GESELLSCHAFT
FÜR INFORMATIK

Gesellschaft für Informatik (Hrsg.)

SKILL 2020
Studierendenkonferenz Informatik

30. September/01. Oktober 2020

Gesellschaft für Informatik e.V. (GI)

Lecture Notes in Informatics (LNI) - Seminars
Series of the Gesellschaft für Informatik (GI)

Volume S 16

ISSN 1614-3213
ISBN 978-3-88579-750-0

Volume Editors
Gesellschaft für Informatik e.V.
Ahrstraße 45
53175 Bonn
E-Mail: bonn@gi.de
Redaktion: Michael Becker
E-Mail: michael.becker@uni-leipzig.de

Series Editorial Board
Heinrich C. Mayr, Alpen-Adria-Universität Klagenfurt, Austria
(Chairman, mayr@ifit.uni-klu.ac.at)
Torsten Brinda, Universität Duisburg-Essen, Germany
Dieter Fellner, Technische Universität Darmstadt, Germany
Ulrich Flegel, Infineon, Germany
Ulrich Frank, Universität Duisburg-Essen, Germany
Michael Goedicke, Universität Duisburg-Essen, Germany
Ralf Hofestädt, Universität Bielefeld, Germany
Wolfgang Karl, KIT Karlsruhe, Germany
Michael Koch, Universität der Bundeswehr München, Germany
Peter Sanders, Karlsruher Institut für Technologie (KIT), Germany
Andreas Thor, HFT Leipzig, Germany
Ingo Timm, Universität Trier, Germany
Karin Vosseberg, Hochschule Bremerhaven, Germany
Maria Wimmer, Universität Koblenz-Landau, Germany

Dissertations
Steffen Hölldobler, Technische Universität Dresden, Germany

Seminars & Thematics
Andreas Oberweis, Karlsruher Institut für Technologie (KIT), Germany

© Gesellschaft für Informatik, Bonn 2020
printed by Köllen Druck+Verlag GmbH, Bonn

This book is licensed under a Creative Commons BY-SA 4.0 licence.

Vorwort

Die Studierendenkonferenz Informatik (SKILL) ist eine jährliche Konferenz für Studentinnen und Studenten der Informatik sowie angrenzender Disziplinen aus ganz D-A-CH. Die Intention des SKILL-Konferenzformats ist es, sehr guten studentischen Arbeiten eine öffentliche Plattform zur Diskussion zu bieten. Die Studierenden können Erfahrungen zum wissenschaftlichen Publizieren sammeln und ihre Ergebnisse vor einem breit gefächertem Publikum vorstellen.

In diesem Jahr feiert die SKILL ihr zehnjähriges Jubiläum. Die aktuelle COVID-19-Pandemie wirkt sich auch auf die SKILL aus. In diesem Jahr findet die Konferenz ausschließlich online statt. Dadurch können mehr Teilnehmerinnen und Teilnehmer erreicht werden. Auf der anderen Seite fehlt der unmittelbare und persönliche Austausch, der sich nur sehr schwer mit digitalen Mitteln unterstützen lässt.

Die Erfahrungen der letzten zehn Jahre zeigen, dass ein Format wie die SKILL wichtig ist, um Studierende für eine wissenschaftliche Diskussion und in letzter Konsequenz für eine wissenschaftliche Arbeit zu begeistern. Die SKILL wird daher auch in Zukunft offen für alle Themen der Informatik bleiben und die gesamte Vielfalt unseres Faches widerspiegeln.

In diesem Jahr wurden insgesamt 34 Beiträge als Full oder Short Paper eingereicht und wissenschaftlich begutachtet. In diesem Band erscheinen 12 Beiträge, die auf zwei Konferenztagen durch die Studierenden präsentiert werden. Sie bilden aktuelle wissenschaftlichen Diskussionen ab und beschäftigen sich u.a. mit natürlicher Sprachverarbeitung und Künstlicher Intelligenz bis hin zu gesellschaftlichen Fragestellungen der Digitalisierung.

Die Mitglieder des Organisationskomitees der SKILL 2020 bedanken sich bei den Autorinnen und Autoren, ohne deren hochwertige Beiträge die Konferenz nicht möglich wäre. Wir freuen uns darüber hinaus, dass wir auch in diesem Jahr namhafte Gutachterinnen und Gutachter gewinnen konnten, die den Studierenden mit hilfreichen und ausführlichen Kommentaren zu ihren Arbeiten zur Seite standen.

Karlsruhe, 30. September 2020

Organisationskomitee der SKILL 2020

- Michael Becker, Universität Leipzig, Institut für Angewandte Informatik e.V.
- Judith Michael, Rheinisch-Westfälische Technische Hochschule Aachen
- Thomas Riechert, Hochschule für Technik, Wirtschaft und Kultur Leipzig
- Johannes Schmidt, Universität Leipzig
- Ludger Porada, Gesellschaft für Informatik e.V.

Gutachterinnen und Gutachter der SKILL 2020

- Michael Bader, Technische Universität München
- Axel Böttcher, Hochschule München
- Thomas Efer, Universität Leipzig
- Julia Friedrich, Institut für Angewandte Informatik e.V.
- Kurt Geihs, Universität Kassel
- Frank Grimm, FH Zwickau
- Erik Körner, Universität Leipzig
- Olaf Herden, Duale Hochschule Baden-Württemberg Stuttgart Campus Horb
- Otthein Herzog, Tongji University, Shanghai, und Universität Bremen
- Walter Hower, Hochschule Albstadt-Sigmaringen
- Felix Hüning, FH Aachen
- Paul Jähne, Fraunhofer-Institut für Zelltherapie und Immunologie
- Christine Jakobs, Technische Universität Chemnitz
- Friedbert Kaspar, Hochschule Furtwangen
- Amit Kirschenbaum, Universität Leipzig
- Stephan Klingner, Universität Leipzig
- Juliane Krämer, Technische Universität Darmstadt
- Christian Kücherer, Hochschule Reutlingen
- André Langer, Technische Universität Chemnitz
- Jan-Patrick Lehr, Technische Universität Darmstadt
- Ludger Martin, Hochschule RheinMain Wiesbaden
- Thomas Meier, HfT Leipzig
- Roy Meißner, Universität Leipzig
- Lars-Peter Meyer, Universität Leipzig
- Lisa Jasmin Moos, Hochschule für Wirtschaft und Gesellschaft Ludwigshafen
- Marcel Müller, Technische Universität Darmstadt
- Felix Neumeister, Karlsruhe Institut für Technologie
- Alexander Paar, Duale Hochschule Schleswig-Holstein
- Olaf Resch, Hochschule für Wirtschaft und Recht Berlin
- Petra Sauer, Beuth Hochschule für Technik Berlin
- Peter Schartner, Alpen-Adria-Universität Klagenfurt
- Thomas Schmid, Universität Leipzig
- Andreas Schmidt, Saarland Informatics Campus
- Andreas Schmietendorf, Hochschule für Wirtschaft und Recht Berlin
- Ingo Scholtes, Bergische Universität Wuppertal
- Michael Schöttner, Universität Düsseldorf
- Oliver Skroch, Hochschule Darmstadt
- Steffen Staab, Universität Stuttgart
- Patrick Struck, Technische Universität Darmstadt

Inhaltsverzeichnis

Digitalisierung im Alltag

Franziska Schulz
Systemische Betrachtung des therapeutischen Roboters Paro im Vergleich zu dem Haustierroboter AIBO 11

Frederike Möllers, Tim Oberthür
Nutzensteigernde Faktoren zur Optimierung von Mobile Payment für Senioren als hilfsbedürftige Anwendergruppe 23

Mehrwert der Digitalisierung

Karlson Hanke
Allgemeines Referenzmodell zur Digitalen Transformation im öffentlichen Sektor 37

Mihail Miller
Vergleichende Analyse der Komplexität musikalischer Dimensionen von Klavierwerken europäischer Komponisten anhand von MIDI-Files 49

Sicherheit

Andreas Hellenbrand
Asynchronous and Decentral Group Management in Messengers with Delegated Proof of Stake 63

Games

Hannes Dröse
Vergleich verschiedener Algorithmen für ein N-Spieler schachähnliches Brettspiel 77

Philip Wilson, Andrej Savinov, Annabella Kadavanich
Developing a game AI for Murus Gallicus 89

Text Mining

Anna Abad Sieper, Omar Amarkhel, Savina Diez, Dominic Petrak
Semantic Code Search with Neural Bag-of-Words and Graph Convolutional Networks 103

Felix J. M. Welter
Quantitative comparison of polarity lexicons in sentiment analysis tasks: Using a lexicon overlap score for similarity measurement between lexicons 117

Short Papers

Christa Strauß, Tina Amann
Klassische Migräne vs. gewöhnliche Migräne: Untersuchung basierend auf Altersstruktur und Geschlecht 131

Jan Krämer
Mensch-Maschinen-Schnittstelle mit Vibrationsaktoren zum „Fühlen“ von Texten 143

Johannes Pavel, Christina Fercher, Frank Herold
Der Einsatz maschinellen Lernens zur inertialen Bewegungsanalyse am Pferderumpf für das digitale Trainingsmonitoring im Leistungssport . . . 149

Autorenverzeichnis

Digitalisierung im Alltag

Systemische Betrachtung des therapeutischen Roboters Paro im Vergleich zu dem Haustierroboter AIBO

Franziska Schulz[1]

Abstract: Roboter sind in der heutigen Zeit nicht nur in der Industrie zu finden, sondern werden immer häufiger in privaten Lebensbereichen eingesetzt. Ein Beispiel hierfür ist der soziale Therapie-Roboter Paro. Dieser ist dem Verhalten und Aussehen einer jungen Robbe nachempfunden, drückt Gefühle aus und wird besonders in Pflegeheimen eingesetzt. Dabei zeigt er positive Auswirkungen auf das Wohlbefinden pflegebedürftiger Menschen. Diese Arbeit stellt den Roboter Paro in einer systemischen Analyse dar: hierbei werden Systemkontext, Anwendungsfälle, Anforderungen und Struktur betrachtet. Anschließend erfolgt eine Analyse des Haustierroboters AIBO, welcher einem Welpen ähnelt und verstärkt der Unterhaltung von Privatpersonen dient. Es werden Gemeinsamkeiten und Unterschiede zwischen den Systemen herausgearbeitet. Dabei wird ersichtlich, dass beide Systeme dem Nutzer vorrangig Gesellschaft leisten, jedoch verschiedene Anforderungen besitzen und in unterschiedlichen Anwendungsdomänen eingesetzt werden. Zudem besitzt AIBO vielfältigere Fähigkeiten und einen höheren Bewegungsgrad als Paro. Dies spiegelt sich in einer komplexeren Struktur der Hardware wider.

Keywords: Intelligent robots, human-robot interaction, robot sensing systems, mental commit robot, pet-type robot

1 Einleitung

Aufgrund des demographischen Wandels sowie den verbesserten medizinischen Möglichkeiten und der damit ansteigenden Lebenserwartung sind immer mehr Menschen im höheren Lebensalter auf Pflege angewiesen. In diesem Lebensabschnitt entscheiden sich Menschen häufig bewusst für die Zurückgezogenheit aus der Gesellschaft oder werden von dieser isoliert. Zudem ändert sich sowohl die geistige, als auch körperliche Verfassung [Ma16]. Häufig werden in Pflegeeinrichtungen tiergestützte Therapieformen angewendet, um diese Problematiken zu adressieren. Tiertherapie hat positive Auswirkungen auf das Wohlbefinden älterer Menschen, führt zu Entspannung sowie verbessertem Kommunikationsverhalten [WS07]. Der Einsatz solcher Therapien wird jedoch oftmals erschwert. Gründe hierfür sind mögliche Allergien oder die Gefahr, dass das Tier dem Menschen Schaden zufügt [WS07]. Eine Alternative zur tiergestützten Therapie ist der Einsatz sozialer Robotik. Dabei sind soziale Roboter stark auf die Kommunikation mit den Nutzern ausgelegt. Einige dieser Roboter zeigen künstliche Emotionen und sind in der Lage, eine Beziehung mit dem Nutzer einzugehen, wie sie zwischen Mensch und Haustier zu finden ist [SWT02]. Ein Beispiel ist der therapeutische

[1] Hochschule Reutlingen, Informatik, Alteburgstraße 150, 72762 Reutlingen,
franziska_katharina.schulz@student.reutlingen-university.de

cc by sa

Roboter Paro, der seit dem Jahr 2004 kommerziell verfügbar ist und dem Aussehen einer jungen Sattelrobbe nachempfunden ist. Seit einigen Jahren wird Paro vermehrt in stationären Einrichtungen für ältere Menschen, in denen keine Tiere erlaubt sind, eingesetzt oder ergänzend zur Tiertherapie angeboten [SWT02].
Ziel der vorliegenden Arbeit ist es, den therapeutische Roboter Paro systemisch zu untersuchen. Des Weiteren wird er mit dem modernen Haustierroboter AIBO verglichen, welcher seit 2017 erhältlich ist und sich vorrangig an Privatpersonen richtet. Dabei soll aufgezeigt werden, wie sich soziale, tierähnliche Roboter im Laufe der letzten Jahre entwickelt haben. Des Weiteren wird betrachtet, wie sich der Einsatzbereich und Kontext eines sozialen Roboters auf dessen Anwendungsfälle, Anforderungen und Hardwarekomponenten auswirkt. Darin unterscheidet sich diese Arbeit von vorangegangenen Arbeiten von Shibata et al. [Sh04, SWT02, SYY97], welche vor allem die Entwicklung des Roboters Paro von den Prototypen zum fertigen Produkt sowie dessen positive Wirkung auf Pflegebedürftige beschreiben.

