatik“, „Verstehen wie Computer denken“ und „Kreativ am Computer“. Zusätzlich gibt es einen Kurserweiterung, die sich „Planet des Internets“ nennt.

Das Hector-Informatikcurriculum, basierend auf den GI-Kompetenzen für Informatische Bildung, zielt darauf ab, Schüler*innen dabei zu unterstützen, Algorithmen in ihrer Lebenswelt anzuwenden, algorithmische Grundbausteine zu nutzen und Algorithmen in Alltagssprache zu beschreiben. Darüber hinaus sollen sie Algorithmen mit den algorithmischen Grundbausteinen Anweisung, Sequenz, Wiederholung und Verzweigung entwerfen, realisieren und testen, Algorithmen in verschiedenen formalen Darstellungsformen darstellen, Algorithmen unter Verwendung der Fachsprache vergleichen und ein Informatiksystem programmieren.

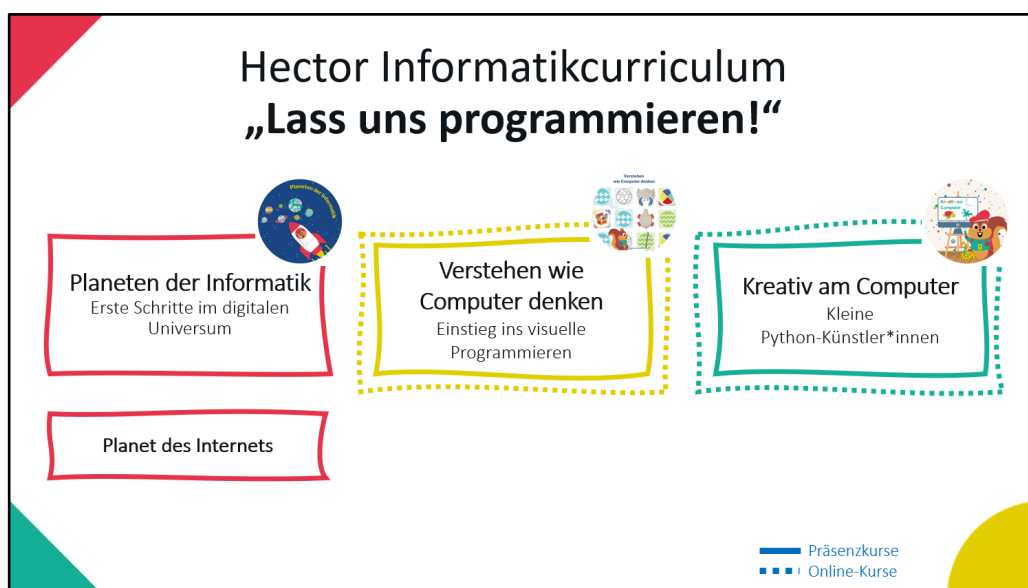


Abbildung 3: Die Kurse des Hector-Informatikcurriculums.

Das Curriculum zielt außerdem darauf ab, die Schüler*innen dabei zu fördern, Automaten in ihrer Lebenswelt als selbstständig arbeitende Maschinen zu beschreiben, Zustände von Automaten zu benennen und zu erläutern, dass ein Automat regelgesteuert seine Zustände verändert. Damit unterstützt es die Schülerinnen und Schüler auch dabei, zu verstehen, dass ihre Lebenswelt von Informatik durchdrungen ist.

Planeten der Informatik

Der Kurs „Planeten der Informatik“ lässt Kinder der ersten und zweiten Klasse in die Informatik-Galaxie eintauchen, wo sie mit den Außerirdischen, den Heckis, von Planeten zu Planeten reisen. Die Kinder lernen auf dem ersten Planeten, dem „Heimatplaneten“ der Heckis, was Befehle sind, wie sie zu Sequenzen zusammengefügt werden, wie diese Sequenzen zum Beispiel in bunten Perlenarmbändern vorkommen und dass ein Algorithmus so etwas wie Anleitung zum Kressepflanzen sein kann. Auf dem „irren Planeten“ müssen die Kinder dann Befehle und Sequenzen anwenden, um sich gegenseitig aus Irrgärten zu befreien. Auf dem „Planet Krypto“ lernen die Kinder und die Heckis den Außerirdischen „Löffel“ kennen, welcher nur in Löffelsprache spricht und den Kindern andere Außerirdische vorstellt, die alle in merkwürdigen Geheimsprachen wie Pixelsprache oder Blinzelsprache sprechen.

Auf dem vierten Planeten, dem „Automaten-Planet“, erfahren die Kinder wie Automaten mit



Abbildung 4: Die gebastelten Heckis auf dem Automatenplaneten.

Zuständen und Zustandsübergängen dargestellt werden können und haben selbst die Möglichkeit Automaten, die sie in ihrem Alltag benutzen so darzustellen. Befehle, Sequenzen und Algorithmen finden auch ihre Anwendung auch auf dem „Planet Rhythmo“, auf dem die Kinder eigene Musik komponieren können, die dann mit dem eigenen Körper als Schlagzeug aufgeführt werden kann. Auf den Planeten sechs („tanzender Planet“) und sieben („verzweigter Planet“) lernen die Kinder kleine Roboter, die Ozobots, kennen und wie man diese

mit gemalten Linien und verschiedenen Farben zum Tanzen bringt und über verzweigte Wege führen kann. Anschließend kehren die Kinder mit den Heckis zurück und erzählen den zurückgebliebenen Heckis von ihrer Reise, indem sie alle Planeten nochmal Revue passieren lassen.

Der Kurs hält sich bei der Wahl der Inhalte an die von der Gesellschaft für Informatik vorgegeben Themen und die zum Ende der zweiten Klasse zu erreichenden Kompetenzen wie „Informationen und Daten“ (Planet Krypto), „Algorithmen“ (Heimatplanet, irrer Planet, Planet Rhythmo, verzweigter Planet) und „Sprachen und Automaten“ (Automaten Planet), „Informatiksysteme“ (Automaten-Planet, tanzender bzw. verzweigter Planet). Das Themengebiet „Informatik, Mensch und Gesellschaft“ wird über alle Kurssitzungen, also Planeten, betont, indem die Kinder lernen die theoretischen Konzepte mit ihrer Umgebung und ihrem Alltag in Verbindung zu bringen, wie zum Beispiel Sequenzen in der Musik oder Automaten auf dem Schulweg. Der Kurs findet dabei bis auf die Verwendung der Ozobots unplugged statt und selbst die Ozobots werden nur zur Unterstützung des verkörperlichten Ansatzes verwendet.

Planet des Internets

Anschließend an den Kurs „Planeten der Informatik“ können Kinder der dritten und vierten Klasse den Kurs „Planet des Internets“ besuchen. In zwei Doppelstunden lernen die Kinder hier, wie das Internet als Netzwerk von Netzwerken aufgebaut ist, wie Webseiten aufgerufen werden und Nachrichten über das Internet verschickt werden können. Der Kurs verwendet dafür Spiele, die die Funktionsweise veranschaulichen. Als Einstieg wird eine Variante von „Stille Post“ gespielt. Alle Kinder bekommen eine Dobble-Karte mit sechs verschiedenen Symbolen und nur Kinder mit einem bestimmten Symbol werden als Verbindungspunkte des Senders und Empfängers der „Stillen-Post“-Nachricht eingesetzt. Das darauffolgende Netzwerk-Spiel lässt die Kinder nur mit Hilfe leitender Fragen der Kursleitung ein Netzwerk bilden, in welchem Nachrichten übermittelt werden können. Die Kinder entwickeln hier selbstständig Ideen die Adressen oder Routern entsprechen. Schließlich verkörperlichen die Kinder im Internet-Verbindungs-Spiel einzelne Elemente des Internets wie den Anbieter oder den Router, nachdem der Ablauf des Spiels mithilfe von Kartonfiguren gezeigt wurde. Die Materialien dieses Kurses sind als Open Educational Resource (OER) unter folgendem Link verfügbar http://hdl.handle.net/10900.3/OER_IMOLFSPPE.

Verstehen wie Computer denken

Im Gegensatz zu den Planeten-Kursen, die sich inhaltlich auch auf verschiedene Bereiche der

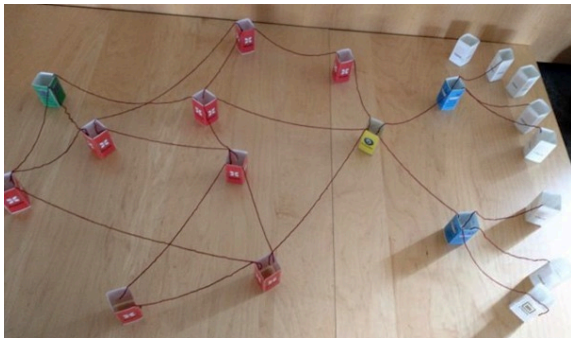


Abbildung 5: Das Internet-Vernetzungs-Spiel.

Informatik wie Automaten, Kryptographie oder Netzwerke beziehen, fokussiert sich der Kurs „Verstehen wie Computer denken“ ganz besonders auf die Förderung von informatischem Denken.

Mit dem speziell für den Kurs entwickelten lebensgroßen Spiel „Schildkröten und Krabben“ (Tsarava, Moeller, et al., 2019) werden die Kinder spielerisch an die grundlegenden informatischen Konzepte (Sequenzen, Schleifen, bedingte Verzweigung und Ereignisse) herangeführt. „Schildkröten und Krabben“ enthält eine Serie von drei Lernspielen: „Die Schatzsuche“, „Die Muster“ und „Das Wettrennen“, welche entweder als Karten- oder Brettspiel konzipiert sind. Die Spiele wurden bewusst nicht als digitale Spiele entwickelt, um die Kinder erleben zu lassen, dass informatische Konzepte nicht nur in digitalen Umgebungen anwendbar sind. Sie zielen darauf ab kognitive Prozesse des informatischen Denkens wie algorithmisches Denken, Abstraktion und Mustererkennung zu trainieren. Die Brettspiele sind als OER unter folgendem Link verfügbar http://hdl.handle.net/10900.3/OER_MDCKSMXP.

Der Kurs geht schrittweise dazu über die im Spiel erlernten Konzepte und Prozesse in der

Programmierungsumgebung Scratch³ und der „open-hardware“-Plattform Arduino einzusetzen. Im letzten Modul des Kurses verwenden die Kinder Open Roberta Lab⁴ eine interaktive Robotersimulation, um mit einer einfach erlernten blockbasierten Programmiersprache eigenständig Probleme zu lösen. Das Kursmanual für diesen Kurs ist zu finden bei Leifheit (2020, S. 118-367).

Kreativ am Computer

Kreativ am Computer ist der letzte Kurs des Informatikcurriculums und richtet sich an Kinder der vierten Klasse, die schon etwas Programmiererfahrung haben und den Übergang zu textbasiertem Programmieren machen wollen. Dafür wird die dual hybride Programmierungsumgebung BlockPy (für weitere Informationen siehe Bart et al., 2017) verwendet, die den Kindern jederzeit erlaubt, sowohl block- wie auch textbasiert zu programmieren. BlockPy basiert auf der Programmiersprache Python und erlaubt die Verwendung des Python Moduls Turtle Graphics⁵, welches die Steuerung einer simulierten Schildkröte ermöglicht. Die Schildkröte hält dabei einen Stift, der durch die Bewegung über den Bildschirm bunte Linien zeichnen kann.

Der Kurs folgt vom didaktischen Ansatz dem EIS-Prinzip (Bruner, 2009), bei dem zuerst mithilfe von Metaphern ein bestimmtes Konzept enaktiv (unplugged) eingeführt wird, im nächsten Schritt die Kinder das Konzept mithilfe der Programmblöcke ikonisch verwenden und schlussendlich beim textbasierten Programmieren symbolisch einsetzen. Die Metaphern sind dabei zum Beispiel eine Box für das Konzept der Variable. Die Box kann mit einem Namen beschriftet werden (Variablenname) und dieser Name steht als Platzhalter für den Inhalt (Variablenwert). Eine Variablenzuweisung ent-

³ Scratch <https://scratch.mit.edu/>.

⁴ Open Roberta Lab <https://lab.open-roberta.org/>.

⁵ Für die Dokumentation des Moduls siehe <https://docs.python.org/3/library/turtle.html>.

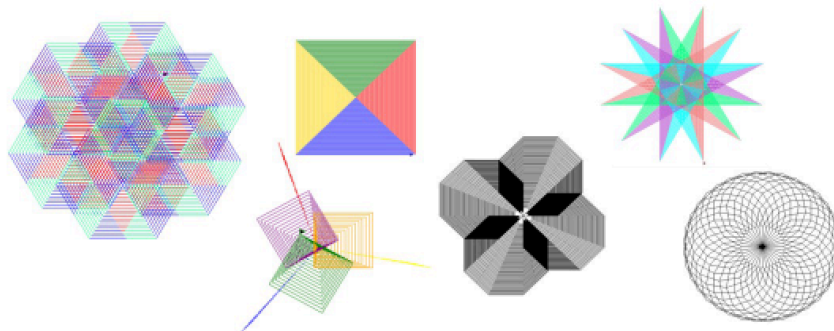


Abbildung 6: Beispiele für die von den Kindern programmierten Bilder im Kurs „Kreativ am Computer“.

spricht also dem Hinzufügen von einem Zettel mit einer darauf gedruckten Zahl in die Box. Zu jedem so eingeführten Konzept folgen zwei Aufgaben für die Kinder. Zuerst sollen die Kinder eine Aufgabe lösen, die einen bestimmten Lösungsweg und einen geplanten Einsatz des neuen Konzepts verlangt, und dann folgt eine Aufgabe, bei der die Kinder das Konzept kreativ einsetzen und eigene Bilder erstellen können. Zum Abschluss des Kurses programmieren die Kinder ein Computerspiel. Die erste Online-Version des Kurses ist als OER unter folgendem Link verfügbar http://hdl.handle.net/10900.3/OER_UEVLLJBZ.

Die Wirksamkeit der Kurse

Die evidenzorientierte Strategie (siehe Abbildung 7) der Hector Kinderakademien sieht vor, alle Kurse auf ihre Wirksamkeit zu überprüfen. Nachdem spezifische Bedürfnisse und Lernziele identifiziert wurden, werden auf der Basis aktueller wissenschaftlicher Erkenntnisse das Kurskonzept und die didaktische Umsetzung entwickelt. Anschließend durchlaufen alle Kurse mehrere Stufen der empirischen Überprüfung: Zuerst wird der Kurs im kleinen Rahmen (ca. 30 Kinder) von den Kursentwicklerinnen auf die Durchführbarkeit untersucht (Pilotierung). Dabei werden Materialien und die geplanten Zeiten für einzelne Übungen angepasst und eine erste Einschätzung getroffen, ob der Kurs seine Ziele erfüllt. Beim Informatikcurriculum beinhalten die Ziele natürlich zuvorderst die Förderung informatischen Denkens. Es werden aber auch die Auswirkungen des Kurses auf Motivation, Selbstkonzept und Kreativität der Kinder untersucht.

In nächsten Schritt („Wirksamkeitsstudie“; rund 100 Kinder) wird die Wirksamkeit des Kurses auf die Erreichung der Ziele untersucht, wenn gezielt ausgewählte Kursleitungen an den Hector Kinderakademien den Kurs unterrichten und sich dabei strikt an den Vorgaben des Kurshandbuchs orientieren. In einem weiteren

Schritt („Effektivitätsstudie“; rund 200 Kinder) wird untersucht, ob der Kurs seine Wirksamkeit beibehält, wenn der Kurs von regulären Kursleitungen der jeweiligen Akademien unterrichtet wird, die zuvor eine Fortbildung erhalten haben. In einem letzten Schritt kann („Scaling-up“) geprüft werden, inwieweit die Wirkung des Kurses auch nach vollständiger Implementation im Programm stabil ist.

Der Kurs „Verstehen wie Computer denken“ hat die drei Phasen von Pilotstudie bis Effektivitätsstudie durchlaufen. Wir konnten dabei feststellen, dass er einen signifikant positiven Effekt hatte und sich das informatische Denken der Kinder verbessert hatte (Tsarava, 2020; Tsarava, Leifheit, et al., 2019). Auch die Motivation und das Selbstkonzept für das Programmieren der Kinder, die am Kurs teilnahmen, war im Vergleich zu Kindern der Kontrollgruppe signifikant höher (Leifheit, 2020; Leifheit et al., 2019).

Zum jetzigen Zeitpunkt haben die Kurse „Planeten der Informatik“, „Planet des Internets“ und „Kreativ am Computer“ nur die Pilotierung durchlaufen. Die Praxistauglichkeit aller drei Kurse konnte dabei festgestellt werden. Die Pilotstudien signalisiert, dass der Kurs „Planet des Internets“ mittelstarke bis starke Effekte auf die Korrektur von Missverständnissen von Grundschulkindern über das Internet hat (Nutz et al., 2024). Ebenso deuten die Pilotstudien darauf hin, dass der Kurs „Kreativ am Computer“ die informatischen Denkfähigkeiten der Kinder verbessert und potenziell den Übergang von blockbasiertem zu textbasiertem Programmieren erleichtern kann, während die Motivation der Kinder erhalten bleibt (Kunz et al., 2023).

Fazit

Das Hector-Informatikcurriculum der Hector Kinderakademien zielt darauf ab, eine Lücke in der informatischen Bildung von begabten und hochbegabten Kindern zu schließen, die an ihren Grundschulen keinen Informatikunter-

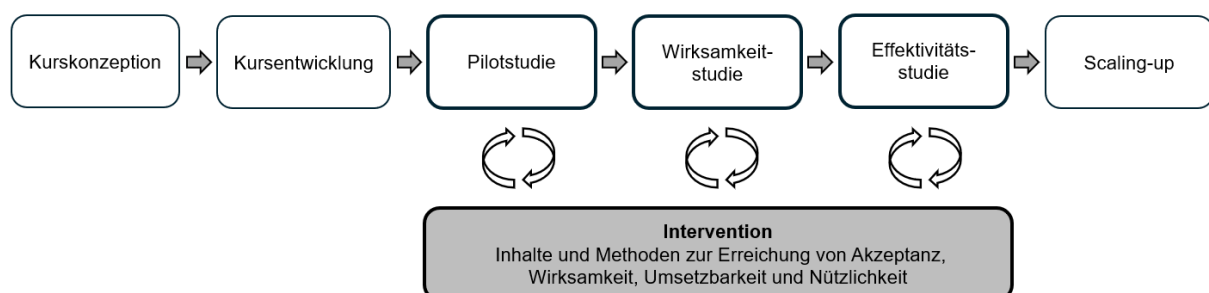


Abbildung 7: Phasen der Kursentwicklung und -bewertung als Interventionen
(Abbildung adaptiert von Trautwein et al., 2023).

richt erhalten. Die von uns durchgeführten empirischen Feldversuche haben gezeigt, dass diese Kinder nicht nur Interesse an informatischen Konzepten haben und diese früh verstehen können, sondern auch, dass informatisches Denken auf angemessene Weise früh gefördert werden kann. Dies ist eine wichtige Erkenntnis, da informatisches Denken als Problemlösungskompetenz über die Bereiche der Informatik hinaus eingesetzt werden kann.

Unsere eigenen Arbeiten zum informatischen Denken von Grundschulkindern sowie viele weitere außerunterrichtliche und außerschulische Angebote können natürlich nicht direkt die Frage beantworten, inwieweit Informatik und informatisches Denken grundsätzlich in die Bildungspläne der Länder für die Grundschule aufgenommen werden sollten. Sie verdeutlichen aber, dass die Kultusministerien sich ernsthaft mit diesem Thema auseinandersetzen sollten.

Soll informatisches Denken in der Grundschule unterrichtet werden, werden dafür ausgebildete Lehrkräfte benötigt, die auf aktuelle, altersgerechte Unterrichtsmaterialien zugreifen können. Ohne dies wird das Vorwissen der Kinder in der Sekundarschule aufgrund der Nutzung außerunterrichtlicher Angebote sehr heterogen sein, was einen noch differenzierteren und adaptiveren Unterricht erfordert, damit alle Schüler*innen entsprechend ihres Lernstands gefördert werden. Aufgrund der Ergebnisse unserer Arbeiten und der Bedeutung von informatischem Denken als „21st-century skill“ plädieren wir für die frühe Implementierung informatischer Inhalte und deren Aufnahme in die Bildungs-Lernpläne für Grundschulen.

Quellen

Alle Webseiten/Links wurden zuletzt geprüft am 08.08.2024.

Alrubaye, H., Ludi, S., & Mkaouer, M. W. (2020). Comparison of block-based and hybrid-based environments in transferring programming skills to text-based environments. *CASCON 2019 Proceedings - Conference of the Centre for Advanced Studies on Collaborative Research - Proceedings of the 29th Annual International Conference on Computer Science and Software Engineering*, April, 100–109.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12. *ACM Inroads*, 2(1), 48–54. <https://doi.org/10.1145/1929887.1929905>

Barsalou, L. W. (2008). Grounded Cognition. *Annual Review of Psychology*, 59(August), 617–645. <https://doi.org/10.1146/annurev.psych.59.103006.093639>

Bart, A. C., Tibau, J., Tilevich, E., Shaffer, C. A., & Kafura, D. (2017). BlockPy: An Open Access Data-Science Environment for Introductory Programmers. *Computer*, 50(5), 18–26. <https://doi.org/10.1109/MC.2017.132>

Bell, T., & Vahrenhold, J. (2018). CS Unplugged—How Is It Used, and Does It Work? (pp. 497–521). https://doi.org/10.1007/978-3-319-98355-4_29

Best, A., Borowski, C., Büttner, K., Freudenberg, R., Fricke, M., Haselmeier, K., Herper, H., Hinz, V., Humbert, L., Müller, D., Schwill, A., & Thomas, M. (2019). Kompetenzen für informatische Bildung im Primarbereich. Gesellschaft für Informatik e.V.

Bocconi, S., Chiocciariello, A., Kampylis, P., Dagienė, V., Wastiau, P., Engelhardt, K., Earp, J., Horvath, M., Jasutė, E., Malagoli, C., Masiulionytė-Dagienė, V., & Stupurienė, G. (2022). State of play and practices from computing education REVIEWING COMPUTATIONAL THINKING IN COMPULSORY EDUCATION. <https://doi.org/https://dx.doi.org/10.2760/126955>

Bruner, J. S. (2009). The process of education. In Bruner, Jerome S. *The process of education*. Harvard university press, 2009.

Epp, E. (2023, May 19). Präsident des Lehrerverbandes fordert: Grundschule soll sich auf drei Fächer konzentrieren. *Stern*. <https://www.stern.de/gesellschaft/grundschule--lehrerverband-praesident-will-zwei-faecher-streichen-33482516.html>

Garcia-Peñalvo, F. J. (2016). What Computational Thinking Is. *Journal of Information Technology Research*, 9(3), v–vi(October).

Golle, J., Herbein, E., Hasselhorn, M., & Trautwein, U. (2017). Talentförderung in der Grundschule durch Enrichment: Das Beispiel der Hector-Kinderakademien. In U. Trautwein & M. Hasselhorn (Ed.), *Begabungen und Talente* (Hogrefe, pp. 191–210).

Körber, N., Bailey, L., Greifenstein, L., Fraser, G., Sabitzer, B., & Rottenhofer, M. (2021). An Experience of Introducing Primary School Children to Programming using Ozobots (Practical Report). *The 16th Workshop in Primary and Secondary Computing Education*, 1–6. <https://doi.org/10.1145/3481312.3481347>

Kunz, K., Moeller, K., Ninaus, M., Trautwein, U., & Tsarava, K. (2023). Making the Transition to Text-Based Programming: The Pilot Evaluation of a Computational Thinking Intervention for Primary School Students. *Proceedings of the 18th WiPSCE Conference on Primary and Secondary Computing Education Research*, 1–6. <https://doi.org/10.1145/3605468.3609770>

Leifheit, L. (2020). The Role of Self-Concept and Motivation Within the "Computational Thinking" Approach to Early Computer Science Education. <http://dx.doi.org/10.15496/publikation-55314>

Leifheit, L., Tsarava, K., Moeller, K., Ostermann, K., Golle, J., Trautwein, U., & Ninaus, M. (2019). Development of a Questionnaire on Self-concept, Motivational Beliefs, and Attitude Towards Programming. Proceedings of the 14th Workshop in Primary and Secondary Computing Education, 1–9. <https://doi.org/10.1145/3361721.3361730>

Nutz, M., Kunz, K., & Tsarava, K. (2024). Development and Empirical Assessment of an Intervention on the Internet's Structure and Functioning for Third and Fourth Graders. 2024 IEEE Global Engineering Education Conference (EDUCON), 01–10. <https://doi.org/10.1109/EDUCON60312.2024.10578725>

Prottzman, K. (2014). Computer science for the elementary classroom. ACM Inroads, 5(4), 60–63. <https://doi.org/10.1145/2684721.2684735>

Seckel, M. J., Salinas, C., Font, V., & Sala-Sebastià, G. (2023). Guidelines to develop computational thinking using the Bee-bot robot from the literature. Education and Information Technologies, 28(12), 16127–16151. <https://doi.org/10.1007/s10639-023-11843-0>

Stanat, P., Schipolowski, S., Schneider, R., Sachse, K. A., Weirich, S., & Henschel Sofie (Eds.). (2022). IQB-Bildungstrend 2021: Kompetenzen in den Fächern Deutsch und Mathematik am Ende der 4. Jahrgangsstufe im dritten Ländervergleich. Waxmann Verlag.

Trautwein, U., Golle, J., Jaggy, A., Hasselhorn, M., & Nagengast, B. (2023). Mutual benefits for research and practice: Randomized controlled trials in the Hector Children's Academy Program. Annals of the New York Academy of Sciences, 1530(1), 96–104. <https://doi.org/10.1111/nyas.15074>

Tsarava, K. (2020). Computational Thinking as a Cognitive Construct: Cognitive Correlates, Assessment & Curriculum Design [University of Tübingen]. <http://dx.doi.org/10.15496/publikation-55320>

Tsarava, K., Leifheit, L., Ninaus, M., Román-González, M., Butz, M. V., Golle, J., Trautwein, U., & Moeller, K. (2019, October 23). Cognitive correlates of computational thinking: Evaluation of a blended unplugged/Plugged-in course. ACM International Conference Proceeding Series. <https://doi.org/10.1145/3361721.3361729>

Tsarava, K., Moeller, K., & Ninaus, M. (2018). Training Computational Thinking through board games: The case of Crabs & Turtles. International Journal of Serious Games, 5(2), 25–44. <https://doi.org/10.17083/ijsg.v5i2.248>

Tsarava, K., Moeller, K., & Ninaus, M. (2019). Board games for training computational thinking. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Vol. 11385 LNCS. https://doi.org/10.1007/978-3-030-11548-7_9

Tsarava, K., Moeller, K., Pinkwart, N., Butz, M. V., Trautwein, U., & Ninaus, M. (2017). Training computational thinking: Game-based unplugged and plugged-in activities in primary school. Proceedings of the 11th European Conference on Games Based Learning, ECGBL 2017, October, 687–695.

Weintrop, D., & Wilensky, U. (2017). Comparing block-based and text-based programming in high school computer science classrooms. ACM Transactions on Computing Education, 18(1). <https://doi.org/10.1145/3089799>

Lizenz



Dieser Artikel steht unter der Lizenz CC BY NC 4.0 zur Verfügung.

Kontakt

Dr. Katerina Tsarava
E-Mail: katerina.tsarava@uni-tuebingen.de

Katrin Kunz
E-Mail: katrin.kunz@uni-tuebingen.de

Prof. Dr. Ulrich Trautwein
E-Mail: ulrich.trautwein@uni-tuebingen.de

Hector-Institut für Empirische Bildungsforschung

Symmetrische Kryptologie und ihre Veranschaulichung

Koch, A.

DOI: 10.18420/ibis-02-02-07

Zusammenfassung

Die Kryptologie, die Wissenschaft des Geheimen, gliedert sich in Kryptographie, die Wissenschaft der Geheimschriften und in Kryptoanalyse, die sich mit der Untersuchung der Sicherheit von Geheimschriften beschäftigt. Die Lehrpläne der Bundesländer enthalten eine große Vielfalt an Themen der Kryptologie. Beispielsweise sehen die Lehrpläne baden-württembergischer Gymnasien monoalphabetische Substitutionsverschlüsselungen und ihre Kryptoanalyse in Klassenstufe 7, polyalphabetische Substitutionsverschlüsselungen und Transpositionsverfahren in Klassenstufe 8 sowie symmetrische und asymmetrische Verfahren in der Sekundarstufe II vor.

Mit Verschlüsselungsverfahren, häufig kurz als Verschlüsselungen oder noch knapper als Verfahren bezeichnet, sind formal injektive Codierungen gemeint, deren Umkehrbarkeit auf Kenntnis eines Geheimnisses beruht. Dieses Geheimnis kann auch das Verfahren selbst sein. Die Umkehrung entspricht dem Entschlüsselungsverfahren. Parameter, die zur Durchführung der Ver- und Entschlüsselung benötigt werden, werden als Schlüssel bezeichnet. Das nach dem Niederländer Auguste Kerckhoffs (1835-1903) benannte Kerckhoffs'sche Prinzip besagt, dass die Sicherheit eines Verfahrens nicht auf Geheimhaltung des Verfahrens selbst beruhen darf. Dies impliziert als notwendige Bedingung, dass sichere Verfahren Schlüssel bedürfen.

In diesem Artikel werden Möglichkeiten für einen anschaulichen Unterricht zur symmetrischen Kryptologie aufgezeigt. Bei symmetrischen Verfahren sind die Schlüssel zur Entschlüsselung und Verschlüsselung gleich.

Transpositionsverschlüsselungen

Transpositionsverschlüsselungen sind Verfahren, bei denen die Positionen der Zeichen des Klartexts vertauscht werden, um den Geheimtext zu bilden. Die Spartaner verwendeten mit der Skytale (griechisch *skytālē*, für „Stab“), einem runden Holzstab, bereits vor über 2500 Jahren ein derartiges Verfahren, um ihre Nachrichten zu verschlüsseln. Dazu wickelten sie ei-

nen Pergamentstreifen um ihre Skytale, beschrieben ihn längs und drehten am Ende angekommen die Skytale weiter. Abbildung 1 zeigt die moderne Variante einer Skytale aus Gardinenstange und zusammengeklebten DIN A4-Blattstreifen.

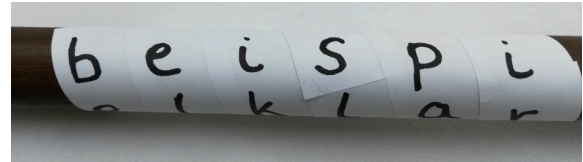


Abbildung 1: Skytale (Andreas Koch / CC BY-SA 4.0)

Der Klartext *beispielklartext* aus Abbildung 1 wird zunächst auf dem Papierstreifen notiert, der dabei sukzessive gedreht wird. Anschließend wird der Streifen abgerollt und die Leerzeichen werden entfernt. So ergibt sich der Geheimtext *BETELEIKXSLTPAIR* (Geheimtexte werden in diesem Artikel zur besseren Abgrenzung vom Klartext groß geschrieben). Durch das Entfernen der Leerzeichen wird die Kryptoanalyse erschwert. Der Schlüssel entspricht der Skytale selbst bzw. der Anzahl der Windungen, also in diesem Beispiel der Zahl 6. Er wird demzufolge durch den Umfang der Skytale festgelegt. Bei fünf Windungen als Schlüssel würde sich der Geheimtext *BIATEERILTS-KEPLX* ergeben.

Als Veranschaulichung, insbesondere zur Vorbereitung auf die Implementierung des Verfahrens, bietet sich ein Matrix-ähnliches Schema wie in Abbildung 2 an.

#s(palten)=schluessel=6						
#z(eilen)=3						
z/s	0	1	2	3	4	5
0	b	e	i	s	p	i
1	e	l	k	l	a	r
2	t	e	x	t		
Durchlaufreihenfolge (von oben links nach unten rechts):						
0+0*6=0, 0+1*6=6, 0+2*6=12						
1+0*6=1, 1+1*6=7, 1+2*6=13						
2+0*6=2, 2+1*6=8, 2+2*6=14						
3+0*6=3, 3+1*6=9, 3+2*6=15						
4+0*6=4, 4+1*6=10, 4+2*6=16						
5+0*6=5, 5+1*6=11, 5+2*6=17						

Abbildung 2: Veranschaulichung der Verschlüsselung (Andreas Koch / CC BY-SA 4.0)

```

01 public static String Skytale(String text, int schluessel)
02 {
03     String geheimtext = "";
04     int z = (text.length() / schluessel) + 1; // Zeilenanzahl
05
06     for (int i=0; i<schluessel; i++) {
07         for (int j=0; j<z; j++) {
08             if (i+j*schluessel<text.length()) { // Platz gefüllt?
09                 geheimtext += text.charAt( (i+j*schluessel) );
10             }
11         }
12     }
13     return geheimtext;
14 }

```

Abbildung 3: Java-Implementierung der Skytale-Verschlüsselung (Andreas Koch / CC BY-SA 4.0)

Der Klartext kann demzufolge zeilenweise in ein zweidimensionales Array geschrieben werden, das anschließend spaltenweise ausgelesen wird, um den Geheimtext zu erhalten. Die tatsächliche Verwendung des Arrays ist in einer Implementierung der Verschlüsselung allerdings nicht notwendig, da die Indizes auch berechnet werden können (siehe die Rechnungen zur Durchlaufreihenfolge in Abbildung 2). Zu beachten ist, dass vorab überprüft werden muss, ob der Eintrag tatsächlich existiert, wie die Beispiele der Indizes 16 und 17 aus Abbildung 2 zeigen.

Die Implementierung der Entschlüsselung erfolgt in ähnlicher Weise. Die Skytale erfüllt das Kerckhoffs'sche Prinzip nicht, da die Anzahl der Schlüssel im Allgemeinen gering ist. Eine obere Schranke für die Schlüssellänge ist die Klartextlänge. Mit überschaubarem Aufwand können per Bruteforce verschiedene Windungszahlen in einem Schema wie in Abbildung 2 durchprobiert werden, bis sich ein sinnvoller Text ergibt.

Substitutionsverschlüsselungen

Bei Substitutionsverschlüsselungen wird jeweils ein Zeichen des Klartexts durch genau ein Zeichen (monoalphabetische Verschlüsselungen) wie bei der Cäsar-Verschlüsselung oder mehrere Zeichen (polyalphabetische Verschlüsselungen) des Geheimtextalphabetes wie bei der Vigenère-Verschlüsselung ersetzt.

Cäsar-Verschlüsselung

„... wenn etwas Geheimen zu überbringen war, schrieb er in Zeichen, das heißt, er ordnete die Buchstaben so, dass kein Wort gelesen werden konnte: Um diese zu lesen, tauscht man den vierten Buchstaben, also D für A aus und ebenso mit den restlichen.“ (Aus „Sueton: De Vita Caesarum: Divus Julius LVI“.)

Die Cäsar-Verschlüsselung ist demzufolge eine monoalphabetische Substitutionsverschlüssel-

ung, die das Kerckhoffs'sche Prinzip nicht erfüllt, da keine Schlüssel zur Ver- und Entschlüsselung benötigt werden.

Mithilfe einer Codetabelle wie in Abbildung 4 kann jede monoalphabetische Substitutionsverschlüsselung mit endlichem Alphabet definiert und dargestellt werden.

Beispielsweise besitzt der Klartext `geheim` den Geheimtext `JHKHLP`. Die Verschiebezahl 3, um die ein Klartextbuchstabe im Alphabet verschoben wird, kann variiert werden, wodurch die Cäsar-Verschlüsselung zu einem Verfahren mit Schlüssel wird. Unter den 26 Schlüsseln befinden sich abzüglich der Verschiebezahl 0 genau 25 sinnvolle Schlüssel. Mithilfe der sogenannten Cäsar-Scheibe aus Abbildung 5 können die 26 Verschlüsselungen veranschaulicht und Ver- und Entschlüsselung von Texten schnell realisiert werden.

Für eine Implementierung des Verschlüsse-

a	b	c	d	e	f	g	h	i	j	k	l	m
D	E	F	G	H	I	J	K	L	M	N	O	P
n	o	p	q	r	s	t	u	v	w	x	y	z
Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Abbildung 4: Codetabelle zum Cäsar-Verfahren (Andreas Koch / CC BY-SA 4.0)

ungsverfahrens kann der Schlüssel entweder als Zahl oder als Buchstabe in Char- oder String-Codierung eingegeben werden. Die Implementierung aus Abbildung 6 zeigt letztere Variante für die Verschlüsselung von Kleinbuchstaben, deren ASCII-Zahlenbereich von 97 zur Codierung des Buchstabens „a“ bis 122 für „z“ reicht. In Zeile 9 muss daher vom ASCII-Wert des Schlüsselbuchstabens der Wert 97 abgezogen werden, um die korrekte Verschiebezahl zu berechnen. In den Zeilen 10 und 11 wird überprüft, ob der ASCII-Bereich der Kleinbuchstaben überschritten wurde. Falls ja, muss der ASCII-Wert korrigiert werden, was die Cäsar-Scheibe durch ihren zyklischen Aufbau „automatisch“ macht.