1.1 Der systemische Vergleich

Die beiden tierähnlichen Roboter werden mithilfe der Modellierungssprache Systems Modeling Language (SysML) dargestellt. Diese basiert auf der Unified Modeling Language 2 (UML2), welche so erweitert wurde, dass die Sprache nicht nur für Software Engineering, sondern für das Systems Engineering geeignet ist [WS14]. Die Analyse selbst wird dabei an den Prozess der SYSMOD angelehnt. Bei diesen handelt es sich um ein Vorgehen, welches die Modellierung der Architektur komplexer Systeme unterstützt. Im Rahmen des SYSMOD Prozesses werden die Idee und Ziele eines Systems, sowie dessen Basisarchitektur und Systemkontext analysiert. Zudem werden Anwendungsfälle betrachtet [WS14]. Kriterien zum Vergleich der beiden Roboter liegen demnach in deren Zielen, Einsatzbereich, Anwendungsfällen und Hardwarekomponenten. Zusätzlich werden Anforderungen an die beiden Systeme betrachtet.

Die Arbeit stellt dazu zuerst den therapeutischen Roboter Paro vor. Dieser wird unter den oben genannten Gesichtspunkten analysiert. Danach wird der Haustierroboter AIBO beschrieben und unter denselben Punkten analysiert. Im Rahmen einer Diskussion werden die Unterschiede zwischen den beiden Systemen sowie Ähnlichkeiten herausgearbeitet. Zuletzt wird ein Fazit über die gesamte Arbeit gezogen.

2 Systemische Analyse des therapeutischen Roboters Paro

Wird die Entwicklung des therapeutischen Roboters Paro betrachtet, wird deutlich, dass das ursprüngliche Ziel darin lag, ein künstliches Wesen zu entwickeln, welches nicht über eine Schnittstelle, sondern direkt mit dem Menschen interagiert. Eine zentrale Eigenschaft des künstlichen Wesens sollte darin liegen, dass es Charakteristiken und Verhaltensweise eines lebendigen Tieres aufweist [SYY97]. Die kommerziell verfügbare Version des Roboter Paro bewegt die Flossen und den Kopf, imitiert die Geräusche einer Robbe und

drückt auf diese Weise Gefühle aus. Außerdem erkennt er seinen eigenen Namen und lokalisiert die Richtung einer Stimme. Zuletzt ist es möglich, dass Paro das Verhalten erlernt, welches von dem Nutzer bevorzugt wird. Dabei unterscheidet der Roboter zwischen Zuneigung und groben Behandeln und erinnert sich an das Verhalten, welches er zuvor zeigte. Als Konsequenz wird er das jeweilige Verhalten verstärkt oder nicht mehr zu Tage legen [Pa14]. Aktuelle Ziele beim Einsatz des Systems liegen darin, das Wohlbefinden pflegebedürftiger Menschen zu erhöhen. Außerdem hat der Einsatz positive Auswirkungen auf das Kommunikationsverhalten der Pflegebedürftigen [Pa14].

2.1 Kontext des Systems

Die Analyse des Wechselspiels zwischen System und Umwelt ist ein essentieller Faktor zum Verständnis des Systems und wird in Abbildung 1 dargestellt. Von besonderem Interesse sind Schnittstellen, welche die Grenzen zwischen System und Umwelt festlegen. Diese Grenze wird von dem physischen Körper des Roboters dargestellt. Der Grund hierfür ist, dass Paro keinen aktiven Austausch von Daten mit seiner Umwelt unterstützt und sich nicht mit dem Internet verbindet [Pa15].

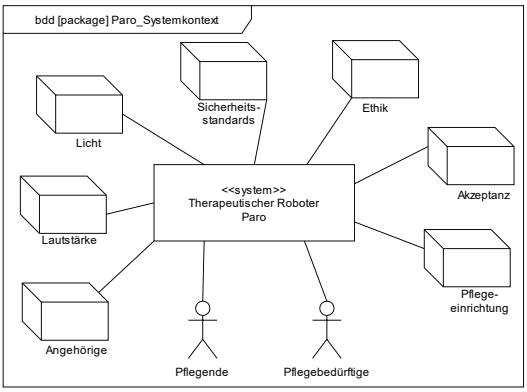


Abb. 1: Systemkontext des Roboters Paro

Unter Systemakteuren werden direkte Interaktionspartner verstanden. Diese werden zum einen von den *Nutzern* selbst repräsentiert. Nutzer des therapeutischen Roboters sind häufig ältere Menschen, können jedoch auch weitere Gruppen Pflegebedürftiger sein. Ein weiterer Systemakteur sind die *Pflegenden*, welche von professionellen Pflegern oder Privatpersonen repräsentiert werden, da der Roboter neben Pflegeeinrichtungen auch im privaten Bereich eingesetzt werden kann. In der vorliegenden Arbeit liegt der Fokus auf dem Einsatz des Roboters in Pflegeheimen, weshalb die *Pflegeeinrichtung* im Systemkontext aufgeführt ist. Auch spielen die Einstellung und Toleranz der *Angehörigen* der pflegebedürftigen Menschen eine große Rolle. Umwelteinflüsse, auf die der Roboter reagiert und die eine mögliche Störung des Systems hervorrufen, sind wechselnde

Lichtverhältnisse. Diese können das Sehvermögen des Systems beeinflussen sowie bei direktem Einfluss den Roboter überhitzen. Zudem beeinflusst die *Lärmkulisse* um den Roboter herum dessen Fähigkeiten im Richtungshören und der Erkennung seines Namens [Pa15]. Außerdem wird das System von geltenden *Sicherheitsstandards* beeinflusst. So definieren Normen mögliche Gefahren und Risiken, die von Robotern in der Pflege ausgehen und geben Empfehlungen ab. Des Weiteren spielen *ethische Aspekte* wie der Ersatz eines Lebewesens durch einen Roboter sowie mögliche Täuschungen der Pflegebedürftigen über die Lebendigkeit des Roboters eine Rolle [Ca11]. Zuletzt muss der Roboter von den Pflegebedürftigen sowie dem Personal akzeptiert werden.

2.2 Anwendungsfälle

Anwendungsfälle (engl. Use Cases) beschreiben mögliche Aufgaben und Ziele, die ein Nutzer verfolgt, wenn er mit einem System interagiert. Aktive Akteure lösen in der Regel den Anwendungsfall selbst aus und interagieren direkt mit dem System [WS14]. Als Akteure sind auch hier die *Pflegenden* und *Pflegebedürftigen* zu nennen.
Die Pflegebedürftigen interagieren mit dem Roboter ausschließlich auf eine soziale Art und Weise. Ein Use Case liegt darin, dem Roboter einen *Namen zu gebe* [Pa15]. Ein weiterer Use Case der Pflegebedürftigen liegt darin, eine *Beziehung zu Paro aufzubauen*. Diese wird durch die regelmäßige, intensive Interaktion zwischen Mensch und Roboter aufgebaut [Wa05]. Ein anderer Use Case seitens der Pflegebedürftigen ist das *Umsorgen* des Roboters, welcher durch Streicheln und dem Schenken von Zuneigung erweitert werden kann. Der Pflegebedürftige erhält dabei das Gefühl von dem Roboter gebraucht zu werden.
Werden die Anwendungsfälle der Pflegenden betrachtet, liegen diese nicht in der sozialen Interaktion mit dem Roboter. Vor allem sorgen sie dafür, dass Paro funktionsfähig bleibt, indem sie sich um das *Aufladen der Batterien* kümmern. Außerdem *reinigen* sie regelmäßig das künstliche Fell des Roboters.

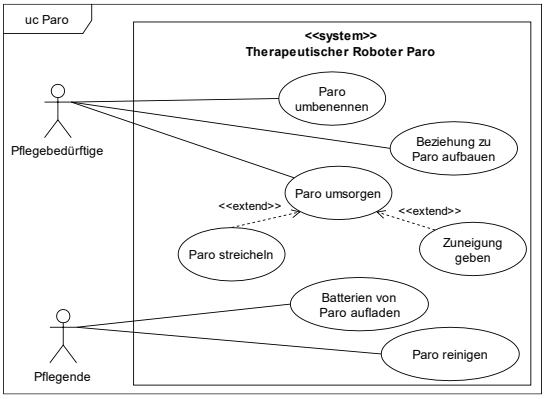


Abb. 2: Use Cases des Roboters Paro

2.3 Systemanforderungen

Die vorgestellten Anforderungen spielen besonders für die Tauglichkeit beim Einsatz des Roboters in der Altenpflege sowie dessen Wahrnehmbarkeit als lebendiges Wesen eine entscheidende Rolle. Eine der wichtigsten nicht-funktionalen Anforderungen ist die *Sicherheit*: Paro darf zu keiner Zeit eine Gefahr für den Nutzer darstellen. Ein Teil dieser

Anforderung stellt die Anforderung nach *Hygiene* dar: häufig wird Paro in Pflegeheimen verwendet, daher muss das künstliche Fell des Roboters antibakteriell und einfach zu säubern sein [Pa15]. Das *Aussehen* und *Verhalten* des Roboters müssen stimmig miteinander und kompatibel zu dem der Sattelrobbe sein. Zudem muss sich der Roboter verhalten, als sei er lebendig und habe einen Charakter [WS07]. Eine funktionale Anforderung liegt darin, dass Paro die *Stimme* der Person, die am meisten mit ihm interagiert, *erkennen* muss. Daraus ergibt sich eine nicht-funktionale Anforderung: der Roboter muss *lernfähig* sein. Des Weiteren ist die *Reaktionszeit* eine entscheidende Anforderung der Performance des Roboters. Paro muss in Echtzeit auf Umweltreize reagieren. Besonders bedeutend ist dies bei der Interaktion mit dem Nutzer, da dieser keine Verzögerung in den Reaktionen negativ wahrnehmen soll. Zuletzt ist die *intuitive Interaktion* mit dem Roboter eine weitere Anforderung. Dabei soll die Interaktion mit Paro möglichst einfach erfolgen.

2.4 Struktur des Systems

Im Folgenden wird die Struktur des Systems anhand der verwendeten Hardware illustriert. In Abbildung 3 wird die Hardware von Paro dargestellt. Diese ist in den Arbeiten von Shibata et al. [WS07], [SYY97] weniger detailliert erläutert, daher basiert die Darstellung zusätzlich auf der Von-Neumann-Architektur. Zentrum der Hardware sind die beiden 32Bit RISC-*Prozessoren* [WS07], [SYY97]. Nach der Von-Neumann-Architektur erfolgt die Kommunikation zwischen den Prozessoren und den anderen Komponenten vorrangig über Bussysteme. Auf diese Weise erfolgt auch der Austausch mit dem *Speicher*. Zudem senden die Prozessoren einseitig Daten an den *Lautsprecher,* welcher lediglich als Ausgabemedium dient. Des Weiteren senden die Prozessoren Befehle an die sieben *Aktuatoren*, die ebenfalls der Ausgabe dienen und für Paros Bewegungen zuständig sind. Diese befinden sich jeweils in den Augenlidern, dem Kopf sowie den vorderen und hinteren Flossen des Roboters [WS07]. Zusätzlich stehen die Prozessoren im wechselseitigen Austausch mit den *internen Sensoren*. Diese werden von taktilen Sensoren, Licht-, Temperatur- und Haltungssensoren sowie drei Mikrofonen repräsentiert [WS07]. Flexible, taktile Sensorik, durch die der Roboter die Intensität von Druck sowie dessen Position auf einer Oberfläche wahrnehmen kann, ist unterhalb des Fells [Sh04] an Kopf, Kinn, Torso und Flossen angebracht. Weitere taktile Sensoren sind an den Schnurrharren zu finden [Pa15].

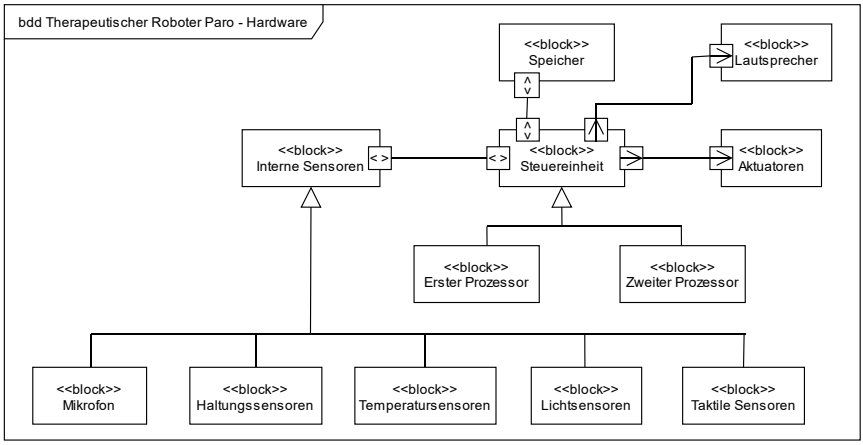


Abb. 3: Blockdefinitionsdiagramm der Hardware des Roboters Paro

3 Systemische Analyse des Haustierroboters AIBO

Im Jahr 1999 veröffentlichte Sony die erste Version von AIBO, einem Spielzeugroboter der Ähnlichkeiten zu einem Hundewelpen aufweist. Bis 2006 entstanden drei Generationen des Roboters, die erfolgreich verkauft wurden. 2018 wurde die neueste Version von AIBO, die in der vorliegenden Arbeit betrachtet wird, vorgestellt [KW18]. Ziel dieses Roboters ist es dem Nutzer Gesellschaft zu leisten und ihn zu unterhalten. Zudem bietet er eine futuristische Alternative zu einem Haustier [So20]. AIBO ist ein intelligenter Roboter, der in der Lage ist zu lernen. Die Persönlichkeit des Haustierroboters wandelt sich durch Interaktion mit dem Nutzer und den wahrgenommen Umweltreizen. Der Nutzer kann mit dem künstlichen Haustier spielen, es füttern und ihm verschiedene Kommandos beibringen. Während der Kommunikation reagiert AIBO auf visuelle und auditive Reize sowie auf Berührungen. Zuletzt zeigt AIBO eigene Wünsche und Emotionen, welche sein Verhalten beeinflussen [So18].