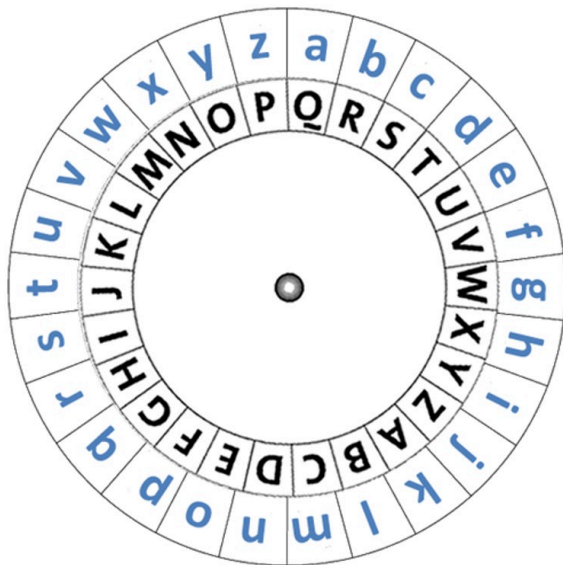


Abbildung 5: Cäsar-Scheibe
(Andreas Koch / CC BY-SA 4.0)

Die Implementierung der Entschlüsselung ergibt sich durch Anpassung der Zeilen 9 bis 11.

Geheimtexte einer monoalphabetischen Substitutionsverschlüsselung können mithilfe einer Häufigkeitsanalyse ohne Kenntnis des Verfahrens entschlüsselt werden, unter der Voraussetzung, dass der Klartext zumindest näherungsweise die charakteristische statistische Häufigkeit des zugrundeliegenden Sprachalphabets aufweist. Die Häufigkeiten der Geheimtextbuchstaben werden lediglich permutiert und erlauben so die Rekonstruktion der Zuordnung zum passenden Klartextbuchstaben, sofern o.g. Voraussetzung erfüllt ist. Ein Vergleich der Abbildungen 7 und 8 lässt den Schluss zu, dass der exemplarische Geheimtext Cäsar-verschlüsselt ist, da sich die Häufigkeiten lediglich um eine Konstante unterscheiden, die dem Schlüssel, der Verschiebezahl 3, entspricht.

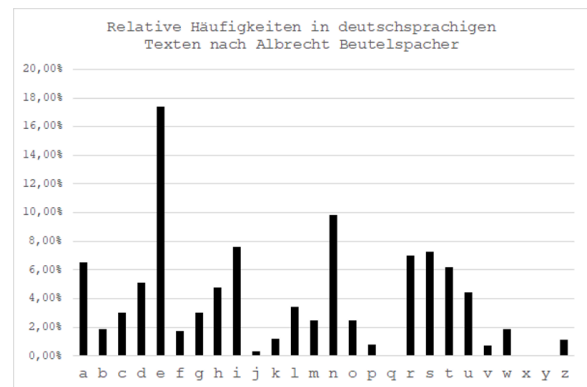


Abbildung 7: Relative Häufigkeiten in deutschsprachigen Texten nach Albrecht Beutelspacher. Grafische Aufbereitung vom Autor (Andreas Koch / CC BY-SA 4.0)

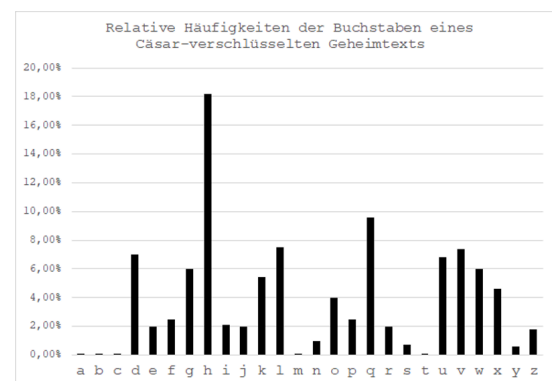


Abbildung 8: Relative Häufigkeiten der Buchstaben eines Cäsar-verschlüsselten Geheimtexts

Das Kerckhoffs'sche Prinzip ist demzufolge auch bei der Cäsar-Verschlüsselung mit Schlüssel nicht erfüllt.

Vigenère-Verschlüsselung

Die Vigenère-Verschlüsselung stammt vom französischen Kryptographen Blaise de Vigenère aus dem Jahr 1586. Sie setzt mehrere Cäsar-Verschlüsselungen positionsabhängig ein, womit sie ein Beispiel für eine polyalphabetische

```

01 public static String Caesar(String text, String schluessel)
02 {
03     String geheimtext = "";
04     for (int i=0; i<text.length(); i++) {
05         char c = text.charAt(i);
06         int z = (int) c;
07         int s = (int) schluessel.charAt( 0 );
08         if ((97<=z) && (z<=122)) {
09             z += s-97;
10             if (z>122) {
11                 z -= 26;
12             }
13         }
14         geheimtext += (char) z;
15     }
16     return geheimtext;
17 }

```

Abbildung 6: Java-Implementierung der Cäsar-Verschlüsselung (Andreas Koch / CC BY-SA 4.0)

Substitutionsverschlüsselung ist. Anstelle eines einzelnen Buchstabens werden Wörter als Schlüssel verwendet. Damit eignet sich eine Häufigkeitsanalyse, die auf einen Vigenère-verschlüsselten Geheimtext angewendet wird, im Allgemeinen nicht zu dessen Kryptoanalyse. Abbildung 9 zeigt exemplarisch, wie die relativen Häufigkeiten der Buchstaben um den Wert einer Gleichverteilung auf 26 Buchstaben streuen. Die Vigenère-Verschlüsselung glättet sozusagen die charakteristischen statistischen Häufigkeiten des Sprachalphabets und räumt damit die zentrale Schwachstelle monoalphabetischer Substitutionsverschlüsselungen aus.

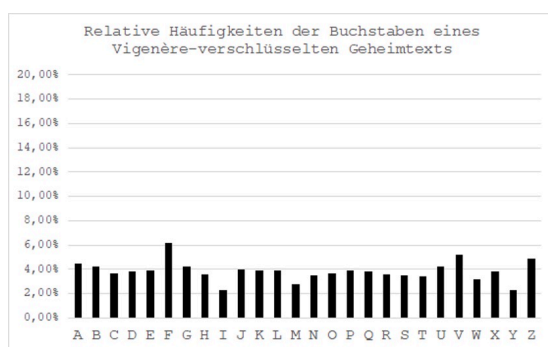


Abbildung 9: Relative Häufigkeiten der Buchstaben eines Vigenère-verschlüsselten Geheimtexts
(Andreas Koch / CC BY-SA 4.0)

Ver- und Entschlüsselung lassen sich wie in den Abbildungen 10 und 11 veranschaulichen.

Verschlüsselung mit Schlüssel ulm

Klartext	a	n	a	n	e	n
+	u	l	m	u	l	m
Geheimtext	L	Z	U	Y	Q	H

Abbildung 10: Vigenère-Verschlüsselung
(Andreas Koch / CC BY-SA 4.0)

Verschlüsselung mit Schlüssel ulm

Klartext	a	n	a	n	e	n
+	u	l	m	u	l	m
Geheimtext	L	Z	U	Y	Q	H

Cäsar-Verschlüsselung
mit Schlüssel u

Abbildung 11: Vigenère-Entschlüsselung
(Andreas Koch / CC BY-SA 4.0)

Ein methodisches Hilfsmittel für Schülerinnen und Schüler stellt das sogenannten Vigenère-Quadrat aus Abbildung 12 dar, das alle 26 Cäsar-Verschlüsselungen tabellarisch auflistet und so gegenüber der Cäsar-Scheibe, die vor jedem Ver- und Entschlüsselungsvorgang korrekt einzustellen ist, ein zügigeres Anwenden des Verfahrens ermöglicht.

Die Implementierung der Vigenère-Verschlüsselung lässt sich auf Basis der Cäsar-Verschlüs-

selung durch wenige Anpassungen vornehmen, wenn das Einlesen des Schlüsselbuchstabens wie in Abbildung 6 bereits als String erfolgt. Der Schlüssel muss zyklisch gelesen werden, da seine Länge n die des Texts unterschreiten kann, was der Regelfall ist. Dies lässt sich entweder mithilfe der Modulo-Arithmetik lösen, indem die Schlüsselposition durch $j \bmod n$ für Textpositionen j berechnet wird, oder wie in Abbildung 13 durch Verwenden einer weiteren Variable für den Schlüsselindex.

		Klartext																											
		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z		
Schlüsselbuchstabe	a	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z		
	b	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A		
	c	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B		
	d	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C		
	e	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D		
	f	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E		
	g	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F		
	h	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G		
	i	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H		
	j	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I		
	k	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J		
	l	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K		
	m	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L		
	n	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M		
	o	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N		
	p	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O		
	q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P		
	r	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q		
	s	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R		
	t	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S		
	u	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T		
	v	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U		
	w	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V		
	x	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W		
	y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X		
	z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y		

Abbildung 12: Vigenère-Quadrat
(Andreas Koch / CC BY-SA 4.0)

Ein Verfahren zur Kryptoanalyse der Vigenère-Verschlüsselung wurde erst ca. dreihundert Jahre nach ihrer Einführung publik. Der preußische General Friedhelm Kasiski veröffentlichte im Jahr 1863 ein Verfahren zur Bestimmung der Schlüssellänge n . Wenn diese bekannt ist, kann ein Vigenère-verschlüsselter Geheimtext in n Cäsar-verschlüsselte Geheimtexte durch Auswahl der jeweils n -ten Buchstaben zerlegt werden. Nun liefern n Häufigkeitsanalysen das Schlüsselwort.

Die Bestimmung der Schlüssellänge macht sich zu Nutze, dass ein Schlüsselwort im Allgemeinen erheblich kürzer als der Klartext ist. Dadurch ergeben sich im Geheimtext gleiche Textteile, wenn der Schlüssel bei seiner zyklischen Verwendung auf den gleichen Klartext trifft. Umgekehrt gilt: Mit einer Wahrscheinlichkeit von $1 - 26^{-2}$ sind gleiche Bigramme im Geheimtext mit dem gleichen Schlüsselteil verschlüsselt, gleiche Trigramme mit einer Wahrscheinlichkeit von $1 - 26^{-3}$ usw. Die Abstände gleicher Textteile im Geheimtext liefern demzufolge mit entsprechender Wahrscheinlichkeit Kandidaten für Vielfache der Schlüssellänge. Durch sukzessive Auswahl gleicher Primfaktoren dieser Ab-

```

01 public static String Vigenere(String text, String schluesel)
02 {
03     String geheimtext = "";
04     int j=0;
05     for (int i=0; i<text.length(); i++) {
06         char c = text.charAt(i);
07         int z = (int) c;
08         int s = (int) schluesel.charAt( j );
09         if ((97<=z) && (z<=122)) {
10             z += s-97;
11             if (z>122) {
12                 z -= 26;
13             }
14         }
15         geheimtext += (char) z;
16         j++;
17         if (j>schluesel.length()) {
18             j = 0;
19         }
20     }
21     return geheimtext;
22 }

```

Abbildung 13: Java-Implementierung der Vigenère-Verschlüsselung (Andreas Koch / CC BY-SA 4.0)

stände lässt sich so meist die Schlüssellänge berechnen wie Abbildung 14 beispielhaft zeigt.

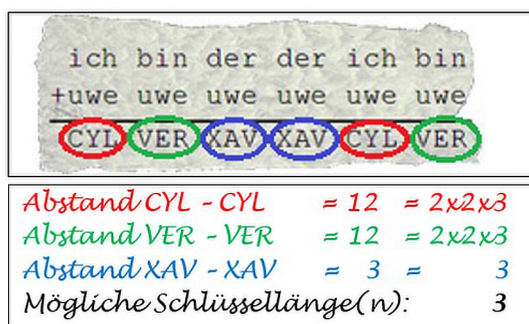


Abbildung 14: Bestimmung der Schlüssellänge mittels Kasiski-Verfahren (Andreas Koch / CC BY-SA 4.0)

Die Sicherheit des Vigenère-Verfahrens hängt somit von der Wahl des Schlüssels ab. Wenn er mindestens so lang wie der Klartext ist und die einzelnen Schlüsselbuchstaben für jeden Verschlüsselungsvorgang zufällig gewählt werden, spricht man von einem absolut sicheren Verfahren, einem One-Time-Pad (Einmal-Abriss-Block). In diesem Fall ist das Kerckhoffs'sche Prinzip erfüllt.

Visuelle Verschlüsselung

Das Verfahren der Visuellen Verschlüsselung von Moni Naor und Adi Shamir aus dem Jahr 1993 ist ein One-Time-Pad und erfüllt damit das Kerckhoffs'sche Prinzip. Durch Übereinanderlegen zweier Folien, von denen eine dem Geheimtext und die andere dem Schlüssel entspricht, entsteht das Originalbild. Für sich genommen enthalten die Folien laut Naor und Shamir nur „random noise“ (zufälliges Rauschen), d.h. sie

geben keinerlei Information über das Originalbild preis.

Am Beispiel von schwarz-weißen Pixelbildern lässt sich das Verfahren wie in Abbildung 16 veranschaulichen. Ein Pixel im Originalbild (schwarz oder weiß) wird jeweils zu vier Pixeln (grau) auf jeder Folie verschlüsselt. Die Wahl des Diagonalmusters auf der ersten Folie (Schlüssel) erfolgt zufällig, die Wahl des Diagonalmusters auf der zweiten Folie dazu gespiegelt.

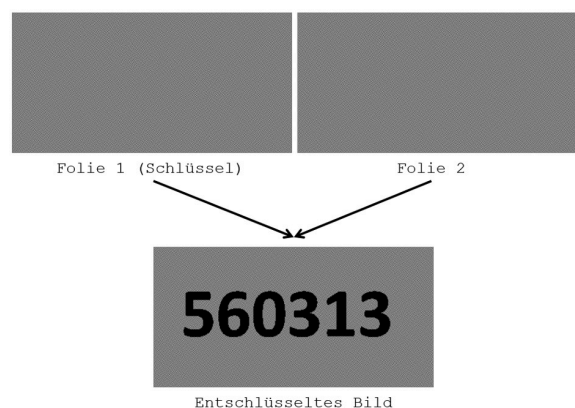


Abbildung 15: Visuelle Verschlüsselung (Andreas Koch / CC BY-SA 4.0)

Die Entschlüsselung erfolgt analog. Dabei ist zu beachten, dass weiße Pixel im Originalbild vier Pixeln mit Diagonalmuster entsprechen, wodurch der Eindruck eines grauen Pixels entsteht.

Das Verfahren lässt sich auf digitale schwarz-weiße Pixelbilder übertragen, indem eine passende Zuordnung der Pixelkonstellationen von schwarz, weiß und grau zu den Bits 0 und 1 erfolgt. Eine mögliche Variante zeigt Abbil-

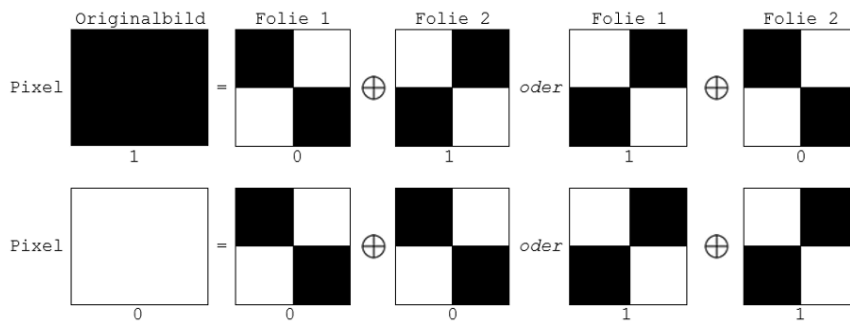


Abbildung 16: Verschlüsselung von schwarz-weißen Pixelbildern (Andreas Koch / CC BY-SA 4.0)

Abbildung 16: Zuerst werden dem schwarzen Pixel (im Beispiel 1) und dem weißen Pixel (0) die Bitwerte 0 bzw. 1 zugeordnet. Da die beiden Folien lediglich die binäre Information abbilden, ob die Diagonalmuster gleich sind und damit einem weißen Pixel im Originalbild entsprechen oder verschieden sind und damit einem schwarzen Pixel im Originalbild entsprechen, kann ihnen ebenfalls der Bitwert 0 oder 1 zugeordnet werden. Für die erste Folie wird er zufällig gewählt und für die zweite mithilfe der binären XOR-Funktion berechnet, um die Gleichheit bzw. Verschiedenheit der Diagonalmuster korrekt abzubilden. Der Bitwert des Pixels der Folie 2 berechnet sich für ein schwarzes Pixel im Originalbild und des durch 0 zufällig gewählten Pixelbitwerts auf Folie 1 zu $1 \text{ XOR } 0 = 1$. Wäre der Pixelbitwert auf Folie 1 stattdessen 1, so ist der Pixelbitwert auf Folie 2 durch $1 \text{ XOR } 1 = 0$ gegeben.

Abbildung 17 zeigt den Screenshot eines Programms zur Visuellen Verschlüsselung von schwarz-weißen Pixelbildern, das unter www.andreas-koch.de heruntergeladen werden kann.

Eine Implementierung ohne grafische Ausgabe zeigt Abbildung 18. Das Originalbild wird als String eingelesen, der nur mit 0 oder 1 zu belegen ist. So erleichtert sich die Verarbeitung, da beim Konkatenieren der Ausgabe-Strings

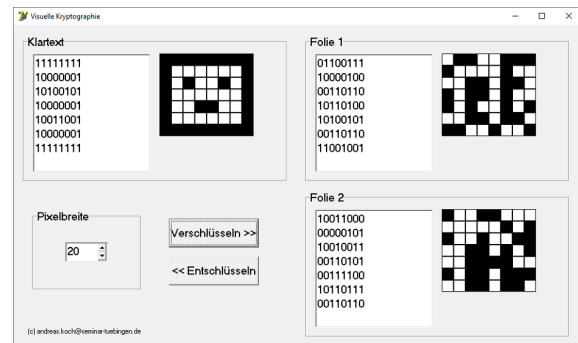


Abbildung 17: Screenshot eines Programms zur Visuellen Verschlüsselung (Andreas Koch / CC BY-SA 4.0)

keine Typecasts notwendig sind. Dadurch muss allerdings in Zeile 9 der ASCII-Wert der char-Variablen zuerst in einen int-Wert umgerechnet werden (ASCII-Code 48 bei 0 und 49 bei 1). Alternativ wäre ein Typecast mittels Methodenaufruf `Character.getNumericValue(c)` denkbar.

Eine Methode zur Entschlüsselung benötigt zwei Parameter, nämlich die Strings der Bitwerte der beiden Folien. Die Implementierung ist verglichen mit der Entschlüsselung leichter. Die Bitwerte der Strings müssen sukzessive XOR-verknüpft ausgegeben werden. Eine Zufallszahl muss nicht bestimmt werden.

Die Herleitung der Implementierung zeigt auch, dass das Verfahren verallgemeinert werden

```

01 public static void Visuell(String bits)
02 {
03     String folie1 = "";
04     String folie2 = "";
05     for (int i=0; i<bits.length(); i++) {
06         char c = bits.charAt(i);
07         int z = (int) (Math.random()*2); // 0 oder 1 würfeln
08         folie1 += z;
09         if (z != ((int) c - 48)) { // XOR
10             folie2 += "1";
11         } else {
12             folie2 += "0";
13         }
14     }
15     System.out.println( "Folie 1: " + folie1 );
16     System.out.println( "Folie 2: " + folie2 );
17 }

```

Abbildung 18: Java-Implementierung der Visuellen Verschlüsselung (Andreas Koch / CC BY-SA 4.0)

kann: Die Art des Musters der Folien muss sich nicht auf Diagonalen beschränken. Die Anzahl der Folien kann auf n Stück ausgeweitet werden, indem die Pixelzahl pro Originalbildpixel auf $n \cdot n$ Stück erhöht wird.

Unterrichtsgang

Die vorgestellten Verfahren eignen sich zur Ausgestaltung eines acht Stunden umfassenden Unterrichtsgangs zum Thema „Symmetrische Kryptologie“ unter Veranschlagung von jeweils einer Doppelstunde pro folgendem Verfahren: Transposition am Beispiel der Skytale, monoalphabetische Substitution am Beispiel der Cäsar-Verschlüsselung, polyalphabetische Substitution am Beispiel der Vigenère-Verschlüsselung und One-Time-Pad am Beispiel der Visuellen Verschlüsselung.

Die vorgestellten Veranschaulichungen ermöglichen auch einen Unterricht in der Sekundarstufe I. Hier sollte der Schwerpunkt auf der händischen Durchführung und Beurteilung der Sicherheit der Verfahren liegen. In der Sekundarstufe II sollte er auf die Implementierung der Verfahren gelegt werden. Nach einer gemeinsamen Erarbeitung der Implementierung der Verschlüsselung bietet sich zur Lernzielkontrolle eine Übungsphase an, in der die Schülerinnen und Schüler die Entschlüsselung selbstständig implementieren. Zu beachten ist, dass bei der Skytale der Schwierigkeitsgrad einer Implementierung der Entschlüsselung aufgrund der notwendigen Anpassung der Indizes gegenüber der Verschlüsselung höher als bei den anderen Verfahren ist.

Java-Implementierungen aller Verfahren für BlueJ inklusive einer Vorlage für Schülerinnen und Schüler können angefragt werden.

Die Diskussion des logistischen Problems des Schlüsselaustauschs bei symmetrischen Verfahren ist ein sinnvoller Ausgangspunkt für die Einführung asymmetrischer Verfahren.

Quellen

Alle Webseiten/Links wurden zuletzt geprüft am 02.01.2024.

Beutelspacher, A. (2005): Kryptologie. 7. Auflage. Vieweg Verlagsgesellschaft, Wiesbaden, Seite 10.

de Vigenère, B. (1586): Traicté des Chiffres ou Secrètes Manières d'Ecrire. Paris.

Kasiski, F. (1863): Geheimschriften und die Dechiffrierkunst – Mit besonderer Berücksichtigung der deutschen und der französischen Sprache. Mittler und Sohn, Berlin.

Ministerium für Kultus, Jugend und Sport Baden-Württemberg (2016): Bildungsplan Aufbaukurs Informatik 7. <https://www.bildungsplaene-bw.de/Lde/LS/BP2016BW/ALLG/SEK1/INF7>.

Naor, M.; Shamir, A. (1994): Visual Cryptography, EUROCRYPT, S. 1-12.

Lizenz



Dieser Artikel steht unter der Lizenz CC BY-NC 4.0 zur Verfügung.

Kontakt

Andreas Koch
E-Mail: kocha@posteo.de

Fachfremd unterrichten, selbstreguliert lernen – eine Lösung für den Informatikunterricht

Lenk, S.

DOI: 10.18420/ibis-02-02-08

Zusammenfassung

Seit drei Jahren unterrichte ich Informatik fachfremd an unserer Oberschule. Meine anfängliche Euphorie stieß schnell an die Grenzen einer Überlastung. Doch dann fand ich mit der Methode "Modern Classrooms Project" einen Weg des Blended Learning, das Fach entspannt und erfolgreich zu unterrichten.

Einleitung

Neben meiner falschen Vorstellung davon, dass sich „Digital Natives“ natürlicherweise für Informatik begeistern, war mein größtes Problem die Überforderung das neue Fach im Computerraum zu unterrichten. Sie ergab sich aus dieser Vielzahl an Schwierigkeiten:

1. Der Computer lenkte die SchülerInnen von meinen Erklärungen ab.
2. Durch die Tischreihen konnte ich von vorn nicht auf die Monitore der SchülerInnen sehen.
3. Fehlende Disziplin und impulsives Verhalten erzeugten zu viel Krach.



Abbildung 1: Computerraum vorher: Tischreihen frontal ausgerichtet (Foto von S. Lenk)

4. Die meisten SchülerInnen brauchten gleichzeitig individuelle Unterstützung, die ich allein nicht schnell genug geben konnte.
5. Das Arbeitstempo der SchülerInnen war sehr unterschiedlich.
6. Kontinuität im Lernen war bei einer Stunde pro Woche schwierig herzustellen.
7. Ausfall, Abwesenheit und neue SchülerInnen, die im Schuljahr dazu kamen, erschwerten es, den Überblick über das Lernen aller zu behalten.

Raus aus dem Frontalunterricht mit dem „Modern Classrooms Project“

Als Englischlehrerin recherchiere ich pädagogische Fragen auch im englischsprachigen Raum. So entdeckte ich das „Modern Classrooms Project“ von Kareem Farah aus den USA. Er hatte es 2018 als gemeinnützige Initiative gegründet, um Lehrkräften dabei zu helfen, Lernräume zu gestalten, die individuelles Lernen ermöglichen, indem die Lehrkraft aus der frontalen Position herausgelöst wird. Das Modell stützt sich auf drei Säulen: Lernen im eigene Tempo, Weiterlernen nur nach Erreichen eines Lernziels und Einsatz von Technik, um selbstreguliertes Lernen innerhalb einer Gemeinschaft zu fördern. Der Online-Kurs dauerte nur wenige Stunden und war frei zugänglich. Als ich ihn absolviert hatte, war ich überzeugt, dass diese Methode meine Unterrichtssituation verbessern könnte.

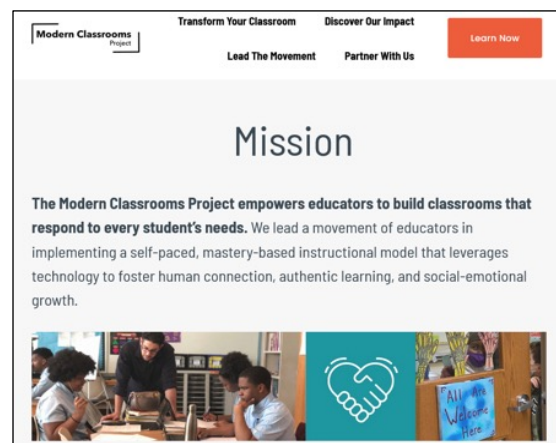


Abbildung 2: Webauftritt des „Modern Classroom Project“ - www.modernclassrooms.org

Während der Sommerferien stellte ich meinen Raum und meine Inhalte auf das Konzept des „Modern Classrooms Project“ um. Ich veränderte die Tischordnung, um den Raum zu öffnen, Platz zu schaffen für Zusammenarbeit und nicht zuletzt, um die Bildschirme besser im Blick zu haben.



Abbildung 3: Computerraum mit offener Tischordnung
(Foto von S. Lenk)

Ordnung des Materials

Die Technik wird in diesem Modell als die Lehrkraft entlastendes Unterrichtsmedium eingesetzt. Das heißt Lektionen werden in 5-Minuten-Videos eingesprochen und die SchülerInnen schauen sie selbstständig. Dazu gibt es sogenannte „begleitete Notizen“, z.B. in Form von Arbeitsblättern, die das Wesentliche des Lernstoffs festhalten. Durch die Übungen führen digitale Aufgaben am Computer. Sie können verschiedene Formate haben. Die Umsetzung kann auf jeder digitalen Lernplattform erfolgen. Ich habe dazu die Lernmodule auf Lernsax genutzt, um interaktive Übungen am Computer zu gestalten, aber auch, um zu Partnerarbeiten zu animieren oder die SchülerInnen zu analogem Material im Raum zu leiten. Die Materialien sind entsprechend ihrer digitalen Struktur in physischen Haupt- und Unterordnern sortiert und stehen in der Reihenfolge des Lehrplans in of-



Abbildung 4: Materialboxen mit Arbeitsblättern in Hängordnern, daneben Anschauungsmaterial
(Foto von S. Lenk)

fenen Regalen. Neben den Arbeitsblättern finden die SchülerInnen dort auch weiteres Anschauungsmaterial. Für den Lernbereich „Informatik im Alltag“ gibt es z.B. PCs zum aufschrauben. Im Fach für den Lernbereich „Algorithmen“ stehen Mikrocontroller (Calliope Mini) bereit und im Lernbereich „Netzwerke“ Kabeltüten, um unterschiedliche Kabelarten anzuschauen.

Bewertung

Die Feststellung des Lernfortschritts erfolgt durch digitale Tests (für Fachbegriffe) oder Projektaufgaben, bei denen die SchülerInnen Kompetenzen zeigen, z.B. Programmieren, Datenbank bauen oder Mediengestaltung mit Anwendersoftware. Die Projektaufgaben folgen den didaktischen Prinzipien PRIMM oder UMC - d.h. die SchülerInnen untersuchen vorhandene Codes, z.B. in Scratch, ändern diese ab und leiten aus ihren Beobachtungen eigene Ideen für Abfolgen ab. Bei der Bewertung weiche ich (noch) vom Modell des „Modern Classroom Projects“ ab, weil ich die Ergebnisse der SchülerInnen an meiner Schule alle vier Wochen als Noten festhalten muss. Sie können also eher nicht, wie im Modell vorgesehen, solche Tests und Projekte so lange wiederholen, bis sie sie meistern. Einen kleinen Spielraum zur Überarbeitung gewähre ich dennoch auf freiwilliger Basis und ändere dann im Nachgang Noten, wenn nötig.

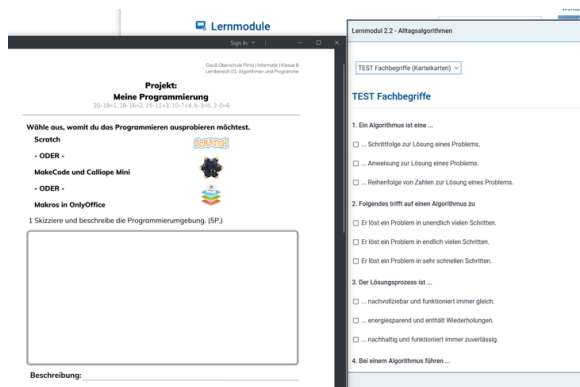


Abbildung 5: Projektaufgabenblatt (links) und digitaler Test (rechts) (Screenshot von S. Lenk)

Dokumentation mit dem Lernplan

Für jeden Lernbereich habe ich ein Deckblatt erstellt (wie bei der Wochenplanarbeit), das einen Überblick über die durchzuarbeitenden Themen gibt. Die Schüler notieren bei Anwesenheit das Datum, um zu dokumentieren, wann sie an etwas gearbeitet haben. Ich kontrolliere nach jeder Stunde die Hefter und den Fortschritt in den digitalen Lernmodulen und

verteile Punkte für die Vollständigkeit von Lektionsnotizen und Übungsaufgaben. So sehen die SchülerInnen zu Beginn jeder neuen Stunde, wo sie sich im Lernplan befinden und finden schnell wieder den Einstieg dort, wo sie aufgehört haben. Ich mache mir Notizen zu einzelnen SchülerInnen, wenn mir bei der Kontrolle etwas auffällt und unterstütze dann gezielt bei Schwierigkeiten.

Mein Lernfortschritt in Lernbereich 2 - Netzwerke				
Stunde	Lernbereich/ Lernmodul	Thema	Datum	Kontrolle/ Note
01	LB2/ 1.1	Möglichkeiten der Datenübertragung		
02	LB2/ 1.2	Bedeutung von Netzwerken		
03	LB2/ 1.3	Netzwerkarchitekturen		
04				Fachbegriffe Test
05	LB2/ 2.1	Aufbau eines Netzwerks mit Filius		
06	LB2/ 2.2	Netzwerkprotokolle		
07	LB2/ 2.3	Dienste		
08				Netzwerke- Werkstatt
09	LB2/ 3.1	Begriff Kryptografie		
10	LB2/ 3.2	Klartext, Geheimentext, Schlüssel		
11	LB2/ 3.3	einfache symmetrische Verschlüsselungsverfahren		
12	LB2/ 3.4	Verschlüsselung in der Gegenwart		Kryptografie Führerschein
13	WB			
14	WB			
15				
16				LB2 Hefter/ Lernmodule

Abbildung 6: Deckblatt für Lernbereich Netzwerke zur Dokumentation des Lernfortschritts (Screenshot von S. Lenk)

Probleme gelöst

Die zu Beginn aufgelisteten Probleme konnte ich mit Hilfe des Modells sehr gut lösen, denn

1. Der Computer steht nun als Unterrichtsmedium im Mittelpunkt.
2. Die veränderte Tischordnung zeigt mir die Monitore und ich sehe sofort, wenn jemand nicht am Thema arbeitet.
3. Ein großer Teil der SchülerInnen kann nun selbstständig arbeiten und ist damit voll beschäftigt.
4. Weniger SchülerInnen brauchen enge Begleitung, die ich jetzt leichter gewähren kann.
5. Es ist fast kein Problem mehr, dass es unterschiedliches Arbeitstempo gibt.
6. Die SchülerInnen können sich in der Folgestunde nach einer Woche selbstständig in

wenigen Minuten wieder in das Thema einfinden.

7. Durch die Dokumentation des Lernfortschritts auf dem Deckblatt sind Ausfälle, Abwesenheit und Neuzugänge kaum noch problematisch.

Fazit und Ausblick

Nach Abschluss des dritten Schuljahres fühle ich mich wohl in der Routine, die mir das Konzept des "Modern Classroom Project". Insbesondere die größere Nähe zu den SchülerInnen durch die vielen individuellen Gespräche lassen sie mich gut kennenlernen, obwohl ich sie nur ein Mal in der Woche sehe. Das gibt mir als Pädagogin eine stabile Beziehungsbasis für meine Arbeit. Die Abnahme der ersten mündlichen Prüfung in diesem Schuljahr wurde durch die hohe Transparenz der vorbereiteten Lernumgebung digital und analog sehr erleichtert. Die Prüflinge können sämtliche Lernmodule zur Wiederholung abspielen und durchklicken. Da die Hefter zur Kontrolle kaum den Raum verlassen, haben sie weitestgehend vollständige Mitschriften oder können diese mühelos in der Dateiablage einsehen. In jedem Lernbereich entsteht eine Sammlung an Fachbegriffen auf Karteikarten, die ebenfalls im Hefter aufbewahrt werden und vor der Prüfung gelernt werden können.



Abbildung 7: Hefterablage der Lernenden - Sammlung von Mitschriften und Fachbegriffen (Foto von S. Lenk)

Dennoch bleiben Probleme, an deren Lösung ich in den kommenden Schuljahren weiter arbeiten möchte.

Erstens: Nicht alle SchülerInnen können sofort oder immer selbstreguliert lernen. Sie anzuleiten und den Prozess zu begleiten erfordert Mühe und Geduld. Manche meiner SchülerInnen benötigen bis heute engen Kontakt und viel Bestätigung, damit sie kleine Schritte selbstständig gehen können. Es gelingt nicht, dies in jeder Stunde für alle zu leisten, besonders dann, wenn durch Planänderungen die ganze Klasse statt einer Gruppe unterrichtet werden muss.

Zweitens: Viele Stunden laufen ohne richtige Interaktion ab. Um den Stundenanfang und das -ende probiere ich gerade verschiedene Abläufe aus, die ein kurzes Zusammenkommen der Lerngruppe sinnvoll gestalten. Während der Stunden ist es manchmal schwierig, dass die SchülerInnen in eine Partnerarbeit finden, da ja nicht alle zur selben Zeit am selben Punkt sind.

Drittens: In den Grenzen von Lehrplan, Zeitrichtwerten und Regelungen der Schule steht das Abarbeiten des Lernplans noch zu sehr im Vordergrund. Für lustvollen, neugierigen oder kreativen Umgang mit der Informatik fehlt den Lernenden einfach die Zeit. Hier überlege ich, die Folge der Lernbereiche frei wählen zu lassen oder auch innerhalb der Bereiche die Folge der Themen nach Interesse bearbeiten zu können, um das strenge Zeitraster etwas aufzulockern.

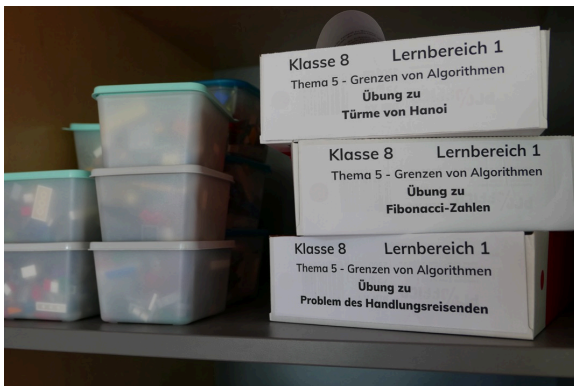


Abbildung 8: Zusatzmaterial "zum anfassen": Legokisten und Phänomene für Grenzen von Algorithmen
(Foto von S. Lenk)

Quellen

Alle Webseiten/Links wurden zuletzt geprüft am 01.07.24

www.modernclassrooms.org

www.lernsax.de

Lizenz



Dieser Artikel steht unter der Lizenz CC BY NC 4.0 zur Verfügung.