3.1 Kontext des Systems

Die Systemgrenze umschließt nicht mehr nur den physischen Roboter, da dieser eine Internetverbindung über WLAN oder LTE herstellen kann [So18]. Darüber nimmt er Verbindung zu einer *Cloud* auf, durch die verschiedene Daten wie Fotos sowie AIBOs erlernte Tricks über einen Server gespeichert werden. Nur durch die Verwendung dieser Cloud ist AIBO in der Lage zu lernen. Über eine dazugehörige *App* kann der Nutzer mit dem Roboter kommunizieren, Personalisierungen vornehmen und die Tricks verwalten [So18]. Damit umfasst das System neben dem Roboter selbst, auch eine Cloud sowie eine App. Dies wird in Abbildung 4 dargestellt.

AIBO richtet sich vorrangig an Privatpersonen, als Systemakteur kann daher lediglich der *Nutzer* genannt werden. Dieser interagiert aktiv mit dem System und formt die Persönlichkeit des Roboters. AIBO ist zudem an geltende *Sicherheitsstandards* angepasst [So20]. Im täglichen Gebrauch reagiert der Roboter auf seine *räumliche Umgebung*: er weicht Hindernissen aus und findet den kürzesten Weg zu seinem Ziel [So20]. Zudem reagiert er auf zugehörige *Accessoires* wie Spielzeug. Umwelteinflüsse, auf die der Roboter reagiert und die eine mögliche Störung des Systems hervorrufen, sind wechselnde *Lichtverhältnisse* sowie starke Lichteinflüsse. Diese können sich auf die Leistung der Sensoren auswirken [So18].

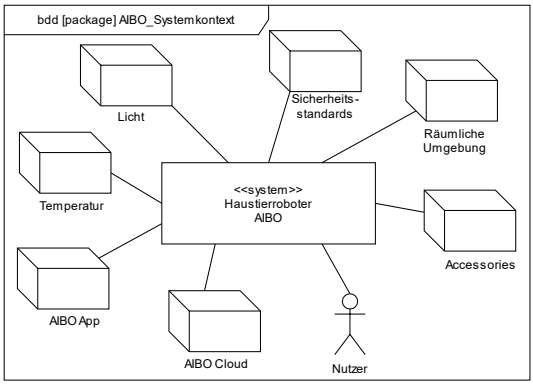


Abb. 4: Systemkontext des Haustierroboters AIBO

3.2 Anwendungsfälle

AIBO bietet eine Vielzahl an Anwendungsfällen, welche stetig durch Softwareupdates erweitert werden. Use Cases die sich auf die Unterhaltungsfunktion beziehen, werden in Abbildung 5 illustriert. Da AIBO primär ein Spielzeugroboter ist, liegt ein Use Case darin mit *ihm zu* spielen [So18]. Ein weiterer Use Case ist es *AIBO zu erziehen*. Dabei versucht der Nutzer dem Roboter bestimmte Verhaltensweisen abzutrainieren und stattdessen andere zu fördern sowie die Entwicklung der Persönlichkeit voranzutreiben [So18]. Dieser Use Case kann dadurch erweitert werden, dass AIBO neue *Tricks antrainiert* werden [So18]. Ein weiterer Use Case liegt in der *Personalisierung* des Roboterhunds. Dieser Use Case kann optional erweitert werden, indem der *Name des Roboters geändert* wird. Zuletzt ist es möglich, dass das gesamte *Bewusstsein* und damit alle Erinnerungen und Gelernte des Roboters in die *Cloud gespeichert* werden kann. Dieses bietet eine Backup-Funktion, die besonders dann sinnvoll ist, wenn der physische Roboterkörper Schaden nimmt und die Persönlichkeit AIBOs in einen neuen Körper geladen werden kann [KW18].

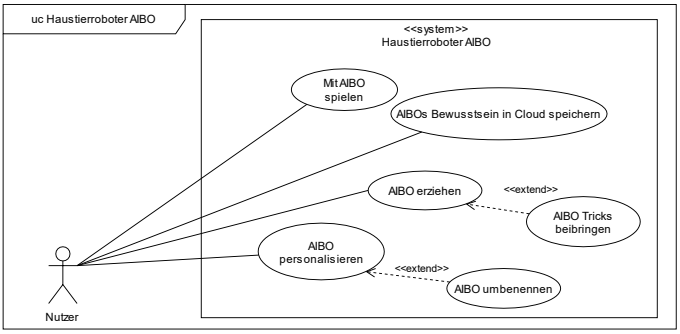


Abb. 5: Anwendungsfalldiagramm des Haustierroboters AIBO

3.3 Anforderungen an das System

In diesem Abschnitt werden speziell die Anforderungen an den Roboter betrachtet, die für dessen Unterhaltungsfunktion essentiell sind. Eine Anforderung, die stets gewährleistet werden muss, ist die *Sicherheit* des Menschen [So20]. Wird die Anforderung an das *Aussehen* des Roboters betrachtet, fällt auf, dass dieser nicht wie ein lebendiger Hund aussehen soll, sondern wie ein futuristisches Haustier. Die Merkmale eines Roboters sollen weiterhin deutlich sein [So18]. Weitere Designanforderungen sind die *Bewegungen* des Roboters. Diese sollen zum einen flüssig ablaufen, zum anderen muss AIBO in der Lage sein, eine Vielzahl verschiedener Bewegungen auszuführen. Das *Verhalten* des Roboters ist auch an das eines Hundes lediglich angelehnt. Eine weitere Anforderung ist, den künstlichen *Emotionen Ausdruck verliehen zu können*. Dies geschieht durch Bewegungen sowie den wechselnden Ausdruck in den Augen. Eine weitere Anforderung an das System, ist AIBOs *Lernfähigkeit*. Der Roboter muss sein Verhalten an den Nutzer anpassen sowie neue Tricks erlernen. Eine funktionale Anforderung, die mit der Lernfähigkeit zusammenhängt, ist das Unterscheiden und *Erkennen verschiedener* Personen [So18], [So20]. Zuletzt muss er *Objekte und Bewegungen erkennen*. Dies ist notwendig, damit sich AIBO ohne Kollisionsgefahr frei bewegen kann [So18]. Eine Anforderung, welche die Leistung des Systems betrifft, ist die *Reaktionszeit* des Roboters. Diese muss in Echtzeit erfolgen, damit der Nutzer eine angenehme Erfahrung mit dem Roboter hat und dessen Verhalten glaubwürdig erscheint.

3.4 Struktur des Systems

Das Blockdefinitionsdiagramm der Hardware wird in Abbildung 6 dargestellt. Zentrum der Hardware ist der 64Bit Vierkern-Prozessor [So20]. Dieser kommuniziert mit den *internen Sensor*en [Qu20], bei denen es sich um sieben verschiedene Typen handelt. Durch die *Bewegungs-* und *Entfernungssensoren* bewegt sich AIBO in einem Raum fort, ohne mit Menschen oder Hindernissen zu kollidieren. Der *Simultaneous Localization and Mapping (SLAM) Sensor* ermöglicht dem Roboter Orientierung, indem er Karten seiner

Umgebung erstellt und darin seine ungefähre Position schätzt. Zudem unterstützt der *Time off light (TOF) Sensor* den Roboter darin, Hindernissen auszuweichen und seine Umgebung schneller zu überblicken. Mithilfe des *Lichtsensors* können Helligkeitsunterschiede erkannt werden. Durch das eingebaute *Mikrofon* kann der Roboter Audiosignale wahrnehmen. Durch die *taktilen Sensoren* registriert der Roboter Berührungen. Diese Sensoren befinden sich in AIBOs Rücken, Kopf und Kiefer [So20]. Des Weiteren kommuniziert der Hauptprozessor mit einem Bewegungsprozessor. Dieser spricht über eine Schnittstelle die 22 *Aktuatoren* des Roboters an [So20]. Hierbei besitzt AIBO drei Aktuatoren im Kopf, jeweils einen in Mund, Nacken und Hüfte, zwei in den Ohren, zwölf Aktuatoren in den Beinen und zwei im Schwanz [So20]. Zudem wird die Hardware durch eine *SIM-Karte* ergänzt, durch die eine Verbindung mit dem mobilen Internet möglich ist. Außerdem besitzt AIBO einen *Speicher*. Zuletzt kommuniziert der Hauptprozessor mit verschiedenen Ausgabemedien, unter anderem dem *Lautsprecher* sowie zwei *OLED-Bildschirmen* in AIBOs Augen und der *Frontkamera* in dessen Nase.

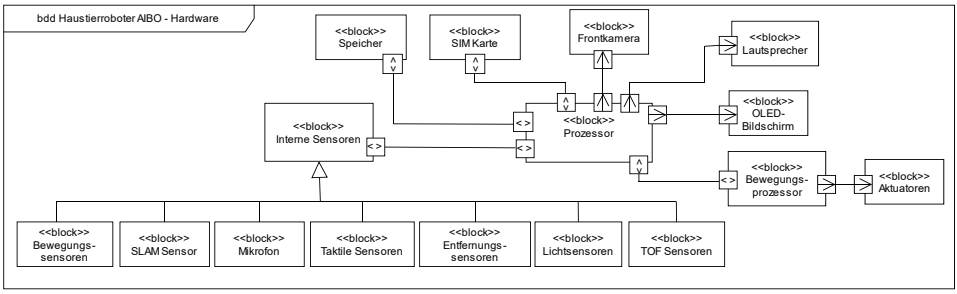


Abb. 6: Blockdefinitionsdiagramm der Hardware des Haustierroboters AIBO

4 Vergleich zwischen dem Haustierroboter AIBO und dem therapeutischen Roboter Paro

Broekens et al. bezeichnet sowohl den therapeutischen Roboter Paro, als auch den Haustierroboter AIBO als Begleitroboter. Diese bieten dem Menschen an erster Stelle Gesellschaft und damit eine Steigerung des psychischen Wohlbefindens [Br09]. Allerdings unterscheiden sich die beiden Systeme grundlegend in ihren Zielen. Das initiale Ziel hinter Paro lag vor allem darin ein künstliches Wesen zu erschaffen, welches ein lebensechtes Verhalten und Aussehen aufweist. Bei AIBO hingegen handelt es sich primär um eine futuristische Alternative zu einem Haustier mit Unterhaltungsfunktion.

Beim Vergleich der Systemgrenzen fällt auf, dass die systemischen Grenzen des therapeutischen Roboters Paro mit dessen physischen Grenzen übereinstimmen. Hingegen wird AIBO von einer Cloud und App ergänzt, was die Systemgrenzen erweitert. Dies ist für Paro nicht möglich, da dieser nicht über einen Zugang zum Internet verfügt. Dies hängt auch mit den Zeitpunkten zusammen, in denen die beiden Roboter jeweils entwickelt

wurden. Gemeinsamkeiten im Systemkontext bestehen zwischen beiden Systemen bezüglich des Einflusses von Licht sowie den Sicherheitsstandards. Bei Paro ist die Lärmkulisse der Umwelt zusätzlich entscheidend, da sie beeinflusst, wie gut der Roboter Stimmen wahrnehmen kann. Bei AIBO wurde diese Einflussgröße nicht aufgeführt, da der Roboterhund sich hauptsächlich in privaten Räumen aufhält, in denen Lärm keine größere Problematik darstellt. Zusätzlich spielen durch den Einsatz Paros in Pflegeeinrichtungen ethische Aspekte eine stärkere Rolle als bei AIBO. Zuletzt besitzt AIBO einige Variablen im Systemkontext, die für Paro unerheblich sind. Ein Beispiel ist die räumliche Umgebung, auf die das System reagieren muss. Hingegen kann Paro sich nicht selbstständig fortbewegen, somit entfällt diese Variable.

Im Bereich der Anwendungsfälle gibt es kaum Überschneidungen zwischen den beiden Systemen. Wird die Zielgruppe von Paro, die pflegebedürftigen Menschen, mit den Nutzern von AIBO verglichen, so unterscheiden sich deren Anwendungsfälle stark. Ein Grund hierfür ist, dass Paros Fähigkeiten eingeschränkter sind, als die des Roboterhunds. Paros Reaktionen auf das Verhalten des Nutzers bestehen aus Bewegungen und Geräuschen. Dabei reagiert er vorrangig auf seinen Namen sowie darauf, ob er gestreichelt oder grob behandelt wird. Demnach beschränken sich die Use Cases bei Paro darauf, ihm Zuneigung zu geben, sowie eine Beziehung zu ihm aufzubauen. Im Vergleich besitzt AIBO vielfältigere Fähigkeiten, ein verbessertes Lernverhalten und eine größere Bewegungsfreiheit. Hier steht weniger das Umsorgen im Fokus, als vielmehr der Spaß.

Anforderungen, die bei von beiden Systemen erfüllt werden müssen, sind Sicherheitsaspekte sowie die Reaktion in Echtzeit, um eine angenehme und authentische Interaktion zu ermöglichen. Die Anforderungen an Aussehen und Verhalten unterscheiden sich zwischen den beiden Robotern. Während Paro möglichst nahe an das Erscheinungsbild und Verhalten einer Robbe heranreichen soll, ist AIBO deutlich als Roboter erkennbar. Bei dem Lernverhalten der beiden Roboter zeigt sich ein ähnliches Bild: Paro lernt lediglich das Verhalten zu zeigen, das vom Nutzer bestärkt wird. AIBO ist zusätzlich fähig, Tricks zu lernen. Bei Paro ist hingegen die intuitive Interaktion deutlich wichtiger. Es ist essentiell, dass sich die verschiedensten Personengruppen mit Paro beschäftigen können, ohne vorher ein Handbuch lesen zu müssen.