Kontakt

Susann Lenk
Carl Friedrich Gauß Oberschule
E-Mail: lenk.s@osgauss.lernsax.de

ClusterLabor: Ein Werkzeug zur interaktiven Visualisierung und Analyse von Clusteralgorithmen

Andres, D., Joachim, S. und Hennecke, M.

DOI: 10.18420/ibis-02-02-09

Zusammenfassung

In diesem Beitrag wird die Webanwendung ClusterLabor (verfügbar unter <https://www.ddi.informatik.uni-wuerzburg.de/cluster>) vorgestellt. ClusterLabor ermöglicht eine interaktive Visualisierung und Analyse von Clusteralgorithmen in zweidimensionalen Datensätzen. Damit können verschiedene Algorithmen hinsichtlich ihrer Ergebnisse in Abhängigkeit von der gewünschten Anzahl an Clustern verglichen werden. Anwender können aus verschiedenen Beispieldatensätzen wählen, eigene Datensätze hochladen oder Datensätze direkt durch manuelle Eingabe generieren. Zum Clustern stehen verschiedene Methoden zur Verfügung: der *k-Means-Algorithmus* mit Varianten wie Lloyd oder MacQueen, der *k-Medoids-Algorithmus* sowie *hierarchische Clusteranalyse* mit unterschiedlichen Distanzmaßen und Fusionierungsalgorithmen. Ein besonderer Fokus liegt dabei auf dem Unsupervised Learning, einem Bereich der Künstlichen Intelligenz (KI), bei dem Algorithmen Muster und Strukturen in unbeschrifteten Daten selbstständig erkennen. Zur Bestimmung der "optimalen" Clusterzahl k sind zudem Visualisierungen des Elbow Plots (Ellenbogen-diagramms), des Average Silhouette Plots (ASW-Kurve) sowie des Dendrogramms integriert.

Einleitung

Politik und Wirtschaft schreiben Datenanalysetechniken ein großes Innovationspotential zu. Diesen Techniken erkennen in immer größer werdenden Datenbeständen Strukturen und Muster und gewinnen so neue Informationen. Gleichzeitig stehen vielseitige Bedenken im gesellschaftlichen Diskurs: Seien es die Gefahren für den Datenschutz oder die für Fachfremde oft intransparenten Abläufe und Algorithmen (BT 20/5149).

Zur Versachlichung der Diskussion besteht die Möglichkeit fachliche Kompetenzen zur Funktionsweise von und zum Umgang mit KI-Systemen bereits in der Schule zu vermitteln. Einige Lehrpläne haben die Thematik bereits aufgegriffen und integrieren z. B. bekannte Clusteralgorithmen aus dem Bereich der Datenanalyse in ihre Curricula. Clusteralgorithmen sind Methoden des maschinellen Lernens, die dazu

dienen, ähnliche Datenpunkte in Gruppen oder Clustern einzuteilen. Beispielsweise kann durch diese Algorithmen aus einem Datensatz der Bildschirmgröße (Breite und Länge) von Geräten, die auf eine Website zugreifen, eine Aufteilung in verschiedene Cluster ermittelt werden. Davon ausgehend können die entstandenen Cluster als Gruppen von Gerätetypen (z. B. "Handy", "Tablet" oder "Laptop") interpretiert werden. Ein Beispiel für einen solchen Lehrplan, ist der bayerische LehrplanPLUS (ISB o. J.), der in Jahrgangsstufe 13 die Implementierung des *k-Means-Algorithmus* und die Analyse der entstehenden Cluster in Abhängigkeit von der Clusterzahl k fordert. Mithilfe von ClusterLabor können einerseits viele dieser Anforderungen im Unterricht umgesetzt und andererseits die Thematik der Clusteralgorithmen didaktisch reduziert vertieft werden.

Verwandte Arbeiten

Für die Visualisierung und die Implementierung des *k-Means-Algorithmus* gibt es bereits mehrere Werkzeuge. Neben diversen Möglichkeiten im Web, die einzelnen Schritte des *k-Means-Algorithmus* zu visualisieren, existiert z. B. mit Andres (2024) eine Möglichkeit, den *k-Means-Algorithmus* unplugged einzuführen und anschließend mit einer Vorlage in der Programmierungsumgebung BlueJ umzusetzen. Eine Möglichkeit, die entstandenen Cluster in Abhängigkeit von k zu analysieren, besteht jedoch nicht.

Eine Analyse der Qualität von verschiedenen Clusterungen bieten unterschiedliche bereits existierende Werkzeuge zum Data Mining:

Orange (Demšar 2013) ist ein Open-Source-Werkzeug für Datenvisualisierung, maschinelles Lernen und Data Mining. Neben vielen anderen Funktionen ist auch das Clustern von Daten mittels des *k-Means-Algorithmus* oder des hierarchischen Clusterings möglich. Gleichzeitig stehen verschiedene Möglichkeiten zur Visualisierung der entstandenen Cluster zur Verfügung. Orange stellt zudem viele weitere Funktionen im Bereich Data Mining bereit. Aufgrund des großen Funktionsumfangs ist es nicht didaktisch auf die Visualisierung des *k-Means-Algorithmus* oder verwandter Algorithmen reduziert. Schülerinnen und Schüler müssen erst in

das Programm eingewiesen und durch die einzelnen Arbeitsschritte und Möglichkeiten geführt werden. Eine mögliche Verwendung von Orange im Zusammenhang mit Clusteralgorithmen im Unterricht der neunten Jahrgangsstufe zeigt Pöhner (2023). In der vorgestellten Unterrichtssequenz werden mittels des Clusteranalysetools in Orange3 Filterblasen in sozialen Medien thematisiert.

Auch Bibliotheken aus R oder Python können verwendet werden, um verschiedene Clusterungen von unterschiedlichen Algorithmen und Methoden darzustellen und miteinander zu vergleichen. Diese Bibliotheken erfordern jedoch ein gewisses Maß an Einarbeitung und zusätzlich ein Grundverständnis der zugrundeliegenden Sprache, was im Schulbetrieb, sollte zuvor noch nicht mit R oder Python gearbeitet worden sein, einen erheblichen Aufwand darstellt.

Neben Abo- und kostenpflichtigen Versionen von professionellen Werkzeugen zur Datenanalyse gibt es auch kostenlose Werkzeuge wie Weka (Holmes 1994), welche mächtige Werkzeuge zur allgemeinen Datenanalyse sind. Diese gehen jedoch weit über den schulischen Aspekt des Clusters von Daten mittels einfacher Algorithmen hinaus und sind nicht für die didaktische Arbeit reduziert.

Anforderungen an das Programm

Ziel von ClusterLabor ist es, Schülerinnen und Schülern ein Programm zur Verfügung zu stellen, mit dem sie ohne große Einarbeitung Clusteralgorithmen ausführen und sich die Ergebnisse visuell anzeigen lassen können. Insbesondere sollen die Schülerinnen und Schüler einfach mit der Anzahl k der zu bildenden Cluster, dem zentralen Parameter aller Cluster-Verfahren, experimentieren können. Bezogen auf den k -Means-Algorithmus entspricht dies z. B. den Anforderungen der Jahrgangsstufe 13 des bayerischen Lehrplans:

"Die Schülerinnen und Schüler [...] analysieren für verschiedene Eingabedaten die Ergebnisse, die der k -Means-Algorithmus in Abhängigkeit von k liefert." (ISB o.J.)

Ziel dieses Projektes war es, ein Programm zu entwickeln, mit dem dieser Lehrplanpunkt in der Unterrichtspraxis umgesetzt werden kann.

Daraus folgen direkt drei Anforderungen an ein solches Programm: Es müssen verschiedene Datensätze in das Programm geladen werden, das Programm muss in der Lage sein, diese Daten mithilfe eines Clusteralgorithmus zu clustern, und die entstandenen Cluster in Abhän-

gigkeit von k anzuzeigen. Es soll möglich sein zwischen verschiedenen Clusterzahlen k zu wechseln und so die Ergebnisse des Algorithmus zu vergleichen. Das Ziel dieses Vergleichs ist es, die Anzahl der Cluster k zu ermitteln, mit der die Aufteilung als "optimal" angesehen werden kann. Je nach Lehrplan kann es möglich sein, dass diese Analyse über die Betrachtung mit bloßem Auge hinausgehen soll und die Qualität einer Clusterung systematisch eingeschätzt und beurteilt werden muss. Dafür stehen in der Fachwissenschaft verschiedene Metriken und grafische Diagramme zur Verfügung, welche somit auch in ClusterLabor integriert werden sollen.

Der bayerische Lehrplan fordert beispielsweise nicht, dass der k -Means-Algorithmus über die Variation der Clusterzahl hinaus verändert werden soll. Dennoch ist eine Anpassung des k -Means-Algorithmus ein naheliegender Gedankengang. Schülerinnen und Schüler stoßen beim Experimentieren mit dem implementierten Algorithmus schnell auf die Einschränkung des k -Means-Algorithmus, konvexe und gleich große Cluster zu bevorzugen, was zu nicht sinnvollen Clusterergebnissen bei konkaven Datensätzen wie z. B. "zweiMonde" führen kann. Zum tieferen Verständnis des k -Means-Algorithmus sollen in dem Programm folglich Anpassungen der Metrik, der Art der Zentrumsbildung, der Wahl der Startzentren sowie der Zeitpunkt der Aktualisierung des Zentrums (Lloyd vs. MacQueen) möglich sein. Dadurch können die Schülerinnen und Schüler die Auswirkungen dieser Parameter auf den Algorithmus betrachten, ohne dass sie diese oder eine passende Visualisierung selbst programmieren müssen.

Ein differenziertes Bild, welches verschiedene Clusteralgorithmen, deren Stärken und Schwächen und damit folgend ihre unterschiedlichen Einsatzgebiete aufzeigt, kann im Unterricht nicht ohne weiteres entstehen. Deshalb ist eine weitere Forderung an das Programm, dass es verschiedene Clusteralgorithmen zur Verfügung stellt und die Ergebnisse unterschiedlicher Algorithmen miteinander vergleichbar macht. Somit kann den Schülerinnen und Schülern die Fülle an unterschiedlichen Methoden und Algorithmen in der Clusteranalyse aufgezeigt werden, ohne zuvor im Unterricht tiefgehende fachliche Grundlagen geschaffen zu haben. Der vom Lehrplan vorgesehene Clusteralgorithmus kann am Ende einer solchen Sequenz mit seinen Stärken und Schwächen als einer von vielen Algorithmen des Data Mining eingeordnet werden.

Fachlicher Hintergrund

Der k -Means- und der k -Medoids-Algorithmus

Beim k -Means-Algorithmus werden zunächst k Clusterzentren auf k zufällig gewählten Punkten des Datensatzes initialisiert. Danach werden alle Datenpunkte dem Zentrum zugeordnet, dem sie am nächsten sind. Anschließend erfolgt die Neuberechnung der Clusterzentren über Mittelwertbildung. Die so berechnete Zentrumposition muss insbesondere mit keinem echten Datenpunkt des Datensatzes übereinstimmen. Diese beiden Schritte werden so lange wiederholt, bis sich die Position der Clusterzentren nicht mehr ändert (vgl. Ertel 2021). Die linke Spalte von Abb. 1 zeigt den schematischen Ablauf des k -Means-Algorithmus. Der k -Means-Algorithmus hat eine gewisse Ähnlichkeit mit dem Konzept des Schwerpunkts in der Physik, da beide Methoden darauf abzielen, einen zentralen Punkt zu bestimmen, der die "Mitte" eines Systems repräsentiert: Der k -Means-Algorithmus berechnet das Clusterzentrum eines Clusters als den Mittelwert aller Punkte im Cluster, ähnlich wie der Schwerpunkt in der Physik. Der Schwerpunkt eines Körpers ist der Punkt, an dem die gesamte Masse des Körpers so betrachtet werden kann, als ob sie dort konzentriert wäre. Er ist der Durchschnitt der Positionen der Massenpunkte, gewichtet nach ihrer Masse.

Ein wesentlicher Aspekt des k -Means-Algorithmus ist die Wahl der Startzentren, auch Initialisierung genannt. Diese können erheblichen Einfluss auf die endgültige Clusterbildung haben. Verschiedene Startzentren können zu unterschiedlichen Clustern führen. Dies verdeutlicht Abb. 2, die zeigt wie unterschiedliche Ergebnisse des k -Means-Algorithmus in Folge unterschiedlicher Initialisierungen entstehen. Der Datensatz kann per Augenmaß leicht in drei Cluster unterteilt werden, welche aber nur in einem der drei Fälle ermittelt werden. Das Beispiel zeigt, wie stark die Wahl der Startzentren das Endergebnis des Algorithmus beeinflusst.

Der k -Means-Algorithmus reagiert zudem empfindlich auf Ausreißer, da der gebildete Mittelwert für die neuen Koordinaten eines Zentrums leicht durch Ausnahmen beeinflusst wird.

Der k -Medoids-Algorithmus ist eine Variante des k -Means-Algorithmus, der robuster gegenüber Rauschen und Ausreißern ist. Anstatt eines Mittelwertes wird ein sogenannter Medoid gebildet. Ein Medoid ist der am zentralsten gelegene Datenpunkt eines Clusters, d. h. der Datenpunkt mit der geringsten Summe der Abstände zu allen anderen Punkten des Clusters. Man kann sich den Medoid eines Clusters als den "repräsentativsten" Punkt im Cluster vorstellen, ähnlich dem Median in der Statistik, der den "zentralsten" Wert einer sortierten Liste darstellt. Der Medoid bleibt robust gegenüber Ausreißern, da er ein tatsächlicher Datenpunkt

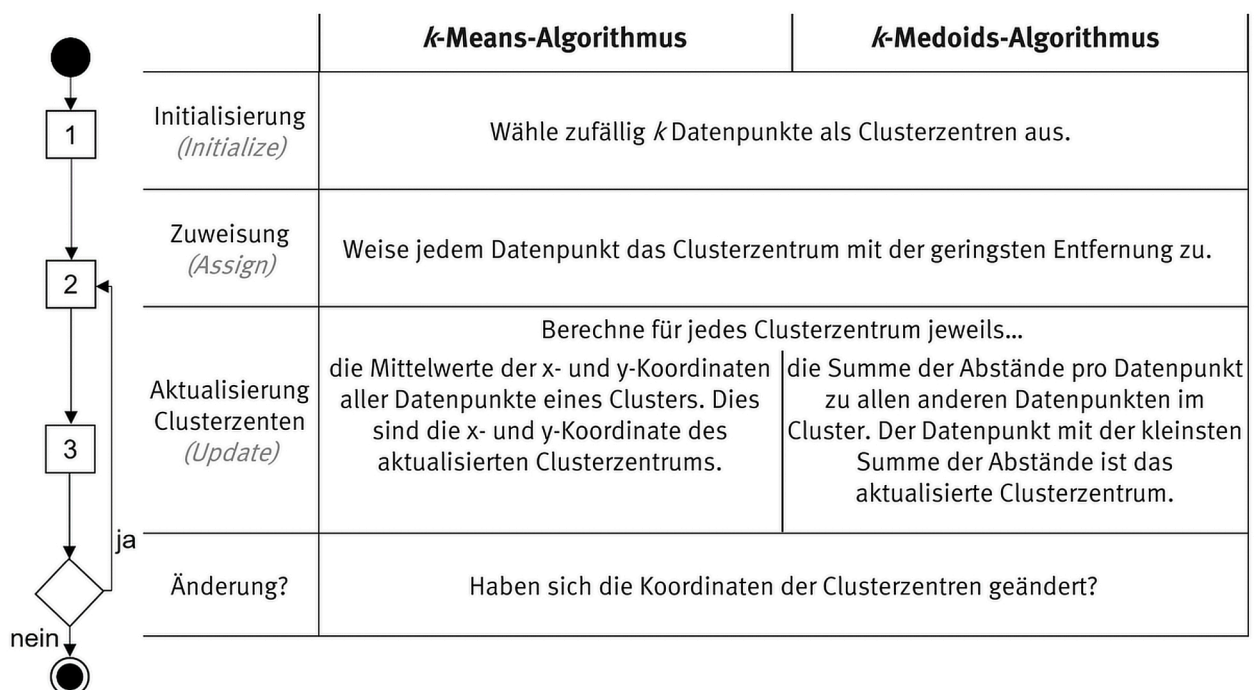


Abbildung 1: Schematischer Ablauf des k -Means- und des k -Medoids-Algorithmus

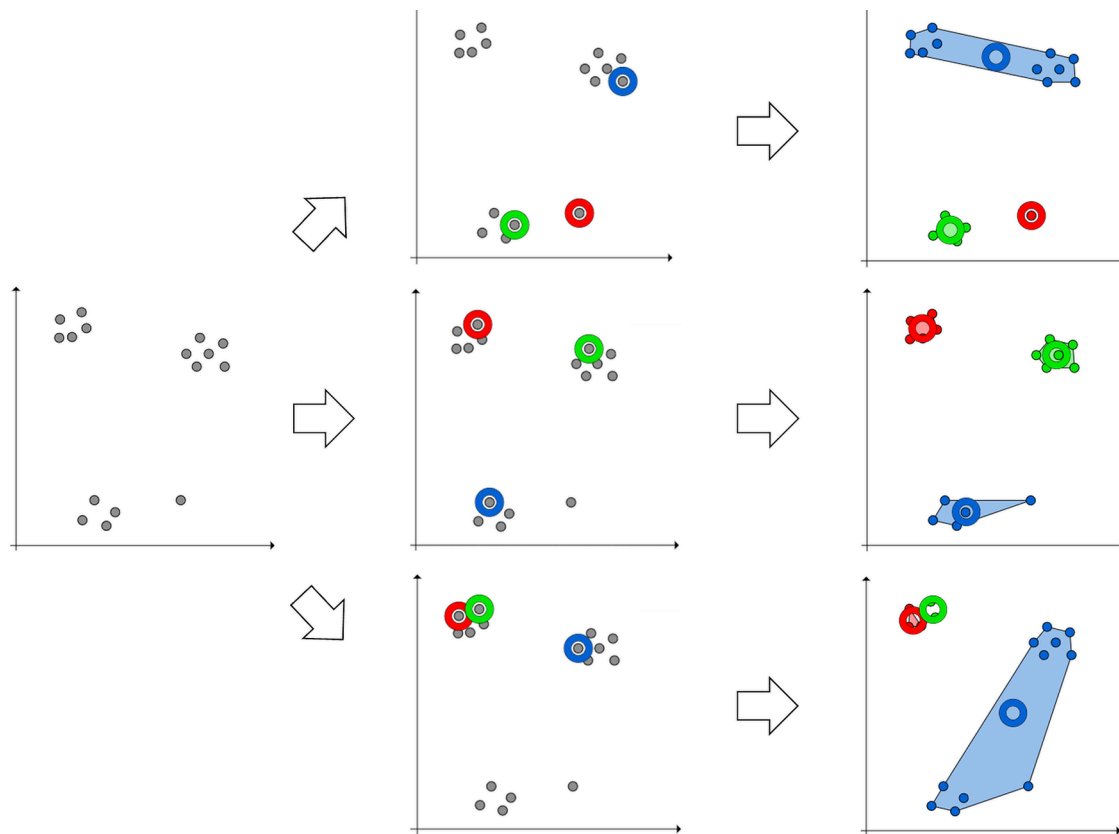


Abbildung 2: Einfluss der Wahl der Startzentren auf die durch den k -Means-Algorithmus entstehenden Cluster.
 Erste Zeile: Der Ausreißer-Datenpunkt wurde als Startzentrum gewählt. Aufgrund der Verteilung der anderen Startzentren ist das rote Zentrum jedoch für keinen anderen Datenpunkt das Zentrum mit der geringsten Distanz und wird demnach nie aktualisiert.
 Zweite Zeile: Günstige Verteilung der Startzentren führt zu dem erwarteten Cluster-Ergebnis.
 Dritte Zeile: Die oberen Startzentren wurden zu nah aneinander gewählt und teilen sich somit das obere Cluster. Visualisierungen mit ClusterLabor

ist, der die kleinsten summierten Distanzen zu anderen Punkten im Cluster hat. Abb. 1 zeigt den schematischen Ablauf des k -Medoids-Algorithmus in Gegenüberstellung zum k -Means-Algorithmus.

Eine bekannte Umsetzung des k -Medoids-Algorithmus ist die Partitionierung um Medoide (Partitioning Around Medoids, PAM), welche auch für die Umsetzung des k -Medoids-Algorithmus in ClusterLab gewählt wurde (vgl. Jin 2010).

Bestimmen der "optimalen" Clusterzahl k mithilfe des k -Means- oder k -Medoids-Algorithmus

Bei realen Datensätzen ist die Anzahl k der Cluster oft nicht im Vorhinein bekannt. Oft gibt es in Abhängigkeit vom spezifischen Anwendungsfall und den Eigenschaften der Daten einen größeren Bereich von k -Werten, die sinnvoll sind. Deshalb stellt sich hier die Frage nach der "optimalen" Clusterzahl k . Die Wahl von k mag in einigen Fällen durch Darstellung der Punkte in einem Koordinatensystem mit dem

bloßen Auge möglich sein, bei höheren Dimensionen versagt diese Methode jedoch. Deshalb wurden verschiedene Heuristiken entwickelt, um ein möglichst gut passendes k bestimmen zu können.

Eine der ältesten und einfachsten Möglichkeiten wird oft in Verbindung mit dem k -Means-Algorithmus angewandt: Die *Ellenbogenmethode* (Elbow Plot). Dafür wird der vorliegende Datensatz zunächst für verschiedene Clusterzahlen k geclustert. Das Ellbogendiagramm zeigt die WCSS-Werte (Within Cluster Sum of Square) der einzelnen Clusterungen auf der y-Achse und die dazugehörigen Werte von k auf der x-Achse. Ein heuristisch guter k -Wert ist der Punkt, an dem das Diagramm einen Ellenbogen, also eine Knickstelle, bildet. Dabei berechnet sich der WCSS-Wert eines Datensatzes X durch:

$$WCSS(X) = \sum_{x \in X} \|x - c(x)\|^2$$

wobei $c(x)$ das Zentrum angibt, welchem der Datenpunkt x durch den k -Means-Algorithmus zugeordnet wurde (vgl. Schubert 2023). Die Ellenbogenmethode ist in der Fachliteratur um-

stritten, da ihre Ergebnisse je nach Datensatz weder eindeutig noch korrekt sind (vgl. Schubert 2023). Aufgrund ihrer Einfachheit und guten Aussagekraft für kleine klar clusterbare zweidimensionale Datensätze ist sie dennoch in ClusterLabor integriert.

Eine andere Methode zur Clusteranalyse ist die *Average Silhouette Method*. Analog zur Ellenbogenmethode wird zunächst der Datensatz für verschiedene k geclustert. Für jedes Ergebnis wird nun der Durchschnitt der Silhouettenkoeffizienten aller Datenpunkte des Datensatzes berechnet. Der Silhouettenkoeffizient $s(i)$ misst, wie gut ein Datenpunkt i zu seinem eigenen Cluster im Vergleich zu benachbarten Clustern passt. Es gilt:

$$s(i) = \begin{cases} 0, & \text{falls } a(i) = 0 \\ \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}, & \text{sonst} \end{cases}$$

wobei $a(i)$ der mittlere Abstand des Punktes i zu allen anderen Punkten desselben Clusters und $b(i)$ der mittlere Abstand zu den Punkten des nächstgelegenen Clusters ist. Es gilt $-1 \leq s(i) \leq 1$, wobei negative Werte darauf hinweisen, dass der Punkt i dem falschen Cluster zugeteilt wurde, und positive Werte dafürsprechen, dass der Punkt i richtig eingeteilt wurde. Für den Vergleich verschiedener Clusterzahlen k , kann man die Durchschnittswerte aller Silhouettenkoeffizienten (Average Silhouette Width, ASW) auf der y-Achse und die zugehörige Clusterzahl k auf der x-Achse in einem Diagramm auftragen, ergibt sich ein Maximum an der Stelle des heuristisch zu wählenden k (vgl. Ertel 2021).

Aufgrund der Ähnlichkeit des k -Means- und k -Medoids-Algorithmus können die Konzepte der Ellenbogenmethode und Average Silhouette Method direkt auf den k -Medoids-Algorithmus übertragen werden.

Das agglomerative hierarchische Clustering

Ein alternativer Algorithmus ist das agglomerative hierarchische Clustering. Beim agglomerativen hierarchischen Clustering fasst man zu Beginn jeden einzelnen Datenpunkt als ein eigenes Cluster auf. Danach werden die beiden Nachbarcluster mit der geringsten Distanz so lange zusammengeführt, bis alle Punkte in einem einzigen Cluster vereinigt sind (vgl. Ertel 2021). Um die Cluster mit minimalem Abstand zu bestimmen, gibt es bestimmte Metriken zum Messen der Distanz zwischen Clustern. Beim Single Linkage bestimmt sich die Distanz zwischen zwei Clustern A und B durch den minimalen Abstand zweier Punkte aus den Clustern, während beim Complete Linkage der maximale Abstand gewählt wird:

$$D_{\text{singleLinkage}}(A, B) = \max\{d(a, b) \mid a \in A, b \in B\}$$

$$D_{\text{completeLinkage}}(A, B) = \min\{d(a, b) \mid a \in A, b \in B\}$$

Beim Average Linkage wird der Durchschnitt aus den Abständen aller Elementpaare beider Cluster gebildet. Beim Centroid Linkage kehrt die Idee eines Clusterzentrums in Form des Schwerpunktes zurück. Die Entfernung zwischen zwei Clustern wird hierbei über die Distanz der beiden Clusterschwerpunkte definiert.

$$D_{\text{averageLinkage}}(A, B) = \frac{1}{|A| \cdot |B|} \sum_{a \in A, b \in B} d(a, b)$$

$$D_{\text{centroidLinkage}}(A, B) = d(\bar{a}, \bar{b})$$

mit \bar{a} und \bar{b} als Schwerpunkte der Cluster A und B (vgl. Miyamoto 2022). Im Vergleich zum k -Means-Algorithmus können mithilfe des agglomerativen hierarchischen Clusterings – besonders bei der Verwendung der Single Linkage Metrik – auch konkave Datensätze korrekt geclustert werden.

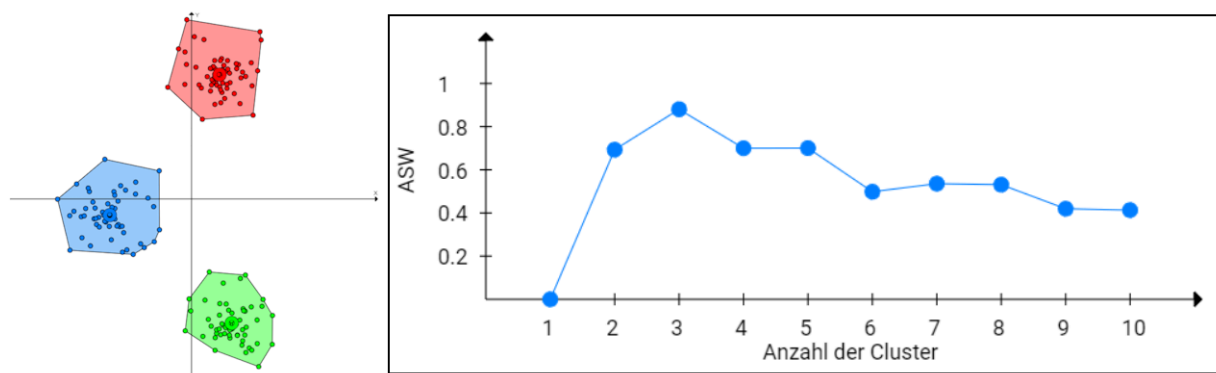


Abbildung 3: ASW Plot eines klar clusterbaren Datensatzes mit Maximum bei $k = 3$, Visualisierungen mit ClusterLabor

Bestimmen der "optimalen" Clusterzahl k mithilfe des agglomerativen hierarchischen Clusterings

Auch bei hierarchischem Clustering kann durch graphische Darstellungen das „optimale“ k ermittelt werden. Der Verlauf des hierarchischen Clusterings kann schematisch als Baum dargestellt werden. Beginnend bei den einzelnen Datenpunkten als Startcluster (die Blätter) werden in jedem Schritt die jeweils nächsten Cluster (Teilbäume) miteinander fusioniert, bis zum Schluss ein einziges Cluster (die Wurzel) erreicht wird, welches alle Datenpunkte des Datensatzes enthält. Trägt man diesen Baum mit dem Abstand der fusionierten Cluster als die y-Koordinate der Knoten in ein Diagramm ein, erhält man das sogenannte *Dendrogramm*. Nach Konstruktion gibt die Höhe der Verbindungslinien an, wie ähnlich die verbundenen Cluster sind. Je niedriger die Höhe der Verbindungslinie, desto ähnlicher sind die in diesem Schritt fusionierten Cluster. Abb. 4 veranschaulicht einen kleinen geclusterten Datensatz mit dem zugehörigen Dendrogramm.

Ein Dendrogramm zu interpretieren kann je nach Datensatz schwierig sein. Oft sucht man für den "optimalen" k -Wert eine Stelle, an der es einen großen Sprung in der Höhe der Verbindungslinien gibt. Schneidet man an dieser Stelle waagrecht durch das Dendrogramm ergibt sich durch die Anzahl der gekreuzten senkrechten Verbindungslinien die vom Dendrogramm prognostizierte "optimale" Anzahl k an Clustern (vgl. Miyamoto 2022).

Funktionsumfang

ClusterLabor führt den Nutzer in drei nummerierten Schritten – "Daten laden", "Daten normalisieren (optional)", "Algorithmus ausführen" – durch die Anwendung eines Clusteringalgorithmus. Zunächst müssen dazu Daten in das Programm geladen werden, welche daraufhin

normalisiert werden können. Zum Abschluss kann der gewünschte Algorithmus ausgewählt und auf die Daten angewandt werden. Hintergrundwissen zur genauen Funktionsweise der Algorithmen ist bei dem bloßen Clustern der Daten nicht erforderlich.

ClusterLabor ermöglicht verschiedene Möglichkeiten zum Laden eines Datensatzes. Es können Beispieldatensätze vom Server oder eigene Datensätze von der Festplatte geladen werden. Zudem bietet ClusterLabor an, einen klar clusterbaren Datensatz oder einen zufälligen Datensatz zu generieren. Schlussendlich ist es möglich mittels Mausklick einen Datensatz in einem interaktiven Editor in ClusterLabor selbst zu erstellen.

Ein geladener Datensatz kann in ebendiesem Editor auch nachträglich bearbeitet werden. So können einzelne Punkte hinzugefügt, verschoben oder gelöscht werden oder die Achsenbenennung angepasst werden. Diese Änderungen sind auch nach einem erfolgten Clustering noch möglich, wodurch das Verhalten der verschiedenen Algorithmen in Abhängigkeit von Ausreißern oder der Streuung der einzelnen Cluster analysiert werden kann.

Neben Datenpunkten ohne Sachzusammenhang (wie "zweiMonde"), welche zur Analyse der Stärken und Schwächen der einzelnen Clusteringalgorithmen dienen, stellt ClusterLabor auch zwei Datensätze mit Realitätsbezug zur Verfügung. Zum einen der bekannte Schwertlilien-Datensatz "iris", zum anderen der Datensatz "geraete", der die Bildschirmbreite und -höhe einiger (fiktiver) Geräte enthält, die auf eine Website zugegriffen haben. Es ergeben sich drei Cluster, die auf die verwendeten Geräte (Handy, Tablet, Laptop/Monitor) hinweisen.

Je nach Sachzusammenhang kann es nützlich oder auch notwendig sein, den Datensatz vor dem eigentlichen Clustern zu normalisieren. ClusterLabor bietet dies als Option an, es ist

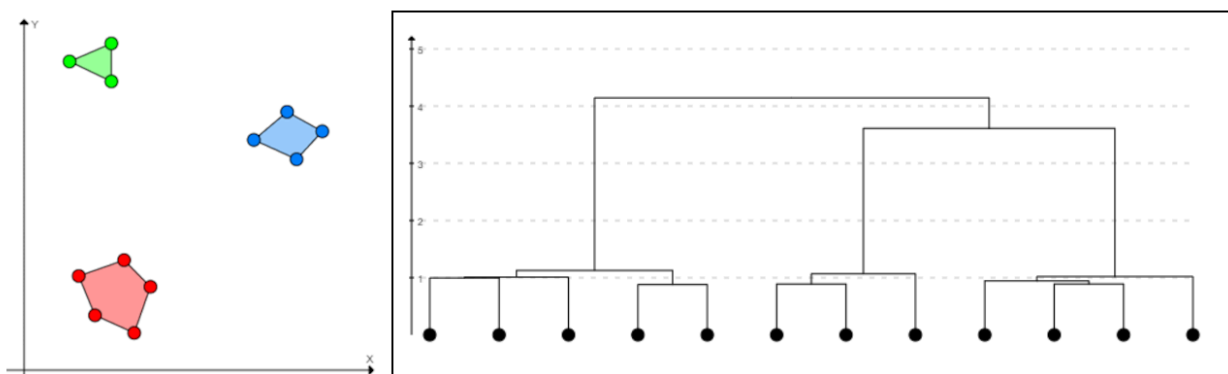


Abbildung 4: Dendrogramm eines kleinen Datensatzes, Visualisierungen mit ClusterLabor

Datei ▾
Anzeige ▾
Visualisierung des Clusterings ▾

1 Daten laden

Beispieldaten laden
zweiMonde ▾

Eigenen Datensatz hochladen

Klar clusterbaren Datensatz erstellen

Zufälligen Datensatz erstellen

Datensatz selbst erstellen

Datensatz bearbeiten

weiter

2 Daten normalisieren (optional)

3 Algorithmus ausführen

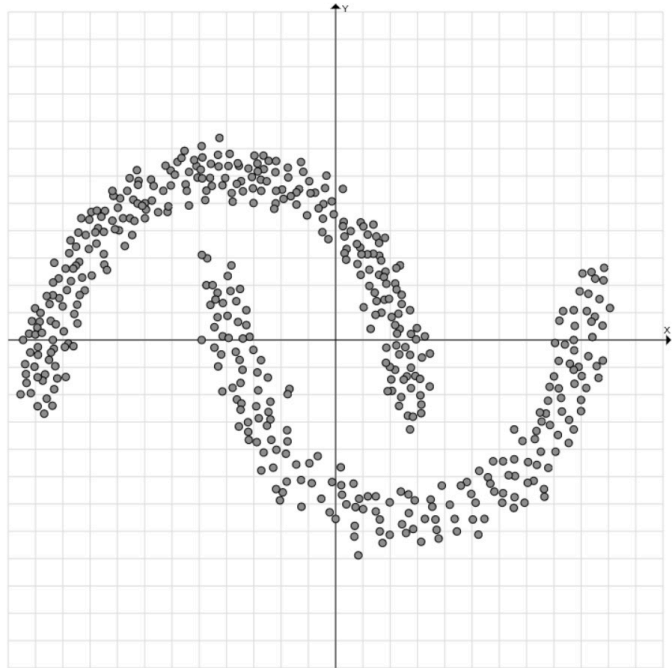


Abbildung 5: Die Ansicht in ClusterLabor, nachdem der Datensatz "zweiMonde" heruntergeladen wurde

aber auch möglich, den Datensatz direkt ohne Normalisierung zu clustern.

Zum Clustern stehen drei Algorithmen zur Auswahl: der k -Means-Algorithmus, der k -Medoids-Algorithmus und das agglomerative hierarchische Clustering. Der k -Means-Algorithmus ist mit den typischen Eigenschaften voreingestellt, sodass nur die Anzahl der Cluster eingegeben werden muss: Die Aktualisierung der Clusterzentren erfolgt durch Mittelwertbildung am Ende der Zuweisung (Lloyd), die Startzentren sind zufällig ausgewählte Punkte des Datensatzes und es wird das euklidische Abstandsmaß als Metrik verwendet. Jede dieser Parameter kann variiert werden. Es stehen verschiedene Möglichkeiten für Metrik, Startzentrenwahl, Aktualisierungsart und Aktualisierungszeitpunkt der Zentren zur Verfügung. Der k -Medoids-Algorithmus bietet weniger Möglichkeiten zur Variation. Hier kann lediglich die verwendete Metrik ausgewählt werden. Für das hierarchische Clustering kann ebenfalls die Metrik zum Messen der Distanz zwischen Datenpunkten variiert werden. Zusätzlich muss für den Algorithmus auch eine Distanz zwischen einzelnen Clustern ermittelt werden können. ClusterLabor bietet dafür die vier bekanntesten Linkage-Methoden – Single Linkage, Complete Linkage, Average Linkage und Centroid Linkage – an.

Nach dem Ausführen wird das Ergebnis des modifizierten Algorithmus in einem Koordinatensystem farbige dargestellt. Wurde der k -Means-

Algorithmus oder der k -Medoids-Algorithmus verwendet, werden die Clusterzentren als farbige Kreise dargestellt. Zusätzlich ist es möglich, die konvexe Hülle der Cluster oder das entsprechende Voronoi-Diagramm einzzeichnen zu lassen.