Bei dem Gesichtspunkt der Hardware wird ebenso deutlich, dass der Aufbau des Haustierroboters AIBO komplexer ist, als der von Paro. Jedoch ähnelt sich der Grundaufbau der beiden Roboter. Im Zentrum beider Systeme steht der Hauptprozessor. Beide Systeme besitzen einen Speicher, Lautsprecher sowie Aktuatoren. Während Paro mit sieben Aktuatoren arbeitet, besitzt AIBO 22 Aktuatoren. Dadurch ist es AIBO möglich vielfältigere und flüssigere Bewegungen auszuführen als Paro. Zudem werden die Aktuatoren von AIBO nicht direkt vom Hauptprozessor angesteuert, sondern von einem Bewegungsprozessor. Beide Roboter besitzen interne Sensoren. Dabei handelt es sich um taktile Sensoren, Lichtsensoren sowie Mikrofon. Paro besitzt zudem Haltungssensoren um die Lage seines Körpers wahrzunehmen. AIBO beinhaltet dagegen verschiedene Sensoren, wie TOF, SLAM und Entfernungssensoren, mit denen er sich orientieren und frei bewegen kann.

Als Grenze der Arbeit soll angemerkt werden, dass der Roboter AIBO 14 Jahre später als Paro auf den Markt kam und somit von einem enormen technologischen Vorsprung profitiert, was besonders beim Vergleich der Hardware und Fähigkeiten der Roboter zu beachten ist. Zudem wurden die Roboter für unterschiedliche Einsatzgebiete entwickelt, welches die Ziele und Anforderungen stark beeinflusst. Des Weiteren ist keine Allgemeingültigkeit des Vergleichs gegeben und somit kein Rückschluss auf weitere tierähnliche Roboter möglich. Die Arbeit bezieht sich lediglich auf die beiden vorgestellten Roboter.

5 Fazit

Der therapeutische Roboter Paro wird vor allem im Bereich der Altenpflege eingesetzt. Der Roboter selbst wird dabei von verschiedenen Variablen in seiner Umgebung wie Akzeptanz, Ethik, den Pflegebedürftigen und dem Pflegepersonal beeinflusst. Bei Betrachtung der Anwendungsfälle der aktiven Nutzer fällt auf, dass Paro vor allem auf soziale Interaktion ausgelegt ist. Die Fähigkeiten des sozialen Roboters beschränken sich vor allem auf einfache Bewegungen und Geräusche, die dem Nutzer das Gefühl vermitteln, der Roboter sei lebendig und emotionsfähig.
Der Haustierroboter AIBO erschien 14 Jahre später als Paro und dient als Spielzeug. Ein Unterschied zu Paro ist die Systemgrenze: das Gesamtsystem wird von einer Cloud und einer App ergänzt. Der Kontext des Systems ist weniger von Ethik und Akzeptanz geprägt, dafür muss er seiner räumlichen Umgebung besser gerecht werden. Die Fähigkeiten von AIBO sind deutlich komplexer als Paros, dadurch ergeben sich mehr Anwendungsfälle. Wird die Hardware des Roboters betrachtet, beinhaltet diese viele Zusatzkomponenten, die AIBOs Fähigkeiten erweitern.
Eine Gemeinsamkeit der beiden tierähnlichen Roboter besteht darin, dass sie selbstständig agieren und auf Umweltreize reagieren. Ihr Verhalten wird so interpretiert, als hätten sie Gefühle. Jedoch beeinfluss der technologische Vorsprung AIBOs sowie der Einsatz in unterschiedlichen Domänen die Ziele, Anforderungen, Struktur und Fähigkeiten der Roboter stark.

Literaturverzeichnis

[Az12] Azeem, M. M. et al.: Emotions in Robots. In Emerging Trends and Applications in Information Communication Technologies. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012; S. 144–153.

[Br09] Broekens, J.; Heerink, M.; Rosendal, H.: Assistive social robots in elderly care: a review. In Gerontechnology, 2009, 8.

[Ca11] Calo, C. J. et al.: Ethical implications of using the paro robot with a focus on dementia patient care. In Proceedings of the 12th AAAI Conference on Human-Robot Interaction in Elder Care, 2011; S. 20–24.

[KW18] Knox, E.; Watanabe, K.: AIBO Robot Mortuary Rites in the Japanese Cultural Context, 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018; S. 2020–2025.

[Ma16] Matolycz, E.: Pflege von alten Menschen. Springer, Berlin, Heidelberg, 2016.

[Pa14] PARO Robots U.S, Inc., www.parorobots.com, Stand: 27.12.19

[Pa15] PARO Robots U.S, Inc., Gebrauchsanweisung, www.parorobots.com/maintenance.asp, Stand: 21.12.19

[Qu20] Qualcomm Technologies, Inc., www.qualcomm.com/products/snapdragon-system-package, Stand: 21.12.19

[Sh04] Shibata, T.: Ubiquitous surface tactile sensor: IEEE Conference on Robotics and Automation, 2004. TExCRA Technical Exhibition Based. IEEE, 2004; S. 5–6.

[So20] Sony Corporation of America, us.aibo.com, us.aibo.com/feature/spec.html, Stand: 27.12.19

[So18] Sony Corporation of America, helpguide.sony.net/aibo/ers1000/v1/en/index.html, Stand: 27.12.19

[SWT02] Shibata, T.; Wada, K.; Tanie, K.: Tabulation and analysis of questionnaire results of subjective evaluation of seal robot at Science Museum in London: Proceedings. 11th IEEE International Workshop on Robot and Human Interactive Communication. IEEE, 2002; S. 23–28.

[SYY97] Shibata, T.; Yoshida, M.; Yamato, J.: Artificial emotional creature for human-machine interaction: 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation. IEEE, 1997; S. 2269–2274.

[Wa05] Wada, K. et al.: Psychological and Social Effects of One Year Robot Assisted Activity on Elderly People at a Health Service Facility for the Aged: Proceedings of the 2005 IEEE International Conference on Robotics and Automation. IEEE, 2005; S. 2785–2790.

[WS07] Wada, K.; Shibata, T.: Living with seal robots - its sociopsychological and physiological influences on the elderly at a care house. In IEEE Transactions on Robotics, 2007, 23; S. 972–980.

[WS14] Weilkiens, T.; Soley, R. M.: Systems Engineering mit SysML/UML. Anforderungen, Analyse, Architektur. dpunkt.verl., Heidelberg, 2014.

Nutzensteigernde Faktoren zur Optimierung von Mobile Payment für Senioren als hilfsbedürftige Anwendergruppe

Frederike Möllers[1] und Tim Oberthür[2]

Abstract: Mobile Zahlungsprozesse, die mit Hilfe eines mobilen Endgerätes durchgeführt und als Mobile Payment bezeichnet werden, finden bereits in weiten Teilen des E-Commerce und stationären Einzelhandels Anwendung. Die angebotenen Lösungen entsprechen dabei allerdings noch nicht den spezifischen Anforderungen aller potenziellen Anwendergruppen. So haben Senioren, welche möglicherweise körperlich oder geistig eingeschränkt sind und damit eine hilfsbedürftige Anwendergruppe darstellen, besondere Anforderungen an Mobile Payment. In dieser Arbeit wird Mobile Payment aus Prozessperspektive betrachtet und es werden unter Berücksichtigung der besonderen Eigenschaften und Bedürfnissen von Senioren als hilfsbedürftige Anwendergruppe nutzensteigernde Faktoren definiert. Neben der Automatisierung bislang manuell aufwändiger Teilschritte mobiler Zahlungsverfahren lassen sich unter anderem die Verbreitung und Akzeptanz innerhalb der Anwendergruppe und im gesamten sozio-ökonomischen System als solche Faktoren identifizieren.

Keywords: Mobile Payment, Senioren, hilfsbedürftige Anwendergruppe, nutzensteigernde Faktoren, mobile Zahlungsverfahren, Prozessperspektive

1 Einleitung

„Nur Bares ist Wahres!“ Diese alte Redewendung verliert durch den zunehmenden Einsatz von Technologie im Handel immer mehr an Bedeutung. Ohne elektronische Zahlungen ist der heutige Onlinehandel undenkbar, aber auch der stationäre Einzelhandel verzichtet zugunsten moderner Zahlungsalternativen immer mehr auf Bargeld. Dabei werden nicht nur Debit- und Kreditkarten zur Zahlung eingesetzt, auch mobile Endgeräte finden zunehmend Anwendung. Die Zahlung mittels dieser Endgeräte, das sogenannte Mobile Payment, ist technologisch bereits weit entwickelt, praktisch ist der Verbreitungsgrad jedoch noch relativ gering. Dies kann daran liegen, dass Mobile Payment nicht für alle Anwendergruppen gleichermaßen zugänglich ist. Besonders für hilfsbedürftige Anwender, zu welchen auch Senioren zählen können, kann die Komplexität mobiler Zahlungsvorgänge abschreckend wirken. Weiterhin können individuelle Eigenschaften sowie soziale Einflüsse die Verwendung von Mobile Payment erschweren. Durch die Bereitstellung von auf die Anwendergruppe angepassten mobilen Zahlungsverfahren für Senioren, kann diese neue Zielgruppe erschlossen werden. Zudem können Senioren durch die Verwendung von Mobile

[1] Europäische Fachhochschule Rhein/Erft GmbH, Fachbereich Technologie und Management, Hammer Landstraße 89, 41460 Neuss, frederike.moellers@eufh-mail.de,
[2] Europäische Fachhochschule Rhein/Erft GmbH, Fachbereich Technologie und Management, Hammer Landstraße 89, 41460 Neuss, tim.oberthuer@eufh-mail.de

Payment und die resultierende Vereinfachung bislang manuell aufwändiger Zahlungsprozesse einen gewissen Grad an Autonomie im Alltagsleben zurückerlangen.

Diese Arbeit hat das Ziel herauszustellen, welche nutzensteigernden Faktoren zur Optimierung von Mobile Payment für Senioren als hilfsbedürftige Anwendergruppe zu berücksichtigen sind. Dazu wird in Kapitel zwei der Begriff Mobile Payment erläutert. Weiterhin wird in Kapitel drei auf die Eigenschaften der Anwendergruppe der Senioren eingegangen. Darüber hinaus wird in Kapitel vier herausgestellt, welche Potentiale und Herausforderungen bereits durch die Verwendung von Mobile Payment für die Anwendergruppe bestehen. Ein abschließendes Fazit wird in Kapitel fünf gezogen.

2 Mobile Payment

Der Begriff Mobile Payment beschreibt ein Bezahlverfahren, bei welchem Zahlungsdaten zwischen Kunden und Anbietern beim Erwerb von Waren oder Dienstleistungen mit Hilfe geeigneter mobiler Endgeräte ausgetauscht werden. Dadurch wird die Abwicklung der Zahlung zugrundeliegender Transaktionen initiiert [Pe15]. Als geeignete Endgeräte können moderne Mobiltelefone angesehen werden, da diese einen hohen Verbreitungsgrad aufweisen und über notwendige Technologien verfügen [Fu09]. Im Folgenden werden zugrundeliegende Zahlungsverfahren (Abschnitt 2.1) und die Verwendung von Mobile Payment mit geltenden Restriktionen erläutert (Abschnitt 2.2) sowie mobile Bezahlvorgänge aus Prozessperspektive betrachtet (Abschnitt 2.3).

2.1 Mobile Zahlungsverfahren

Bei Zahlungsverfahren kann zwischen originären und abgeleiteten Zahlungsverfahren unterschieden werden. Bei originären Zahlungsverfahren werden Werte direkt übermittelt. Bei abgeleiteten Zahlungsverfahren handelt es sich um Dienstleistungen, die von einem Dienstleister angeboten werden und auf mindestens einem originären Zahlungsverfahren basieren [Bu05]. Diese Unterteilung wird im Folgenden für mobile Zahlungsverfahren definiert. Bei einem originären mobilen Zahlungsverfahren wird ein Zahlungsmittel unmittelbar vom Endgerät des Zahlungssenders zum Endgerät des Zahlungsempfängers übermittelt. Bei Verwendung abgeleiteter Zahlungsverfahren kann der Zahlungsprozess mobil initiiert bzw. durchgeführt werden, es wird dabei jedoch auf mindestens ein originäres Zahlungsverfahren zurückgegriffen [Bu05]. Die direkte Übermittlung von elektronisch gespeichertem Buchgeld oder virtuellen Währungen kann als Beispiel für originäre Zahlungsverfahren im Mobile Payment angeführt werden. Oftmals ersetzt das Endgerät eine Debit- oder Kreditkarte, wobei der Zahlungsprozess zwar mobil initiiert, jedoch originär durch den Zahlungsdienstleister abgewickelt wird. Ein solcher Vorgang ist als Beispiel für ein abgeleitetes mobiles Zahlungsverfahren zu verstehen [Bu05].

2.2 Verwendung

Mobile Payment und die zugrundeliegenden originären und abgeleiteten Zahlungsverfahren können im E- bzw. M-Commerce, bei Zahlungen zwischen Privatpersonen und im stationären Einzelhandel Anwendung finden. Dabei zeichnet sich jedes der genannten Anwendungsszenarien durch spezifische Besonderheiten aus.

Unter E-Commerce wird der Handel über das Internet verstanden, bei dem die für den Erwerb von physischen sowie digitalen Waren und Dienstleistungen erforderlichen Produkt- und Zahlungsinformationen unabhängig von Ort und Zeit zwischen den beteiligten Handelspartnern ausgetauscht werden [KS02]. Wird der beschriebene internetbasierte Kaufprozess dabei mit Hilfe eines mobilen Endgerätes abgewickelt, wird von M-Commerce gesprochen [He12]. Dem Mobile Payment im E- bzw. M-Commerce liegen in der Regel abgeleitete Zahlungsverfahren zugrunde, da der Kunde auf der Plattform des Anbieters Zahlungsdaten hinterlegt und die Zahlung mit Abschluss des Kaufprozesses automatisch eingeleitet wird [Th99]. Handelsvorgänge, bei denen Waren und Dienstleistungen gegen Zahlung zwischen Privatpersonen ausgetauscht werden, werden als C2C-Commerce bezeichnet. C2C-Commerce kann entweder persönlich oder analog zum E- bzw. M-Commerce über das Internet abgewickelt werden [Eb07]. Auch Zahlungen im privaten Handelsverkehr können mobil abgewickelt werden, wobei vornehmlich originäre Zahlungsverfahren Einsatz finden. Dabei werden die zu zahlenden Geldwerte oder virtuellen Währungsbeträge direkt mittels dafür entwickelter mobiler Applikationen zwischen den Beteiligten transferiert [Le12]. Auch im klassischen stationären Einzelhandel können Zahlungen beim Erwerb von physischen Waren und Dienstleistungen über ein mobiles Endgerät erfolgen. Dabei können originäre (z. B. Zahlung mittels in proprietären Applikationen hinterlegten Werten) oder abgeleitete Zahlungsverfahren (z. B. Scannen der auf dem Endgerät gespeicherten Kreditkarteninformationen) zum Einsatz kommen [Ko11].

2.3 Betrachtung von Mobile Payment aus Prozessperspektive

Grundsätzlich kann Mobile Payment als Prozess angesehen werden. Ein Prozess beschreibt dabei eine zielorientierte, zeitlich-logische Abfolge von Teilschritten, die unter Verwendung von Informations- und Kommunikationstechnologien durchgeführt werden, um ein definiertes Ziel zu erreichen [Ga10]. Diese besondere Perspektive ermöglicht eine genauere Betrachtung von Mobile Payment und erlaubt die Untersuchung einzelner Teilschritte sowie die Identifikation von Unterprozessen durch Gruppierung von Aktivitäten. Diese bilden die Basis für die Identifikation von nutzensteigernden Faktoren im Mobile Payment für Senioren.

Zur Veranschaulichung dient das in Abbildung 1 dargestellte Prozessmodell, welches gemäß des Modellierungs-Standards BPMN 2.0 entwickelt wurde. Hier wird ein modellhafter Mobile-Payment-Prozess modelliert, bei welchem im stationären

Einzelhandel mit Hilfe eines modernen Mobiltelefons bezahlt werden soll. Dazu stehen sowohl ein originärer als auch ein abgeleiteter Zahlungsvorgang zur Verfügung. Das abgeleitete mobile Zahlungsverfahren basiert dabei auf dem Prinzip einer kontaktlosen Kreditkartenzahlung, wobei die physische Karte jedoch durch ein Mobiltelefon ersetzt wurde. Dabei wird eine automatische Erkennung des Zahlungsmittels bei Annäherung des Mobiltelefons des Kunden an das Kartenauslesesterminal des Anbieters angenommen. Beim originären mobilen Zahlungsverfahren wird die Zahlung über eine vom Händler und dem Kunden akzeptierte Zahlungsapplikation durchgeführt, die jedoch eine manuelle Eingabe verschiedener Zahlungsdaten auf Anwenderseite erfordert.

Start- und Zielpunkt des mobilen Zahlungsprozesses werden durch Kreise gekennzeichnet. Der logische Startpunkt liegt bei der zu tätigenden Zahlung, wobei der Endpunkt den Abschluss des Zahlungsvorgangs darstellt. Einzelne Prozessschritte werden durch abgerundete Rechtecke und mögliche Verzweigungen (exklusive Entscheidungssituationen des Anwenders) durch so genannte X-Konnektoren dargestellt. Teilschritte lassen sich gruppieren, was durch gestrichelte Linien dargestellt wird. Erforderliche Daten, werden durch Rechtecke mit geklappter Ecke symbolisiert.

Durch eine Gruppierung einzelner Teilschritte lassen sich zwei Unterprozesse erkennen, welche jeweils das originäre oder abgeleitete mobile Zahlungsverfahren darstellen. Dabei ist erkennbar, dass das abgeleitete mobile Zahlungsverfahren eine geringere Anzahl von manuellen Teilschritten enthält (hier drei Schritte) und weniger manuelle Dateneingaben erfordert.

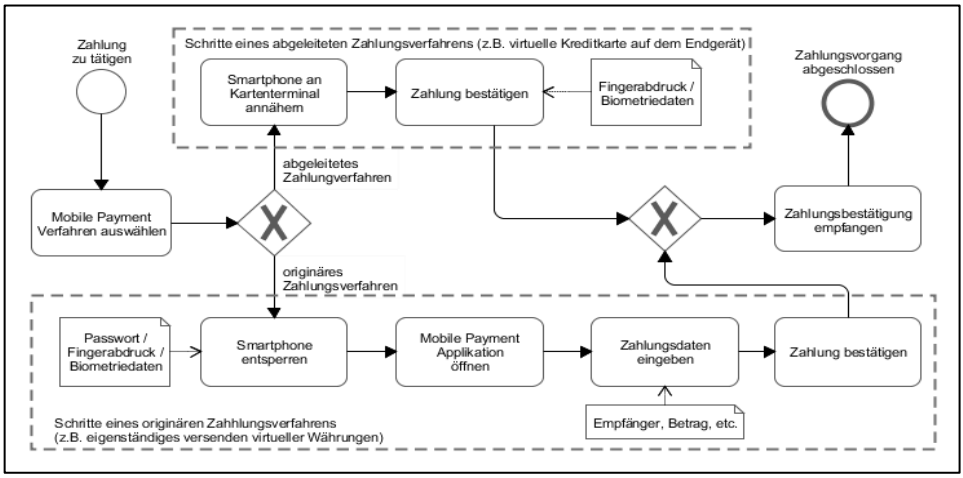


Abb. 1: Prozessmodell Mobile Payment

3 Eigenschaften von Senioren als hilfsbedürftige Anwendergruppe

Der Begriff Senior beschreibt Menschen höheren Alters. Im weiteren Verlauf der Arbeit werden gegebenenfalls hilfsbedürftige Menschen ab einem Alter von 60 Jahren der Anwendergruppe der Senioren zugeordnet. Neben dem Merkmal des Alters teilen sich Senioren auch weitere Eigenschaften. Es kommt vermehrt zu Einschränkungen von geistigen und körperlichen Eigenschaften sowie einem möglichen Fehlen von Akzeptanz gegenüber technischen Innovationen, die im Folgenden als individuelle Eigenschaften (Abschnitt 3.1) bezeichnet werden. Ferner können sich Einflüsse des sozialen Umfeldes (Abschnitt 3.2) auf das Verhalten von Senioren auswirken [FF12; Lu14].

3.1 Individuelle Eigenschaften

Individuelle Eigenschaften beschreiben kognitive sowie physiologische Eigenschaften einzelner Personen oder Personengruppen.

Ein Merkmal von Senioren ist, dass die kognitive Leistungsfähigkeit mit zunehmendem Alter sinkt. Diese Abnahme ist durch Veränderungen im menschlichen Gehirn begründet [Lu14; SW15]. Ferner wird das Gedächtnis, welches für die Speicherung von Daten verantwortlich ist, im hohen Alter schlechter. Diese Folgeerscheinung ist charakteristisch für Demenz. Viele Senioren sind von demenziellen Erkrankungen betroffen, wodurch Beeinträchtigungen im Alltag durch die Kombination von sinkender Gedächtnisfunktionalität sowie dem Rückgang weiterer Hirnleistungen entstehen [St11; HS06].

Mit zunehmendem Alter können Schwächungen in verschiedenen physiologischen Bereichen festgestellt werden, wozu auch die Sinnesorgane zählen [FF12]. Die Fähigkeit zu Sehen, wird bei Senioren beschränkt durch die Abnahme der Sehschärfe. Hinsichtlich der Motorik von Senioren ist festzuhalten, dass sowohl die Gelenkbeweglichkeit als auch die Skelettmuskulatur abnehmen. Ferner lässt die Zusammenarbeit des zentralen Nervensystems und der Skelettmuskulatur nach. Diese Zusammenarbeit zwischen Nerven und Sinnesorganen, auch Koordination genannt, nimmt beginnend mit einem Alter von 50 Jahren ab [Lu14; DHS11]. Es ist jedoch zu beobachten, dass die relative geistige sowie körperliche Leistungsfähigkeit von Senioren im Vergleich zu Angehörigen von Vergleichsgruppen gleichaltriger Anwender in vorhergehenden Zeitperioden weiter steigen [DHS11].

Die grundlegende Akzeptanz technischer Innovationen stellt ebenfalls eine für weitere Betrachtungen wichtige individuelle Eigenschaft dar. Diese wird unter Anderem von Aspekten wie der persönlichen Technikaffinität von Senioren, Nutzungserfahrungen und der Offenheit gegenüber sowie der Kenntnis von technischen Innovationen geprägt [Gi18; Do15].

3.2 Soziale Einflüsse

Der Wunsch nach Zugehörigkeit sowie der Austausch von Mitgliedern innerhalb von Gruppen liegt in der Natur des Menschen [MR15]. Die Kommunikation innerhalb dieser Gruppen führt zwangsläufig zu einer indirekten gegenseitigen Beeinflussung der Individuen. So lassen sich Gruppen beispielsweise im familiären Umfeld, dem Bekanntenkreis oder auch der Gesamtgesellschaft erkennen. Für Senioren stellt vor allem das familiäre Umfeld einen großen Einflussfaktor dar. Sie haben ein sehr großes Vertrauen in die Informationen und Empfehlungen von direkten Familienmitgliedern. Auf Grundlage dieser Informationen und Empfehlungen werden beispielsweise Kaufentscheidungen getroffen oder auch Innovationen in den Alltag von Senioren integriert. Gleiches gilt in etwa für die indirekte Beeinflussung durch Bekannte. Auch den Informationen und Empfehlungen von Bekannten wird von Senioren ein großes Vertrauen gegenübergebracht [Th92; Re01]. Die Verbreitung von Informationen über die Medien (z. B. Printmedien, Radio und Fernsehen) fördert die indirekte Beeinflussung der Gesamtgesellschaft. So können bspw. Politiker oder auch andere, aus den Medien bekannte Persönlichkeiten auf eine große Zielgruppe Einfluss auswirken. Die vermittelten Informationen sind für die genannte Anwendergruppe allerdings von geringerer Bedeutung als die Einflüsse von Familienmitgliedern oder Bekannten [Sc01].

4 Nutzung von Mobile Payment durch Senioren

Aufgrund des umfassenden Spektrums möglicher Anwendungsszenarien erstreckt sich die Zielgruppe von Mobile Payment über viele demographische Gruppen hinweg. So kommen auch hilfsbedürftige Anwender, zu welchen die in Kapitel drei behandelten Senioren zählen, als Verwender in Betracht. Grundsätzlich finden mobile Bezahlmethoden bereits in der Gesellschaft Anwendung. Allerdings lässt sich beobachten, dass Menschen im Seniorenalter bislang nur einen kleinen Teil der Anwendergruppe von Mobile Payment ausmachen [Re19].

Die Verwendung von Technologien im E-Commerce durch die beschriebene Anwendergruppe der Senioren wurde bereits erforscht. Die Ausführungen beziehen sich dabei jedoch bislang nicht spezifisch auf mobile Bezahlmethoden, stattdessen wird das Konsumentenverhalten im M-Commerce sowie in Mobile-Banking-Diensten betrachtet [MM20]. Die Untersuchung der Verwendung von Mobile-Payment-Lösungen durch die Anwendergruppe ist damit ein bedeutsamer Forschungsaspekt. Die in dieser Arbeit identifizierten nutzensteigernden Faktoren können eine Grundlage für weitere Betrachtungen darstellen.

Nachfolgend werden bereits bestehende Potentiale und Herausforderungen, welchen Anwendern von Mobile Payment im Seniorenalter gegenüberstehen, aufgeführt und erläutert (Abschnitt 4.1). Im Anschluss daran werden nutzensteigernde Faktoren für die Verwendung mobiler Bezahlmethoden durch die Anwendergruppe der Senioren ermittelt (Abschnitt 4.2).

4.1 Bereits bestehende Potentiale und Herausforderungen

Die Verwendung von Mobile Payment durch die beschriebene Anwendergruppe der Senioren, kann Potentiale und Herausforderungen hinsichtlich der zuvor genannten Eigenschaften und Restriktionen mit sich bringen. Auf diese wird im Folgenden näher eingegangen.

Für den Einsatz mobiler Zahlungsmethoden ist die Erfüllung technischer und sozio-ökonomischer Grundvoraussetzungen erforderlich. Dazu zählt vor allem die Verfügbarkeit geeigneter Endgeräte auf Kunden- sowie Anbieterseite. Viele Angehörige der betrachteten Anwendergruppe verfügen nicht über ein Smartphone [Re19]. Dies führt dazu, dass entsprechende Senioren keine Möglichkeit haben am mobilen Zahlungsverkehr teilzunehmen. Weiterhin ist für viele mobile Zahlungsmethoden ein mobiler Internetzugang erforderlich. Da Teile der betrachteten Anwendergruppe nicht über einen solchen Internetzugang verfügen, können die Betroffenen nicht am mobilen Zahlungsverkehr teilnehmen. Weiterhin ist der Schutz personenbezogener Daten erforderlich. Besonders relevant in Bezug auf die genannte Anwendergruppe sind neben Zahlungs- auch Konsum- und Gesundheitsdaten [Ko11].