Neben der Möglichkeit, einen Datensatz für eine feste Clusterzahl k mittels eines der Algorithmen clustern zu lassen, ermöglicht ClusterLabor auch eine Clustering über einen Bereich hinweg, z. B. von $k=1$ bis $k=5$. Mittels eines Schiebereglers können die verschiedenen Clusterungen angezeigt und miteinander verglichen werden. Optional können bei einer Clustering über einen Bereich Graphen zur Clusteranalyse bereitgestellt werden. Bei dem k -Means- und dem k -Medoids-Algorithmus kann ein Elbow Plot oder ein ASW Plot angezeigt werden, bei hierarchischem Clustering ein Dendrogramm.

Implementierung

ClusterLabor ist als Webanwendung auf der Basis von Scala.js erstellt worden. Als Webanwendung genügt zur Nutzung von ClusterLabor ein üblicher Browser (wie Mozilla Firefox, Google Chrome oder Microsoft Edge). Die GUI benötigt eine Mindestauflösung von 1024x768 Pixeln und ist für die Bedienung mit Maus und Tastatur optimiert. Weitere Bedingungen müssen nicht erfüllt sein. Die Anwendung selbst befindet sich unter <https://www.ddi.informatik.uni-wuerzburg.de/cluster>

Datei ▾
Anzeige ▾
Visualisierung des Clusterings ▾

1 Daten laden
2 Daten normalisieren (optional)
3 Algorithmus ausführen

k-Means-Algorithmus

Wähle einen Wert oder einen Bereich für die Anzahl k der Cluster:
☒ Wert:
☐ Von: Bis:
Aktualisierung des Zentrums:
☒ Am Ende der Zuweisung (Lloyd)
☐ Schrittweise während der Zuweisung (MacQueen)
Wahl des Zentrums:
☒ Zufälliger Datenpunkt
☐ Zufälliger Punkt
☐ Setze Zentren per Klick in das Koordinatensystem
Wahl der Metrik:
☒ Euklidischer Abstand
☐ Manhattan-Distanz
Wahl der Zentrumsbildung:
☒ Mittelwert
☐ Median

Start

k-Medoids-Algorithmus

Hierarchisches Clustering

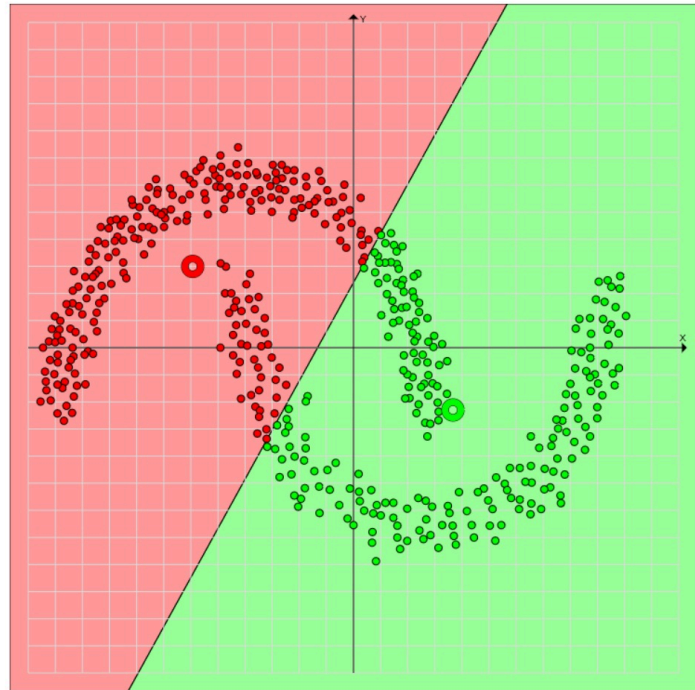


Abbildung 6: Entstandene Cluster bei Anwendung des k -Means-Algorithmus für $k=2$ mit den von ClusterLabor voreingestellten Eigenschaften und eingezeichnetem Voronoi-Diagramm

Ausblick

Das in diesem Beitrag vorgestellte Werkzeug ClusterLabor dient als didaktisches Werkzeug zum Vermitteln und Vergleichen von Clusteralgorithmen. Durch die interaktive Visualisierung und die verschiedenen Analysemöglichkeiten von Cluster-Ergebnissen ermöglicht ClusterLabor ein tieferes Verständnis der Funktionsweise verschiedener Algorithmen und der Auswirkungen unterschiedlicher Parameter auf die Ergebnisse dieser Algorithmen.

In naher Zukunft sollte die Effektivität und der Nutzen von ClusterLabor im Unterrichtsfeld empirisch evaluiert, Arbeitsaufträge formuliert und das Programm auf eine effiziente Anwendbarkeit hin optimiert werden. Zudem sind verschiedene Erweiterungen des Programms denkbar. Beispielsweise könnte es einen Modus geben, in dem das Clustering nicht automatisch komplett ausgeführt wird, sondern eine Animation der einzelnen Schritte bis zum Endergebnis erfolgt. So könnte z. B. die Funktionsweise des hierarchischen Clusterings oder der einzelnen Linkage-Methoden mit einem geführten Arbeitsauftrag von den Schülerinnen und Schülern selbst hergeleitet werden. Auch kleine Erweiterungen wie die Integration von anderen Clusteralgorithmen, die Verwendung zusätzlicher Metriken oder die Integration von weiteren Diagrammen und Methoden, um das "optimale" k zu bestimmen, sind denkbar und könnten im Unterricht produktiv eingesetzt werden.

Quellen

Alle Webseiten/Links wurden zuletzt geprüft am 30.05.2024.

Andres, D., Joachim, S., Hennecke, M.: Den k -Means-Algorithmus verstehen: Mit Stift & Papier und BlueJ. Informatische Bildung in Schulen Praxisbeiträge Jg. 2 (2024) (1), S.44-55. url: <https://doi.org/10.18420/ibis-02-01-06>

Demšar, J.; Curk, T.; Erjavec, A.; Gorup, Č.; Hočevár, T.; Milutinovič, M.; Možina, M.; Polajnar, M.; Toplak, M.; Starič, A.; Štajdohar, M.; Umek, L.; Žagar, L.; Žbontar, J.; Žitnik, M.; Zupan, B.: Orange: Data Mining Toolbox in Python. Journal of Machine Learning Research 14, S. 2349–2353, 2013, url: <http://jmlr.org/papers/v14/demsar13a.html>

Deutscher Bundestag: Drucksache 20/5149. Data-Mining – gesellschaftspolitische und rechtliche Herausforderungen, 2023, url: <https://dip.bundestag.de/vorgang/bericht-des-ausschusses-f%C3%BCr-bildung-forschung-und-technikfolgenabsch%C3%A4tzung-18-ausschuss/295126>

Ertel, Wolfgang (2021): Grundkurs Künstliche Intelligenz: Eine praxisorientierte Einführung, Springer, Wiesbaden. [5. Auflage]

Holmes, G.; Donkin, A.; Witten, I. H.: Weka: A machine learning workbench. In: Proceedings of the Second Australia and New Zealand Conference on Intelligent Information Systems. Australia, 1994, url: <https://ml>

cms.waikato.ac.nz/publications/1994/Holmes-ANZIIS-WEKA.pdf

Jin, X.; Han, J.: K-Medoids Clustering. In (Sammut, C.; Webb, G. I., Hrsg.): Encyclopedia of Machine Learning. Springer US, Boston, MA, S. 564–565, 2010, isbn: 978-0-387-30164-8, url: https://doi.org/10.1007/978-0-387-30164-8_426

Miyamoto, S.: Theory of Agglomerative Hierarchical Clustering. Springer Singapore, 2022, isbn: 978-981-19-0420-2.

Pöhner, N. (2023). Filterblasen verstehen. Informatische Bildung in Schulen 1(1). url: <https://doi.org/10.18420/ibis-01-01-06>

Schubert, E.: Stop using the elbow criterion for k-means and how to choose the number of clusters instead. ACM SIGKDD Explorations Newsletter 25 (1), S. 36–42, 2023, issn: 1931-0153, doi: 10.1145/3606274.3606278, url: <http://dx.doi.org/10.1145/3606274.3606278>

Staatsministerium für Schulqualität und Bildungsforschung München: LehrplanPLUS Gymnasium Bayern, Informatik 13 und spät beginnende Informatik 13 (grundlegendes Anforderungsniveau), url: <https://www.lehrplanplus.bayern.de/fachlehrplan/gymnasium/13/informatik/grundlegend>

Staatsministerium für Schulqualität und Bildungsforschung München: LehrplanPLUS Gymnasium Bayern, Informatik 13 (erhöhtes Anforderungsniveau), url: <https://www.lehrplanplus.bayern.de/fachlehrplan/gymnasium/13/informatik/erhoeht>

Lizenz



Dieser Artikel steht unter der Lizenz CC BY-NC-SA 4.0 zur Verfügung.

Materialien

Weitere Materialien zum Themenfeld Künstliche Intelligenz finden Sie unter <https://go.uni-wue.de/ki>

ClusterLabor
<https://www.ddi.informatik.uni-wuerzburg.de/cluster>

Kontakt

Daniela Andres
E-Mail: daniela.andres@uni-wuerzburg.de

Dr. Silvia Joachim
E-Mail: silvia.joachim@uni-wuerzburg.de

Prof. Dr. Martin Hennecke
E-Mail: martin.hennecke@uni-wuerzburg.de

Didaktik der Informatik, Julius-Maximilians-Universität Würzburg, <https://go.uni-wue.de/ddi>

Unterrichtsreihe zu künstlicher Intelligenz und künstlichen neuronalen Netzen für die gymnasiale Oberstufe

Grabe, K.

DOI: 10.18420/ibis-02-02-10

Zusammenfassung

Diese Unterrichtsreihe erarbeitet in 7 Doppelstunden das Thema „Künstliche Intelligenz und Maschinelles Lernen“ für einen Grundkurs Informatik in der gymnasialen Oberstufe nach dem „Bildungsplan Studienstufe Informatik Hamburg“. Die Rahmenbedingungen für den Ablauf der Sequenz sind, dass innerhalb der ersten 3 Doppelstunden theoretisches Basiswissen für eine Klausur aufgebaut wird, anschließend wird eine Klausur auf Papier geschrieben. Im zweiten, praktischen Teil (2 Doppelstunden) trainieren die SuS künstliche neuronale Netze (im Folgenden mit KNN abgekürzt) und arbeiten mit einer Python-Implementierung. In den letzten 2 Doppelstunden reflektieren sie die Möglichkeiten und Grenzen von Künstlicher Intelligenz (im Folgenden mit KI abgekürzt).

Einleitung

Einige der wichtigsten Zukunftstechnologien aktuell und Teil der Lebenswelt von SuS sind verschiedenste Anwendungen von KI wie ChatGPT, Recommender Systems z.B. bei Netflix

oder Amazon oder Bild-generierende KI wie DALL.E oder Midjourney. Um diese reflektiert nutzen zu können, benötigen die SuS einen Einblick in ihre Funktionsweise und eine Beschäftigung mit davon ausgehenden Chancen und Risiken.

Es gibt bereits unübersichtlich viele Beiträge und Materialien zum Unterrichten von KI im Schulunterricht im Web. Die Idee dieser Reihe ist es, verschiedene bereits existierende Materialien zu einer Sequenz zusammenzustellen, die für andere Lehrkräfte direkt nutzbar ist und Möglichkeiten zur Adaption an die eigene spezifische Lerngruppe aufzeigt (konkrete Optionen habe ich kursiv gekennzeichnet).

Beschreibung der Unterrichtsreihe

Teil 1: Basiswissen

Der erste Teil der Unterrichtsreihe baut innerhalb von 3 Doppelstunden Basiswissen zum Thema KI auf, das direkt danach (oder auch später) gut z.B. in Form einer Klausur abgefragt werden kann. In der Praxis habe ich für diesen Themenblock 4 Doppelstunden gebraucht, weil

DS	Thema	Fachbegriffe	Lernziel
1	Einstieg ins Thema: Klassifikation von Affen	Klassifikation Trainingsdaten Testdaten Entscheidungsbaum Accuracy	SuS erklären die Funktionsweise von Klassifikation mit Hilfe eines Entscheidungsbaumes.
2	Intelligenz vs. KI und die Geschichte von KI	Künstliche Intelligenz Maschinelles Lernen Deep Learning überwachtes Lernen unüberwachtes Lernen Verstärkungslernen	SuS erklären die Grundbegriffe des Themas KI und KNN.
3	KNN	Biologisches Neuron KNN Neuron Input/Output Gewicht Schicht Schwellenwert lineare / ReLU / sigmoide Aktivierungsfunktion	SuS erklären den Aufbau und die Bestandteile eines KNN. SuS berechnen Output von KNN anhand von Input und Gewichten.

Tabelle 1: Übersicht über den ersten Teil der Unterrichtsreihe

er einen großen Stoffumfang hat. Deshalb empfehle ich, die Geschwindigkeit an die eigene Lerngruppe anzupassen und den Stoff auf geeignet-viele Stunden zu verteilen.

1. Doppelstunde

Einstieg: Ich zeige den SuS verschiedene der Affen von AIUnplugged mit der Frage „Welcher dieser Affen beißt“.¹

Dies dient der Öffnung und zum neugierig-Machen, die SuS haben zu wenige Informationen, um die Frage entsprechend der Musterlösung beantworten zu können.

Erarbeitung 1: Anhand der Zuordnung der Affen zu den beiden Klassen „beißt“ und „beißt nicht“ skizzieren einen Entscheidungsbaum, mithilfe dessen entschieden werden soll, ob ein Affe beißt.

Erarbeitung 2: Die SuS bestimmen die Accuracy der Bäume.

Dazu tauschen die SuS die Bäume untereinander aus. Weitere neue Affenbilder werden gezeigt und die SuS sollen mit Hilfe des Entscheidungsbaumes eines anderen S entscheiden, ob der Affe beißt oder nicht. Der prozentuale Anteil der Affen, die von allen neu gezeigten Affen, richtig bestimmt wurden, ist die Accuracy des jeweiligen Entscheidungsbaumes.*

Sicherung: Der Begriff der Klassifikation mit Trainingsdaten und Testdaten wird eingeführt. Anschließend wird im Plenum über die verschiedenen Bäume und ihre Ergebnisse diskutiert.

Folgende Aspekte sollten thematisiert werden: Viele Klassifikationsmodelle (Entscheidungsbäume) haben die Äffchen mehrheitlich richtig klassifiziert. Kaum ein Modell hat alle richtig klassifiziert. Bei klassifizierenden KI ist es ähnlich. Besonders schwierig wird es, wenn ein neues Bild hinzugefügt wird, das ein Merkmal hat, das in den Testdaten nicht vorhanden war. In diesem Fall ist keine sinnvolle Zuordnung möglich und in Realität könnte es gefährlich sein, Beispiel: selbstfahrendes Auto bremst ruckartig vor Blättern im Herbst, weil es im Sommer trainiert wurde.

2. Doppelstunde

Einstieg: Bastele deine eigene KI“ (aus Pappbechern, Stiften und Zettelchen) nach Anleitung aus Erklärvideo.²

¹ <https://www.aiunplugged.org/german.pdf>, vgl. Lindner & Seegerer (o.D.), S. 4-7.

² <https://www.i-am.ai/de/build-your-own-ai.html>, vgl. Matt (o.D.).

Zwischensicherung: SuS tragen zusammen, welche Eigenschaften spezifisch für eine KI sind.

Elementar ist, dass die KI trainiert wird und dass sie Probleme löst.

Erarbeitung 1: SuS erarbeiten mithilfe der Arbeitsblätter von Lardelli (2020-2021)³ den Unterschied zwischen menschlicher Intelligenz und KI.

Wichtig ist, dass der psychologische Begriff Intelligenz keine eindeutige Definition hat und somit KI auch keine eindeutige Definition hat, sondern nur umschrieben werden kann.

Erarbeitung 2: SuS schauen Video zu geschichtlichen KI⁴ mit dem Arbeitsauftrag, die wichtigsten KI zusammenzutragen.

Das Video beinhaltet noch nicht die Entwicklungen von Deep Learning / KNN der letzten Jahre, dies sollte im Unterrichtsgespräch ergänzt werden. Wichtigste Exemplare von KI: 1950 Turing-Tests, 1966 Eliza, 1997 IBM Deep Blue, 2011 IBMs KI Watson, 2016 AlphaGo, 2022 ChatGPT (noch nicht im Video enthalten). Evtl. kann aktuelleres Video gefunden werden.

Erarbeitung 3: SuS erarbeiten mithilfe der Arbeitsblätter von Lardelli (2020-2021) die Fachbegriffe künstliche Intelligenz, maschinelles Lernen (ML, Teilmenge von KI), Deep Learning (DL, Teilmenge von ML) und die Typen des maschinellen Lernens (überwacht - supervised, unüberwacht - unsupervised, verstärkend - reinforcement)

Option: Erarbeitung 1, 2 und 3 könnten auch als Stationsarbeit durchgeführt werden.

Sicherung: Kahoot zu neuen Fachbegriffen und ihre Beziehung untereinander. (Vgl. Anhang Kahoot „Sicherung 2. Doppelstunde“).

3. Doppelstunde

Einstieg: Interaktives Spiel mit der Lerngruppe "Ein neuronales Netz aus Menschen".⁵ In einem Gruppenspiel stellen die SuS Neuronen in einem KNN dar. Es werden Inputs (Striche) gegeben, die Neuronen handeln entsprechend ihrer spezifischen Handlungsanweisung und am Ende muss der Output (eine der Ziffern von 1 bis 8) bestimmt werden.

³ Arbeitsblätter aus Lardelli (2020-2021), S. 6-9 & 15-18. Weiterführende Informationen unter (<https://ki-kit.ch/pages/autor.html>).

⁴ Video von <https://ki-campus.org/videos/geschichteki>, vgl. KI-Campus (o.D.).

⁵ Spiel aus https://www.science-on-stage.de/sites/default/files/material/anleitung_neuronales-netz-als-enaktives-modell.pdf, vgl. Janssen (2022a).

Information 1: SuS erarbeiten sich die Funktionsweise eines KNN mit „Mensch, Maschine!“ (2019) S.32-33.

Information 2: SuS erarbeiten sich die verschiedenen Aktivierungsfunktionen mit den Arbeitsblättern von Lardelli (2020-2021) „Künstliche Intelligenz und Robotik“ S. 32-36.

Für Information 1 und 2 können die Lösungen in Form von Lösungsblättern bereitgestellt werden. Alternativ können die Aufgaben im Plenum besprochen werden.