Darüber hinaus ergeben sich auch hinsichtlich der Akzeptanz sowie der Verbreitung von mobilen Zahlungsmethoden in Bezug auf die beschriebene Anwendergruppe bestimmte Herausforderungen. Besonders relevant ist an dieser Stelle die stark ausgeprägte Bargeldpräferenz der Anwendergruppe, die mit einer Aversion gegenüber alternativen Zahlungsmitteln einhergeht. Weiterhin ist in Bezug auf Mobile Payment ein kritischer Einfluss des sozialen Umfeldes auf das individuelle Zahlungsverhalten zu beobachten, was einer Verwendung mobiler Bezahlverfahren entgegenwirkt [Ko11].

Auch in Bezug auf die Nutzerfreundlichkeit mobiler Bezahlmethoden bestehen Herausforderungen für die genannte Anwendergruppe. Diese können aus möglichen Einschränkungen der kognitiven sowie motorischen Fähigkeiten (Vgl. Abschnitt 3.1) resultieren. So ist beispielweise für das Verständnis der komplexen übergreifenden Zusammenhänge im Mobile Payment eine ausreichende kognitive Leistungsfähigkeit erforderlich. Die Eingabe zahlungsrelevanter Daten (z. B. per Touchscreen) in ein geeignetes mobiles Endgerät erfordert zudem neben kognitiven auch physiologische Fähigkeiten. So können notwendige Interaktionsschritte mit dem Endgerät und damit der Ablauf des zugrundeliegenden mobilen Zahlungsprozesses durch motorische Einschränkungen gestört werden. Dies kann besonders im Zusammenhang mit originären mobilen Zahlungsverfahren problematisch sein, da diese, wie in Abschnitt 2.3 identifiziert, oftmals über eine relativ hohe Zahl an manuell auszuführenden Teilschritten verfügen können. Abgeleitete mobile Zahlungsverfahren hingegen weisen meist eine geringere Anzahl manueller Teilschritte auf, sind aber durch die Beteiligung ggf. mehrerer Zahlungsdienstleister oftmals mit höheren Transaktionskosten verbunden, die durch den Anwender getragen werden müssen [Le12].

4.2 Nutzensteigernde Faktoren

Die bereits in Abschnitt 4.1 benannten Herausforderungen halten bislang den Großteil der Anwendergruppe der Senioren davon ab, Mobile Payment in den Alltag zu integrieren [Re19]. Da jedoch nicht nur Herausforderungen, sondern auch klare Nutzen für die Gruppe vorliegen, stellt sich die Frage nach Faktoren, welche den Nutzen von Mobile Payment für Senioren weiter steigern können, um die bereits vorhandenen Vorteile des Zahlungsprozesses nutzen zu können, die Akzeptanz zu fördern und dadurch den Verbreitungsgrad innerhalb der betrachteten Anwendergruppe zu erhöhen.

Einen dieser Faktoren stellt die ***Verbesserung der Grundvoraussetzungen*** dar. Durch die Erhöhung der Verfügbarkeit geeigneter Mobile-Payment-fähiger Endgeräte, wie bspw. Smartphones, sowie der Verbesserung des mobilen Internetzuganges steigt die Nutzung und damit verbunden die Anwendungsfälle des Mobile Payments. Der daraus entstehende Nutzen für Senioren ist, dass mehr Handelsunternehmen Mobile Payment als Zahlungsmittel akzeptieren. So können durch die Verbreitung und dem daraus entstandenen erweiterten Anwenderkreis neue Märkte erschlossen werden sowie der individuelle Nutzen in der Praxis erlebt werden.

Der Faktor der ***Steigerung des Verbreitungsgrades sowie der Akzeptanz*** ist maßgeblich für den Erfolg von Mobile Payment verantwortlich. Bedingt durch die starke Bargeldpräferenz von Senioren in Verbindung mit der Aversion gegenüber alternativen Zahlungsmitteln ist die Zahl von Nutzern innerhalb dieser Gruppe klein. Um eine Akzeptanz gegenüber Mobile Payment für Senioren zu schaffen, muss Aufklärungsarbeit geleistet werden. So kann bspw. durch eine geeignete Visualisierung sowie Erklärung der Prozessabläufe des Mobile Payment Akzeptanz geschaffen werden, da die Anwender anschließend den Ablauf des Mobile Payments verstehen. Durch eine Steigerung der Akzeptanz wird die Zahl kritischer sozialer Einflüsse im Umfeld kleiner, da immer mehr Menschen Mobile Payment akzeptieren. Der Nutzen dieses Faktors ist die weitere Verbreitung der Zahlung in Form von Mobile Payment und die daraus resultierende Möglichkeit auch an Märkten teilzunehmen, in welchen die alternativen Zahlungsmittel an ihre Grenzen stoßen (z. B. im M-Commerce).

Sind die Grundvoraussetzungen erfüllt und die Zahlung in Form von Mobile Payment in der Gesellschaft akzeptiert und verbreitet, muss das Mobile Payment an die Gruppe der Senioren angepasst werden, um den Nutzen für diese Gruppe weiter zu steigern.

Da Prozesse von hoher Komplexität in Verbindung mit vielen Teilschritten sowie benötigter Informationen, bedingt durch die kognitiven Eigenschaften von Senioren, zwangsläufig zu Problemen führen, wird ein weiterer nutzensteigernder Faktor in Form der ***Erhöhung des Automatisierungsgrades*** dargestellt. Durch eine Reduktion von manuellen Teilschritten sowie erforderlichen Dateneingaben wird der Prozess des Mobile Payments benutzerfreundlicher gestaltet, wodurch ein zusätzlicher Nutzen für Senioren entsteht.

Ein weiterer wichtiger Faktor für die Anwendergruppe der Senioren ist die ***Vereinfachung der Interaktionsmechanismen***. Durch die Gestaltung einer seniorenfreundlichen Benutzeroberfläche wird die Nutzung von Mobile Payment vereinfacht. So müssen mit Rücksicht auf die physiologischen Eigenschaften von Senioren z. B. eine größere Schrift, gute Übersichtlichkeit oder größere Buttons verfügbar sein. Ferner kann der Nutzen einer einfachen Bedienung hinsichtlich kognitiver Eigenschaften durch die Authentifizierung mit Hilfe von Fingerabdruck- und Biometriescans vereinfacht werden.

5 Fazit und Ausblick

Im Verlauf der Arbeit wurden nutzensteigernde Faktoren herausgearbeitet (Vgl. Abschnitt 4.2). Um den Nutzen von Mobile Payment für die Anwendergruppe der Senioren zu erhöhen muss zuerst eine ***Verbesserung der Grundvoraussetzungen***, als auch eine ***Steigerung des Verbreitungsgrades sowie der Akzeptanz*** erzielt werden. Ferner müssen auch technische Anpassungen vorgenommen werden, um die Faktoren der ***Erhöhung des Automatisierungsgrades*** und die ***Vereinfachung der Interaktionsmechanismen*** umsetzen und den Senioren zur Verfügung stellen zu können.

Eine weitere zentrale Erkenntnis dieser Arbeit ist die Betrachtung von Mobile Payment aus der Prozessperspektive. So können originäre, sowie abgeleitete Zahlungsverfahren in einzelne Prozessschritte aufgeteilt und näher betrachtet werden. Dies ermöglicht eine detaillierte Analyse der einzelnen Teilschritte in Bezug auf mögliche Herausforderungen für die Anwendergruppe der Senioren, wodurch sich mögliche nutzensteigernde Faktoren herausstellen ließen.

Werden die in Abschnitt 4.2 genannten Faktoren umgesetzt, kann ein größerer Nutzen für die Anwendergruppe der Senioren generiert werden. Bedingt durch die Erhöhung des Nutzens würde die Zahl der Anwender innerhalb dieser Gruppe steigen. Empfehlen die neu gewonnenen Anwender die Zahlung in Form von Mobile Payment, auf Grund der umgesetzten nutzensteigernden Faktoren, weiteren beispielsweise Freunden, Bekannten oder Familienmitgliedern innerhalb der Anwendergruppe der Senioren, so könnte ein Netzwerkeffekt eintreten. In diesem Fall steigen sowohl die Verbreitung als auch die Akzeptanz von Mobile Payment innerhalb der Anwendergruppe. Dabei wirken sich beide Aspekte wechselseitig positiv aufeinander aus.

Um die Nutzerzahl innerhalb der Anwendergruppe der Senioren weiter zu steigern, sollte ferner ein Fokus auf den Sicherheitsaspekt sowie bestehende Transaktionskosten mobiler Zahlungen gelegt werden. So muss eine Erhöhung der gefühlten Sicherheit bzw. des Kontrollempfindens im Rahmen der Nutzung erfolgen. Dies kann durch eine genauere Erklärung notwendiger Schritte und Abläufe, als auch durch behördliche Zertifizierungen von Mobile-Payment-Lösungen erreicht werden. Die zuvor genannten Aspekte sowie die Optimierung mit Hilfe der identifizierten nutzensteigernden Faktoren dürfen jedoch nicht zu einer Steigerung der Transaktionskosten im Mobile Payment

führen, um im Vergleich zu konkurrierenden Zahlungsmitteln keinen relativen Nachteil zu schaffen.

Mobile Payment hat durch die insgesamt ansteigende Anzahl von Anwendern, innerhalb und außerhalb der betrachteten Zielgruppe, das Potential sich in Zukunft zu einem Substitut für Barzahlungen und weiteren Zahlungsmitteln zu entwickeln.

Danksagung

Unser besonderer Dank gilt Prof. Dr. Nhung Nguyen der Europäischen Fachhochschule Rhein/Erft GmbH für die fachliche und persönliche Betreuung unserer Arbeit.

Literaturverzeichnis

[Bu05] BSI, Bundesamt für Sicherheit in der Informationstechnik: Sichere Zahlungsverfahren für E-Government, https://web.archive.org/web/20120117051921/https:/www.bsi.bund.de/cae/servlet/contentblob/476842/publicationFile/28064/4_Zahlv_pdf.pdf, 2005, Stand 10.05.2020.

[DHS11] Diehl, J.; Hollmann, W.; Strüder, H.: Hochaltrigkeit und körperliche Aktivitäten. In (Müller, L. Hrsg.): Hochaltrigkeit. Herausforderungen für persönliche Lebensführung und biopsychosoziale Arbeit, Wiesbaden 2011. VS Verlag für Sozialwissenschaften, Wiesbaden, S. 101-126, 2011.

[Do15] Doh, M.; Schmidt, L.; Herbolsheimer, F.; Jokisch, M.; Schoch, J.; Dutt, A; Rupprecht, F.; Wahl, H.: Neue Technologien im Alter. Ergebnisbericht zum Forschungsprojekt „FUTA“, https://www.psychologie.uni-heidelberg.de/mediendaten/ae/apa/futa-ergebnisbericht_2015.pdf, Stand: 03.07.2020.

[Dö17] Döpke, C.: Ökonomische und juristische Aspekte des Mobile Payments, www.abida.de/sites/default/files/14%20Mobile%20Payment.pdf, Stand: 10.05.2020.

[Eb07] Ebel, B.: B2B Geschäftsmodell. In (Olfert, K. Hrsg.): E-Business. Kompakt-Training, Ludwigshafen 2007. S. 46-53, Ludwigshafen 2007.

[FF12] Fiehler, R.; Fitzner, W.: Bilder des Alters im Sprachgebrauch. In (Schwitzer, K.P. Hrsg.): Expertisen zum Sechsten Altersbericht der Bundesregierung. Bd. 1: Individuelle und kulturelle Altersbilder, Wiesbaden 2012. VS Verlag für Sozialwissenschaften, Wiesbaden, S. 289-372, 2012.

[Fu09] Fuchs, T.: Mobile Computing. Grundlagen und Konzepte für mobile Anwendungen. Carl Hanser Verlag, München 2009.

[Ga10] Gadatsch, A.: Grundkurs Geschäftsprozess-Management. Methoden und Werkzeuge für die IT-Praxis: Eine Einführung für Studenten und Praktiker, 6. Aufl., Springer Vieweg, Wiesbaden 2010.

[Gi18] Ginnert, M.: Akzeptanz von digitalen Zahlungsdienstleistungen. Eine empirische Untersuchung am Beispiel von Mobile Payment mittels Smartphone im stationären Handel, Springer Gabler, Wiesbaden 2018.

[He12] Heinemann, G.: Der neue Mobile-Commerce. Erfolgsfaktoren und Best Practices, Springer Gabler, Wiesbaden 2012.

[HS06] Hartje, W.; Schulz, J.: Demenz. In (Ziegler, W. Hrsg.): Kognitive Neurologie, Stuttgart 2006. Georg Thieme Verlag KG, Stuttgart, S. 216-229, 2006.

[Ko11] Kollmann, T.: E-Business. Grundlagen elektronischer Geschäftsprozesse in der Net Economy, Gabler, Wiesbaden 2011.

[KS02] Kersten, W.; Schröder, K.: Wissensmanagement im E-Commerce. In (Keuper, F. Hrsg.): Electronic Business und Mobile Business. Ansätze, Konzepte, und Geschäftsmodelle, Wiesbaden 2002. Springer Gabler, Wiesbaden, S. 46-53, 2002.

[Le12] Leschik, S.: Mobile Payment. Techniken – Umsetzung – Akzeptanz, Nomos, Baden-Baden 2012.

[Lu14] Lugert, J: Demografischer Wandel und Nahversorgung. Herausforderungen und Lösungsansätze für das Marketing im deutschen Lebensmitteleinzelhandel, Diplomica Verlag, Hamburg, 2014.

[MM20] Msweli, N.; Mawela, T.: Enablers and Barriers for Mobile Commerce and Banking Services Among the Elderly in Developing Countries: A Systematic Review. In (Hattingh, M.; Matthee, M.; Smuts, H.; Pappas, I.; Dwivedi, Y.; Mäntymäki, M. Hrsg.): Responsible Design, Implementation and Use of Information and Communication Technology. I3E 2020. Lecture Notes in Computer Science, vol 12067. Springer International Publishing, Cham, S. 319-330, 2020.

[MR15] Mümken, A.; Rohmann, A.: Die sozialpsychologische Sicht auf die Gruppe. In (Wutzke, S. Hrsg.): Gruppenpädagogik in der Heimerziehung. 2. Aufl., Frankfurt a Main 2015. Internationale Gesellschaft für erzieherische Hilfen, Frankfurt am Main, S. 25-42, 2015.

[Pe15] Penzel, H. G. et.al.: Auf dem Weg von Electronic Payments in die Welt des mobilen Bezahlens. In (Dittrich, A.; Egner,T. Hrsg.): Trends im Zahlungsverkehr II, Köln 2015. Bank-Verlag, Köln, S. 71-94, 2015.

[Re01] Reitzler, R.: Versicherung und Risikoforschung. Bd. 40: Versicherungen für Senioren. Perspektiven für das Zielgruppen-Marketing, Springer Fachmedien, Wiesbaden, 2001.

[Re19] Regentahl, J.: Statista Umfrage FinTech und Mobile Payment in Deutschland, Https://de.statista.com/download/MTU3MzM5MzAxMCMjNTAxMDkjIzY1MTE1IyMxIyNudWxsIyNTdHVkeQ==, Stand: 10.05.2020.

[Sc01] Schweer, M.: Der Einfluss der Medien im Spiegel aktueller Vertrauensforschung. In (Schweer, M. Hrsg.): Der Einfluss der Medien. Vertrauen und soziale Verantwortung, Opladen, 2001. Leske + Budrich, Opladen, S. 7-20, 2001.

[St11] Streffer, J.: Das gesunde und kranke Gehirn von Hochaltrigen – Neurobiologie des Gehirns im hohen Alter. In (Müller, L. Hrsg.): Hochaltrigkeit. Herausforderungen für persönliche Lebensführung und biopsychosoziale Arbeit, Wiesbaden 2011. VS Verlag für Sozialwissenschaften, Wiesbaden, S. 77-90, 2011.

[SW15] Schaller, H.J.; Wernz, P.: Koordinationstraining für Senioren. 4. Aufl., Meyer & Meyer Verlag, Aachen, 2015.

[Th92] Thomas, A.: Grundriß der Sozialpsychologie. Bd. 2: Individuum – Gruppe - Gesellschaft, Hogrefe • Verlag für Psychologie, Göttingen, 1992.

[Th99] Thießen, F.: Bezahlverfahren im Internet: Systematisierung der Erscheinungsformen. In (Thießen, F. Hrsg.): Bezahlsysteme im Internet, Frankfurt am Main 1999. Knapp, Frankfurt am Main, S. 15-24, 1999.

Mehrwert der Digitalisierung

Allgemeines Referenzmodell zur Digitalen Transformation im öffentlichen Sektor

Karlson Hanke[1]

Abstract: Der öffentliche Sektor steht vor der Herausforderung, seine Verwaltung digital zu transformieren und einen durchgängigen Informationsfluss für alle Projektbeteiligten zu gewährleisten sowie Vorhaben (bspw. das Onlinezugangsgesetz) zu planen und umzusetzen. Ziel dieser Arbeit ist es, eine konsistente Vorgehensweise über alle föderalen Stufen bereitzustellen. Das erstellte Referenzmodell unter Einbeziehung aktueller Literatur umfasst sieben Elemente zur Digitalen Transformation der spezifischen Domäne. Mit dem kombinierten Reifegradmodell wird eine Klassifizierungs- und Vergleichsmöglichkeit nach festgelegten Kriterien geboten, um die Transformation in ihren Verwaltungsleistungen zu priorisieren und Umsetzungen zu verfolgen.

Keywords: Digitale Transformation, Reifegradmodell, Referenzmodell, Onlinezugangsgesetz, E-Government, Prozessmanagement, IT-Management, IT-Strategie, öffentliche Verwaltung

1 Einleitung

Durch die fortschreitende Digitalisierung in allen privaten und wirtschaftlichen Bereichen der Gesellschaft steigen auch die Anforderungen an den öffentlichen Sektor. Hier vollzieht sich ebenfalls eine Digitale Transformation. In den kommenden Jahren wird diese weiter an Bedeutung gewinnen. Durch das Onlinezugangsgesetz (OZG) werden der Bund, die Länder und die Kommunen bis 2022 verpflichtet, ihre Dienstleistungen elektronisch über sog. Verwaltungsportale anzubieten.

Problem bei der Umsetzung der Digitalen Transformation im öffentlichen Sektor ist die lange Kommunikationskette über die gesamte föderale Struktur hinweg. Lokalverantwortliche Verwaltungen bieten in unterschiedlichen Institutionen dieselben Verwaltungsleistungen an, bei denen die Erfahrung einer Umsetzung in einer Institution einer anderen dienen kann. Die beteiligten Fachexperten sind mit etablierten Vorgängen vertraut, jedoch im Kontext der Digitalen Transformation nicht so erfahren und deshalb mit relevanten Erfolgsfaktoren wenig vertraut. Dies mündet in methodischen Defiziten beim Messen und Priorisieren notwendiger technischer Umsetzungsschritte.

Um den Fortschritt und die Vergleichbarkeit der Umsetzung über alle Stufen des föderalen Systems offenzulegen, kann ein Referenzmodell für den öffentlichen Sektor wertvolle Unterstützung bieten. In einem solchen Modell werden alle Elemente der Digitalen Transformation betrachtet und ihre Reifegrade aufgenommen. Es werden digitale Vorreiter der

[1] Technische Universität Clausthal, Institut for Software and Systems Engineering, Arnold-Sommerfeld-Straße 1, 38678 Clausthal-Zellerfeld, karlson.hanke@tu-clausthal.de

cc by sa

Verwaltungen offengelegt und Einrichtungen mit geringerer Innovation erkannt. Hierdurch lässt sich eine weitere Umsetzung erleichtern und im föderalen Kontext harmonisieren. In dieser Publikation wird ein Modell zur Unterstützung der Digitalen Transformation im öffentlichen Sektor unter folgenden Fragestellungen erstellt:

- Wie lässt sich die Digitale Transformation im öffentlichen Sektor beschreiben?
- Welche Elemente lassen sich als Objekte der Digitalen Transformation erfassen und in ihrer Ausprägung messen?
- Wie lassen sich die Ausprägungen der Digitalisierung feststellen und über alle föderalen Stufen offenlegen?

Vor diesem Hintergrund beschreibt diese Publikation ein Referenz- und Reifegradmodell zur Digitalen Transformation im öffentlichen Sektor, in dem die relevanten Objekte dieses Modells aufgezeigt werden und jeweils in ihrer Reife eingestuft werden. Dies dient der Vergleichbarkeit von Projektvorhaben über alle föderalen Stufen. Damit fällt es allen Projektbeteiligten leicht den Informationsfluss in alle Richtungen zu unterstützen.

Das folgende Kapitel 2 beinhaltet die grundlegende Motivation dieser Arbeit, wobei das OZG als Grundlage der Digitalen Transformation verstanden wird und einführende Begrifflichkeiten vorgestellt werden. Aufbauend wird in Kapitel 3 zu bestehenden Forschungsgebieten eine Literaturrecherche zu verwandten Themen abgegrenzt. Kapitel 4 stellt das ausgearbeitete Allgemeine Referenzmodell zur Digitalen Transformation im öffentlichen Sektor mit seinen einzelnen Schritten vor. Abschließend werden in Kapitel 5 die Forschungsfragen noch einmal aufgegriffen und die Ergebnisse gewürdigt.

2 Digitale Transformation im öffentlichen Sektor

Das Onlinezugangsgesetz wurde im August 2017 vom Bundestag verabschiedet und setzt fest, dass innerhalb von fünf Jahren sämtliche Verwaltungsleistungen in Deutschland über ein Portal digital verfügbar sein müssen. Über ein individuelles Nutzerkonto sollen Bürgerinnen und Bürger sowie juristische Personen zukünftig Behördengänge digital erledigen. Der aktuelle Stand wird in Abschnitt 2.1 vorgestellt. Das Verständnis Digitaler Transformation wird in Abschnitt 2.2 aufgeführt. In Abschnitt 2.3 folgt eine Beschreibung von Referenz- und Reifegradmodellen, die eine konsequente Anwendung der vorherigen Abschnitte dieses Kapitels beschreiben.

2.1 Aktueller Stand der Umsetzung des Onlinezugangsgesetz (OZG)

Innerhalb der letzten drei Jahre wurden bereits 575 Leistungen definiert, die arbeitsteilig zwischen Bund, Ländern und Kommunen onlinefähig gemacht werden [BI19]. Hierzu geben sich die Projektbeteiligten für die Umsetzung des OZG selbst ein Reifegradmodell in Form einer Ordinalskala vor, nach der die Leistungen einer gewissen Reife zugeordnet

werden. Die festgelegten Stufen des IT-Planungsrats sind in der Tab. 1 nach dem OZG aufgeführt. In der Informationsplattform [Oz20] sind alle Verwaltungsleistungen mit diesen Stufen aufgeführt.

Reifegrad	**Erläuterung**
Stufe 0, Offline	Auf Verwaltungsportalen sind keine Informationen vorhanden.
Stufe 1, Information	Auf Verwaltungsportalen sind Informationen vorhanden.
Stufe 2 Formular-Assistent	Es wird eine Funktion angeboten, die beim Ausfüllen des Formulars o.ä. unterstützt. Online-Beantragung ist nicht mögl.
Stufe 3 Online-Leistung	Eine Online-Beantragung ist möglich. Ggf. müssen Nachweise in Papierform eingereicht werden.
Stufe 4 Online-Transaktion	Die Beantragung der Leistung kann einschließlich aller Nachweise online abgewickelt werden.
Stufe 5 Vernetzte Online-T.	Die Leistung kann vollständig digital abgewickelt werden. Für Nachweise wird das Once-Only-Prinzip umgesetzt.

Tab. 1 Reifegrade nach OZG in aufsteigenden Stufen [BI19]

Trotz der Meldung positiver Zwischenstände zeigt sich, dass die Komplexität offenbar sehr hoch ist [BI20] und bereits einige Abstriche gemacht wurden. Bspw. wird der „Online-Gateway Portalverbund“ zu einer „Suchen & Finden“ Website [BI20].

Kritik wird in unabhängigen Quellen laut. Eine rasche Harmonisierung von Gesetzen und Begrifflichkeiten fordert Barthel. Er schreibt: „Es ist keine Eins-zu-Eins-Übersetzung der analogen in digitale Prozesse möglich“ [Ba20, S. 88]. Außerdem mangele es an einem Informationsfluss Top-Down bei der Umsetzung des Onlinezugangsgesetzes [Me19]. Es entstünden informelle Netzwerke auf kommunaler Ebene, die erweitert werden müssten [Me19a]. Aus diesem Grund ist eine genauere Beschreibung der Reife zielführend.

2.2 Digitale Transformation im öffentlichen Sektor

Die Digitale Transformation kann als ein ganzheitlicher Ansatz verstanden werden, Kernprozesse und Dienstleistungen der öffentlichen Verwaltung über die herkömmlichen Digitalisierungsbemühungen hinaus zu überarbeiten. Im öffentlichen Sektor wird der Nutzen für diese Gruppen durch Verwaltungsleistungen erbracht. *Verwaltungsleistungen* sind die Ergebnisse der öffentlichen Verwaltungstätigkeit, durch die die ihr übertragenen Aufgaben erfüllt werden. Nach der Gesetzesbegründung des OZG beschränkt sich der Anwendungsbereich zunächst auf solche Verwaltungsleistungen, die über das Internet abgewickelt werden können. Mit den aufgestellten Forschungsfragen sollen die Bemühungen zur Umsetzung der *Transformationsstrategie* unterstützt werden:

- „Sie beschreibt die erforderlichen Veränderungen in der Wertschöpfungs- und Managementstruktur,
- sie gibt den Umgang mit digitalen Technologien vor und
- sie berücksichtigt den finanziellen Handlungsspielraum und -druck.“ [He19, S. 42]

Allerdings ist zu beachten, dass der finanzielle Druck in der jeweils lokal verantwortlichen Verwaltung nicht so hoch ist wie in der freien Wirtschaft. Ein niedriger finanzieller Druck auf das Kerngeschäft kann die bemerkte Dringlichkeit reduzieren. [He19, S. 47]. Der Handlungsdruck findet im öffentlichen Sektor vornehmlich durch die Gesetzgebung statt (Top-Down) in Teilen auch durch die Gesellschaft und die Verwaltungsmitarbeiter (Bottom-Up). Nach mehreren Durchläufen der Verwaltungsleistungen werden folglich wieder Effizienzen gewonnen. Referenz- und Reifegradmodelle helfen, solche Vorhaben zu quantifizieren.

2.3 Kombination von Referenz- und Reifegradmodellen

Nach der Definition von Laudon stellt ein Referenzmodell ein zweckorientiertes selektives Abbild der Realität dar, um eine Wiederverwendung von Modelleigenschaften zu erzielen [LS16, S.410ff]. Diesem Ansatz folgend sind Referenzmodelle für einen bestimmten Anwendungsbereich abstrahiert, d.h. sie haben den Anspruch, ein Ideal-Modell für eine Klasse von Institutionen darzustellen. Der Einsatz von Referenzmodellen zielt auf eine Aufwandsreduktion bei der Entwicklung spezifischer Modelle, der Dokumentation von vorhandenem Wissen und der Vorgabe zur Implementierung von Anwendungssystemen ab [GG20].