Erarbeitung: SuS berechnen den Output von KNN auf einem Arbeitsblatt. (Vgl. Anhang Arbeitsblatt „Berechnung KNN“).

Sicherung: Vergleich der Ergebnisse.

Wichtig ist, dass bei der Berechnung des Outputs Aktivierungsfunktionen und vorangegangene Outputs einbezogen werden.

Teil 2: Optimierung und Implementierung eines KNN

Die Vorfreude ist groß, endlich dürfen die SuS im zweiten Teil der Reihe KNN selbst aktiv optimieren und mit einer Python-Implementierung arbeiten.

Datensatz: Die Erarbeitung findet anhand des 1936 von Ronald Fisher bereitgestellten Iris-Datensatz statt. Dies ist ein bekannter Datensatz, an dem sich Klassifizierung gut demonstrieren lässt. Der Datensatz besteht aus 150 Daten (1 Datum = 1 Exemplar einer Blüte), je 50 zu einer der drei Schwertlilienarten Iris Setosa, Iris Versicolor und Iris Virginica. In Abbildung 1 sind jeweils ein Beispiel für jede der drei Iris-Typen zu sehen.



Abbildung 1 "Drei Beispiele für Iris Setosa, Iris Versicolor und Iris Virginica"⁶

⁶ Bildzusammenstellung durch Autorin, Bildquellen:

Im Datensatz gibt es allerdings zu jeder Blüte nur 5 Attribute aber kein Bild: Länge des Kelchblatts, Breite des Kelchblatts, Länge des Kronblatts, Breite des Kronblatts, Spezies. Ein Beispiel Datum ist: 5.1, 3.5, 1.4, 0.2, Iris-setosa⁷.

Mit diesem Datensatz wird die Optimierung von KNN im Setting Klassifizierung durch überwachtes Lernen erarbeitet.

4. Doppelstunde

Tool: Das von dem Lehrer Dr. Daniel Janssen entwickelte Online Maschine Learning Tool (OMLT) reduziert die komplexe Optimierung von KNN auf wenige Parameter (Netz-Aufbau, Lernrate, Epochen).⁸

Einstieg: Die SuS erarbeiten den Prozess der Klassifizierung selbst aktiv. Dazu bekommen die SuS im Plenum (ausgedruckte) Bilder des Iris-Datensatzes und sollen diese eine der 3 Arten zuordnen.

Lösbar wird die Aufgabe, indem 4 Attribute (nicht die Spezies) auf die Rückseite der Bilder geschrieben wird und indem im Vorhinein markante Exemplare ausgewählt wurden.

Information: Entdeckend lernend arbeiten die SuS sich selbstständig in das OMLT ein und nutzen dazu die 2. im Tool bereitgestellte Variante des Iris-Datensatzes.

Erarbeitung: SuS besprechen im Plenum, welche Parameterkombinationen sie zur Optimierung testen wollen. Arbeitsteilig testen die SuS die ihnen zugeteilten Parameter bei der Optimierung des KNN.

Sicherung: Die SuS präsentieren ihre Optimierung vor der Lerngruppe.

Wichtigste Ergebnisse sollten sein: Besonders geringe Fehlerraten gibt es bei einer hohen Epochenzahl (diese ist aber in der Realität teuer). Je größer die Lernrate ist, desto schneller aber

- iris setosa von Radomil via Wikimedia (CC BY-SA 3.0)
 - iris versicolor von Dlanglois via Wikimedia (CC BY-SA 3.0)
 - iris virginica von Frank Mayfield via Wikimedia (CC BY-SA 2.0)
⁷ Datensatz aus <https://archive.ics.uci.edu/dataset/53/iris>, vgl. UCI Machine Learning Repository (o.D.).
⁸ OMLT von <https://slxs.de/ml/>, vgl. Janssen (o.D.).

DS	Thema	Fachbegriffe	Lernziel
4	Iris-Datensatz in OMLT ausprobieren und optimieren	Iris-Datensatz	SuS optimieren eine KNN mit dem Online Maschine Learning Tool.
		Lernrate	
		Epoche	
5	Implementierung eines KNN für den Iris-Datensatz in Python	Validationsdaten	SuS optimieren ein KNN in Python.
		Googles TensorFlow	

Tabelle 2: Übersicht über den zweiten Teil der Unterrichtsreihe

auch ungenauer ist das Training und desto größer werden die Gewichts- und Schwellenwerte.

5. Doppelstunde

Tool: TensorFlow Playground; Python-Code einer Implementierung eines KNN (vgl. im Anhang 2) Programmcode „KNN für den Irisdatensatz mit Python“), empfohlene Bibliotheken: TensorFlow oder PyTorch, Keras, Sklearn⁹

Einstieg: SuS dürfen selbstständig das Tool TensorFlow Playground ausprobieren unter der Aufgabenstellung: „Manipulieren Sie die Optimierungsparameter und beschreiben Sie ihre Auswirkungen auf die Klassifizierung“.

Option: Lerngruppenspezifisches Auffrischen des Wissens zu Python.

Information: Input zu Validationsdaten als Validierung für Testdaten.

Erarbeitung: Die SuS implementieren ein KNN mit Python für den Iris-Datensatz.

In meiner Lerngruppe waren die Kompetenzen in Python größtenteils gering. Deshalb habe ich den SuS einen funktionierenden Code gegeben und als Aufgabe die Optimierung des KNN anhand der Parameter Netzaufbau, Epochen, Optimizer (Festlegung des Algorithmus zur Anpassung der Gewichte während des Trainings), Batch-Size (Größe des fürs Training übergebenen Datenpakets).

Sicherung: SuS tauschen sich in Kleingruppen zu Optimierungsmöglichkeiten aus.

Ähnlich wie in OMLT führen größere Netze und mehr Epochen zu genaueren Ergebnissen.

Teil 3: Möglichkeiten und Grenzen von KI

Die letzten zwei Doppelstunden leiten die SuS zur Reflexion des Themas KI an. Dazu wird in der 6. Doppelstunde die Optimierung von KNN aus technischer Sicht reflektiert und in der 7. KI gesamtgesellschaftlich aus verschiedenen Blickwinkeln.

6. Doppelstunde

⁹ <https://www.tensorflow.org/> , vgl. Google (o.D.).

DS	Thema	Fachbegriffe	Lernziel
6	Technische Grenzen der Optimierung von KNN	Validationsdaten	SuS reflektieren die Begrenztheit eines KNN anhand der Kriterien Lernrate, Overfitting und Aufbau des KNN.
		Lernrate	
		Overfitting	
		Ockham's Razor	
7	SuS reflektieren KI gesamtgesellschaftlich	Starke und schwache KI	SuS reflektieren KI im Hinblick auf die Themen Arbeitswelt, Ethik, Demokratie, Schule, Regulierung, Nachhaltigkeit & Umwelt.

Tabelle 3: Übersicht über den dritten Teil der Unterrichtsreihe

Einstieg: SuS sammeln bekannte KI-Anwendungen im Plenum am Board.

Erarbeitung: SuS arbeiten mit Ergebnissen aus dem Training der KNN-Python-Implementierung aus der 5. Doppelstunde, anhand dessen Grenzen der Parameter Lernrate, Netzaufbau und Overfitting ersichtlich sind. Die SuS arbeiten arbeitsteilig zu einem der 3 Themen der Optimierungsgrenzen jeweils an einem Arbeitsblatt in PA (vgl. im Anhang die Arbeitsblätter „Grenzen von Optimierung mit KI an Fisher's Iris-Dataset“).

Sicherung: SuS präsentieren die Ergebnisse.

Zentrale Ergebnisse sind: Lernrate darf nicht zu groß oder zu klein sein. Overfitting nennt man das Phänomen der Überanpassung auf den Trainingsdatensatz, sodass neue Daten nicht mehr klassifiziert werden können. Das KNN sollte optimaler Weise nicht zu klein, aber auch nicht zu groß sein, bei gleichen Ergebnissen sollte sich nach Ockham's Razor für den einfachsten Aufbau entschieden werden.

Reflexion: Gemeinsames Reflektieren (Plenum) über Grenzen der im Einstieg gesammelten KI.

7. Doppelstunde

Einstieg: SuS brainstormen zum Thema Möglichkeiten und Grenzen von KI.

Information und Erarbeitung: SuS erarbeiten arbeitsteilig jeweils einen der Artikel der bpb „Aus Politik und Zeitgeschichte“: KI und ... 1. Arbeitswelt 2. Ethik 3. Demokratie 4. Schule 5. Regulierung 6. Nachhaltigkeit und Umwelt; *Option: Input zu Regeln der KI Nutzung an dieser Schule.*¹⁰

Option: SuS könnten die Botschaften der Artikel (z.B. nach dem Lesen) von einer Sprach-generierenden KI (z.B. ChatGPT) zusammenfassen lassen.

Sicherung: SuS präsentieren die Hauptaussagen ihres Artikels.

Diese sind u.a.

¹⁰ <https://www.bpb.de/shop/zeitschriften/apuz/kuenstliche-intelligenz-2023/541496/grauzonen-zwischen-null-und-eins/>, vgl. bpb (2023).

1. *Arbeitswelt: aktuell KI als Assistenz für Jobs, Zahlen für Substitution von Berufen noch nicht kalkulierbar, KI lässt auch neue Berufe entstehen*
2. *Ethik: EU-Gremium HEG-KI fordert Respektierung der menschlichen Autonomie, Vermeidung von Schäden, Fairness, Erklärbarkeit, technischer Sicherheit, Datenschutz*
3. *Demokratie: Misinformation und Missbrauch in demokratischer Öffentlichkeit vs. Nutzen von KI-Entscheidungssystemen*
4. *Schule: viele Chancen für SuS (z.B. Intelligente Tutorsysteme, Unterstützung beim Lernen ...) und LKs z.B. Unterstützung in der Vorbereitung, aber auch Herausforderungen z.B. Adaption der Prüfungsformate notwendig*
5. *Regulierung: Regulierung ist komplex und schwierig; KI-VO-E (2021) erster umfassender Verordnungsentwurf der EU zur Regulierung von KI*
6. *Nachhaltigkeit und Umwelt: auch hier Chancen (z.B. intelligente Steuerung von Ressourcenverbrauch) und Risiken (z.B. Energieverbrauch)*

Reflexion: SuS reflektieren begründet über den Einsatz von KI in den Feldern und beziehen die Reflexion auch auf die eigene individuelle Situation und mögliche Zukunft.

Didaktische Reserve

Online gibt es eine Fülle an KI Anwendungen, die uns staunen lassen oder Spaß machen. Eine kleine Auswahl habe ich hier angeführt, einzelne Elemente können zwischendurch in den Unterricht eingeschoben werden. Während meiner Unterrichtsreihe, habe ich sie den SuS dauerhaft bereitgestellt und schnelle SuS durften, wenn sie mit ihren Aufgaben fertig waren, damit herumspielen. (KI entwickelt sich schnell weiter, vermutlich gibt es bald neuere KI und die Links funktionieren nicht mehr. Trotzdem diese Liste hier als Ausgangspunkt für Suche nach aktuellen KI.)¹¹

- Bilderquiz KI oder Foto: <https://www.geo.de/wissen/quiz/bilder-quiz--ki-bild-oder-fotografie--33364744.html>
- Finde den Schatz schneller als die KI: <https://www.i-am.ai/de/gradient-descent.html>
- Klavierspiel-KI: <https://www.i-am.ai/de/piano-genie.html>

- Quickdraw: <https://quickdraw.withgoogle.com/>
- This house does not exist: <https://thishousedoesnotexist.org>
- This person does not exist: <https://thispersondoesnotexist.com/>
- Videoproduktion mit Sora KI: <https://openai.com/index/sora/>
- Which face is real?: <https://www.whichfaceisreal.com/>
- 10 Spiele für ChatGPT: <https://mpost.io/de/10-best-games-to-play-with-chatgpt/>

Erfahrungen in der Erprobung:

Aus der Durchführung der Unterrichtseinheit ziehe ich verschiedene Unterrichtserfahrungen:

- Anspruch: es gibt viel zu lernen und das Thema ist komplex. Mit dieser Einheit kratzen wir nur an der Oberfläche des Themas KNN und haben uns nicht mit der darunterliegenden Mathematik befassen. Diese könnte ein weiterführender Kurs zumindest ansatzweise beinhalten.
- Lernerfolg: Die Klausur hatte einen Schnitt von fast genau 9 Notenpunkten, nur 3 von 22 SuS hatten 4 Notenpunkte oder schlechter. 9 SuS hatten 10 Punkte oder besser. Dieses Ergebnis entspricht meinem Gefühl beim Unterrichten, dass der Großteil die Kompetenzen dieser Unterrichtsreihe gewinnen konnte. Leider fehlt eine weitere konkrete Rückmeldung der SuS zu dieser Reihe in Form einer Befragung. Dies möchte ich beim zweiten Durchführen gern nachholen.
- Format der Sicherung: Neu Erlerntes sollte in einer iServ Texte Datei (kollaboratives Textdokument) von SuS festgehalten werden. In der Realität war dazu selten Zeit, sodass ich nach jeder Stunde die neuen Fachbegriffe / Erkenntnisse selbst dort eintrug. Für die Klausur war dieses Dokument für die SuS vorteilhaft, ansonsten wurde es wenig genutzt. Hier hätte ich mir entweder mehr Zeit für nehmen sollen oder ich hätte ein anderes Format wählen sollen z.B. dezentrale Sicherung durch die SuS.
- In der 6. Doppelstunde wird thematisiert, welche KI die SuS aktuell nutzen. Dies passte zum dritten Teil, dem Reflexionsblock. Um die subjektiven Konzepte der SuS zu aktivieren, hätte das Aufgreifen der individuell genutzten KI bereits in einer früheren Doppelstunde passieren können. Dann hätte ich

¹¹ Folgende URLs abgerufen am 26.05.2024.

mich im Verlauf der Unterrichtseinheit immer wieder darauf beziehen können.

- Spaßfaktor beim Unterrichten: insgesamt ist das Thema sehr angenehm zu unterrichten, weil das Thema die SuS sehr interessiert und motiviert und weil es so viele Materialien gibt, die ständig erweitert werden, sodass ich selbst in der Vorbereitung immer weiter lerne.

Fazit und Ausblick

Das Thema KI ist gesamtgesellschaftlich brisant und sollte im Unterricht einer gymnasialen Oberstufe in Deutschland 2024 nicht fehlen. Insbesondere das technische Verständnis der Funktionsweise von KI, aber auch die kritische Auseinandersetzung mit Chancen und Risiken ist sehr wichtig. Zudem macht der Unterricht zum Thema KI nicht nur den Lehrenden Spaß, sondern auch den SuS. Herausfordernd für die LK (und auch die SuS) ist es, sich technologisch auf dem neusten Stand zu halten. Vielleicht kann bei dieser Aufgabe auch eine Schulung für das Kollegium oder interessierte Akteure im Schulumfeld abfallen. Es gibt viele weitere hier nicht einbezogene gute pädagogische Materialien z.B. Teachable Machine von Google oder SoekiaGPT. Der (künstlichen) Intelligenz im Schulumfeld sind (aktuell) wenige Grenzen gesetzt.

Anhang

Der Anhang zum Artikel steht in der Onlineversion des Artikels zur Verfügung unter:
<https://www.informatischebildung.de/ibis/article/view/40>



Quellen

Bpb (2023): Aus Politik und Zeitgeschichte (Künstliche Intelligenz). URL: <https://www.bpb.de/shop/zeitschriften/apuz/kuenstliche-intelligenz-2023/541496/grauzonen-zwischen-null-und-eins/> (Stand: 26.05.2024).

Bundesministerium für Bildung und Forschung (2019): Mensch, Maschine! (Wer zeigt hier wem den Weg?). URL: <https://www.wissenschaftsjahr.de/2019/jugendaktion/> (Stand: 26.02.2024)

Chollet, Francois (2021): Deep Learning with Python (Second Edition). Manning: Shelter Island.

European Commission (2019): Building Trust in Human-Centric Artificial Intelligence. Brüssel.

European Commission (2021): Vorschlag für eine Verordnung des europäischen Parlaments und des Rates

zur Festlegung Harmonisierter Vorschriften für künstliche Intelligenz. Brüssel.

Freie und Hansestadt Hamburg - Behörde für Schule und Berufsbildung (2022): Bildungsplan Studienstufe (Informatik). Hamburg.

Falk, Kim (2019): Practical Recommender Systems. Manning: Shelter Island.

Google (o.D.): TensorFlow. URL: <https://www.tensorflow.org/> (Stand 26.02.2024).

Hubwieser, Peter (2001): Didaktik der Informatik (Grundlagen, Konzepte, Beispiel). Berlin Heidelberg: Springer.

Janssen, Daniel (2022a): Ein neuronales Netz aus Menschen. URL: https://www.science-on-stage.de/sites/default/files/material/anleitung_neuronales-netz-als-enaktives-modell.pdf (Stand 26.05.2024).

Janssen, Daniel (2022b): Machine Learning in der Schule (Eine praxisorientierte Einführung in künstliche neuronale Netze, Gesichtserkennung und Co). Rheine: Science on Stage Deutschland e.V.

Janssen, Daniel (o.D.): Online Machine Learning Tool. URL: <https://slxs.de/ml/index.html> (Stand: 26.02.2024).

KI-Campus (o.D.): Eine kurze Geschichte der KI. URL: <https://ki-campus.org/videos/geschichteki> (Stand 26.05.2024).

Lardelli, Marco (2020-2021): Künstliche Intelligenz und Robotik (Ein Lehr- und Bastelbuch für Jugendliche ab 13 Jahre).

Lindner, Annabel & Seegerer, Stefan (o.D.): AI Unplugged – Wir ziehen künstlicher Intelligenz den Stecker (Aktivitäten und Unterrichtsmaterial zu künstlicher Intelligenz ohne Strom). URL: <https://www.aiunplugged.org/german.pdf> (Stand: 26.5.2024).

Matt, Andreas (o.D.): Bastele deine eigene KI. URL: <https://www.i-am.ai/de/build-your-own-ai.html> (Stand: 26.5.2024).

Stein, Walter: Künstliche Intelligenz. Eine Einführung für den Schulunterricht mit Programmierbeispielen.

Wissenschaftsjahr 2019: Mensch, Maschine! Wer zeigt hier wem den Weg? 2019.

UCI Machine Learning Repository (o.D.): Iris. URL: <https://archive.ics.uci.edu/dataset/53/iris> (Stand: 26.02.2024).

Lizenz



Dieser Artikel steht unter der Lizenz CC BY 4.0 zur Verfügung.

Kontakt

Katrin Gisela Maria Grabe

E-Mail: katrin@grabe-online.de