Referenzmodelle lassen sich gut mit Reifegradmodellen als Hilfsmittel des Prozessmanagements zur objektiven Bewertung, Verbesserung und dem Vergleich der Leistung von Prozessen und deren Input und Output-Ökosystem [Gr11] kombinieren. Sie sind ein Mittel, die Qualität der verwendeten Prozesse bei der Erstellung von Leistungen zu bewerten und diese kontinuierlich zu verbessern [Ku05]. Mit einer integrierten Betrachtung beider Modelle können zielgerichtet Stärken und Schwächen in Bezug auf das Prozessmanagement identifiziert und konkrete Handlungen abgeleitet werden.

3 Verwandte Themen

Etliche Forschungen im Rahmen der Digitalen Transformation betrachten Unternehmen, jedoch nicht den öffentlichen Sektor. An dieser Stelle bietet sich die Möglichkeit unter der Berücksichtigung der Besonderheiten der öffentlichen Hand Ergebnisse zu übertragen und auf Evidenz zu prüfen. Reifegradmodelle sind grundlegend für die Erstellung dieser Publikation. Dabei gibt es alleinige Modelle, die nur die Prozesse beschreiben, von denen auf die gesamte Organisation geschlossen wird. Für diese Anwendung vertreten in der Software Entwicklung SPICE (Software Process Improvement and Capability Determination),

als Umsetzung der ISO 15504, und CMMI (Capability Maturity Model Integration) die bekanntesten Werkzeuge zur Verbesserung der Entwicklungsprozesse mit Hilfe anerkannter Maßstäbe [Kn07]. Es definiert fünf Reifegrade (Inital, Managed, Defined, Quantitatively Managed, Optimizing), die eine aufsteigende Güte haben. Inital bedeutet, dass Prozesse auch abbrechen können, da diese nicht umfassend definiert sind. Optimized beschreibt den besten Wert in diesem Reifegradmodell. SPICE sticht in dieselbe Richtung hat aber darüber weitere Ableitungen – beispielsweise zur Verbesserung der Arbeit mit eingebetteten Systemen in der Automobilwirtschaft [VD15] [HM08, S.6ff.].

In der Verwaltung ist der Blick über die eigene Systemgrenze hinaus von hoher Bedeutung. Daher wurde ein ganzheitlicher Ansatz mit differenzierter Perspektive auf das gesamte E-Government-System und dessen Schnittstellen gesucht, wie es Wirtz und Daiser beschreibt [WD18]. Diese Arbeit verwendet den Ansatz von Appelfeller und Feldmann, die mit einem systematischen Leitfaden zur Reifegradmessung „Die Digitale Transformation des Unternehmens“ beschreiben [AF18]. Die Autoren gehen von Unternehmen aller Branchen aus, dies wird aber der Anwendung im öffentlichen Sektor nicht gerecht, daher ist eine Anpassung notwendig.

In verschiedenen Quellen wurden Bereiche der Digitalen Transformation aufgeführt, die ihre Schwerpunkte unterschiedlich legen. So wird zum Teil mehr Wert auf die Unternehmenskultur und das Management [FT17], [RR20], [Ro20] und zum Teil mehr auf die verwendete (IT-)Technologien und die Ablauforganisation [AF18], [Ma20] [WS18] gelegt.

Zusammenfassend auf der Grundlage der Verwaltungsleistungen, die mit Prozessen vergleichbar sind, wurde für diese Arbeit eine Quelle gewählt, die die Geschäftsprozesse im Zentrum hat. Aufgrund der Komplexität der horizontalen und vertikalen Vernetzung des öffentlichen Sektors war es darüber hinaus wichtig, die Akteure der Digitalen Transformation besonders zu berücksichtigen. Aus diesem Grund orientiert sich das Allgemeine Referenzmodell eng an den Elementen von Appelfeller und Feldmann [AF18] mit der Adaption auf den öffentlichen Sektor. Im folgenden Kapitel wird das Referenzmodell in der Weiterentwicklung unter Einbeziehung dieser Quellen vorgestellt.

4 Allgemeines Referenzmodell zur digitalen Transformation im öffentlichen Sektor

Eine ganzheitliche Sicht auf die jeweilige Institution bildet den Erfolgsschlüssel für die Digitale Transformation – aus diesem Grund wird ein spezifisches Referenzmodell von Appelfeller und Feldmann [AF18] für den öffentlichen Sektor abgeleitet. Bei der Verwaltung handelt es sich um ein komplexes System, das auf ein vereinfachtes Subsystem zu seiner Berechenbarkeit reduziert werden muss. Daher beschreibt diese Ausarbeitung ein Referenzmodell mit sieben Elementen von großer Bedeutung für die Digitale Transformation des öffentlichen Sektors, wie beispielsweise Prozesse, Mitarbeiter, Produkte/Services. In einem Reifegradmodell werden die unterschiedlichen Abstufungen der Digitalisierung,

die auftreten können, beschrieben. Hierfür ist folgende Vorgehensweise, wie Reifegrade erfasst und verbessert werden, praktikabel:

1. Einstufung der Reifegrade je Element im Referenzmodell
2. Bewertung der Relevanz nach Strategiehorizont je Element
3. Potential erkennen je Element der Digitalen Transformation
4. Aktuelle Stände vergleichen und verfolgen sowie Umsetzungen planen

Über alle Verwaltungsleistungen werden die Reifegrade und Relevanzen aufgenommen. Anschließend liegen Vergleichszustände offen über dieselbe Verwaltungsleistung gegenüber unterschiedlichen Verwaltungen bzw. alle Verwaltungsleistungen einer Verwaltung gegenüber anderen Verwaltungen der Vergleichsgruppe.

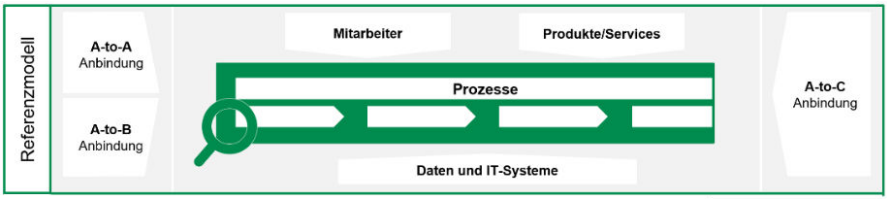


Abb. 1: Referenzmodell mit den 7 Elementen der Dig. Transformation im öff. Sektor (n. [AF18])

Um den öffentlichen Sektor in all seinen Ausprägungen greifbar zu machen und inhärente Komplexität zu reduzieren, wurde ein Referenzmodell, auf Grundlage der vorgestellten Literatur in Kapitel 3, verwendet. Die Elemente der Digitalen Transformation sind die Grundlage für dieses Modell. Abb. 1 zeigt, dass die Prozesse (grün) das Kernelement sind. Nach der Spezifikation des OZG gibt es davon 575, die digitalisiert werden sollen. Die sechs außenstehenden Elemente stützen die Prozesse. Ohne Mitarbeiter, Produkte/Services, Daten und IT-Systeme, Anbindungen unter Verwaltungen (A-to-A; A für engl. Administration), zwischen Verwaltungen und Unternehmen (A-to-B; B für engl. Business) und zwischen Verwaltungen und Bürgern (A-to-C; C für engl. Citizen) ist der öffentliche Sektor nicht handlungsfähig. Aus diesem Grund stützen diese die Verwaltungsleistungen, gehen darin ein oder sind Output der Verwaltungsleistung.

4.1 Einstufung der Reifegrade

Das Referenzmodell bildet die Grundlage für die Erfassung der Reifegrade. Jedes Element wird in seiner aktuell erreichten Reife erfasst. Im Folgenden werden die Kriterien aufgelistet, an denen festzustellen ist, in welchem Reifegrad sich die Verwaltungsleistung derzeit befindet. Bei den Prozessen (Tab. 2) ist entscheidend, in welcher Form diese angestoßen und durchgeführt werden. Hier ist die Steuerung über eine Business Process Engine [TS19] wünschenswert, die stets den Status der Verwaltungsleistung zeigt. Darüber hinaus bietet es die Grundlage für Automatisierung.

In einer digitalisierten Verwaltung ist es für Mitarbeiter wünschenswert, dass diese papierlos oder ortsungebunden arbeiten bzw. durch Roboter (Stichwort RPA, Robotic Process Automation [SW20]) Unterstützung erhalten. Tab. 3 zeigt auf, in welchen Reifegraden diese in das Referenzmodell eingehen. Bei Produkten und Services ist es für den öffentlichen Sektor wünschenswert, dass zukünftig angefragte Leistungen vorausgesagt werden, um dem Bedarf einen Schritt voraus zu sein. Eine Einstufung des Reifegrades in Bezug auf Algorithmen findet nach Tab. 4 statt. Ebenso ist es wünschenswert, dass die Verwaltungsleistungen komplett digital repräsentiert sind und die IT-Systeme ohne Medienbrüche eingesetzt werden. Für eine zukunftsgerichtete Verwaltung ist eine skalierbare Software Voraussetzung. Die Einstufung der Reifegrade für Daten und IT-Systeme sind in Tab. 5 aufgeführt.

Prozesse	**Beschreibung**
Reifegrad 1	analog, manuell, fremdgesteuert
Reifegrad 2	gering digitalisiert und automatisiert
Reifegrad 3	Überwiegend digitalisiert, automatisiert und autonom
Reifegrad 4	(fast) voll digitalisiert, automatisiert und autonom

Tab. 2: Reifegrade Prozesse

Mitarbeiter	**Beschreibung**
Reifegrad 1	Papierbasiert an fixem Schreibtisch/Verwaltungssitz
Reifegrad 2	Teils papierbasiert, teils papierlos an fixem Schreibtisch/ Verwaltungssitz, online während d. Arbeitszeit
Reifegrad 3	Papierlos, Büro Verwaltungssitz, zu Hause und Unterwegs, jederzeit online möglich
Reifegrad 4	Papierlos, flexibel, Zusammenarbeit mit Robotern

Tab. 3: Reifegrade Mitarbeiter

Produkte/Services	**Beschreibung**
Reifegrad 1	Keine Personalisierung, keine Algorithmen zur Optimierung
Reifegrad 2	Optimierung einzelner Eigenschaften mit Algorithmen
Reifegrad 3	Personalisierung, umfassende Optimierung
Reifegrad 4	Personalisierung, Vorhersage und Analyse jeder Leistung

Tab. 4: Reifegrade Produkte/Services

Daten und IT-Systeme	**Beschreibung**
Reifegrad 1	50-75% digitale Daten, System nicht skalierbar, Abteilungsinterne Software
Reifegrad 2	75-100% digitale Daten, System kaum skalierbar, Verwaltungsinterne Software

Daten und IT-Systeme	Beschreibung
Reifegrad 3	100% Anteil digitaler Daten, übergreifende Auswertungen, System bedingt skalierbar, integriert, Cloud-Lösungen
Reifegrad 4	Auswertungen und Vorhersagen, System ohne nennenswerten Aufwand skalierbar, integriert

Tab. 5: Reifegrade Daten und IT-Systeme

Schnittstellen A-to-A, A-to-B und A-to-C

In Unternehmen sind Mittel zur Verbesserung der Lieferkette und Lagerhaltung Standard. Sie bilden die Grundlage für eine weitreichende Vernetzung mit vor- und nachgelagerten Instanzen. Entsprechend muss dieses Vorgehen auch in der öffentlichen Verwaltung gelebt werden. Eine Einstufung der Reife der Schnittstellen für Administration-to-Administration (A-to-A), Administration-to-Business (A-to-B), Administration-to-Citizen (A-to-C) findet in den Reifegraden (analog, gering digital, überwiegend digital und digitalisiert) statt. Der öffentliche Sektor muss standardisierte Schnittstellen anbieten und diese spezifizieren. Diese dienen dazu, die IT-Systeme der Verwaltungen untereinander zu vernetzen, sowie Unternehmen und Bürgern eine technische Integration mit dem Ziel der Automatisierung ermöglichen. Routinetätigkeiten, wie manuelle Erfassungen, lassen sich von der Verwaltung in eine integrierte Instanz verlagern und die eigene Effizienz steigern.

Mit der Einordnung jedes einzelnen Elements liegen alle aktuellen Reifegrade der betrachteten Verwaltungsleistung offen vor. Dies bildet die Grundlage für den folgenden Schritt, bei dem die Relevanz bestimmt wird, inwieweit jedes Element ausgeprägt sein muss.

4.2 Bewertung der Relevanz der Elemente der Digitalen Transformation

Als Grundlage für die Bewertung der Relevanz dienen die Reifegrade je Element der Digitalen Transformation aus dem vorherigen Schritt. Durch die Kriterien höherer Reifegrade erhält der Anwender einen Eindruck, welche Zielerreichung möglich ist. Daraus folgend muss definiert werden, in welchem Planungshorizont für jedes Element eine Verbesserung angestrebt wird:

1: Kein Bedarf innerhalb der nächsten 4 Jahre;
2: Strategisch notwendig, wünschenswert innerhalb der nächsten 4 Jahre;
3: Taktisch notwendig, wünschenswert innerhalb der nächsten 1-2 Jahre;
4: Operativ notwendig, wünschenswert innerhalb der nächsten 12 Monate.

Die Potentialmatrix (Abb. 2) zeigt je Element der Digitalen Transformation eine Koordinate (x, y) = (Reifegrad, Relevanz). Eine vollständig ausgefüllte Matrix sollte sieben Eintragungen, analog der Elemente der Digitalen Transformation, enthalten, die in die Verwaltungsleistung eingebracht werden. In Abb. 2 sind drei Elemente exemplarisch aufgezeigt. Dabei können mehrere Elemente an einer Position (x,y) vertreten sein. Weitergehend als die vorgestellte Literatur im Kapitel 3 „Verwandte Themen“ handelt es

sich bei der Bewertung der Relevanz um ein Mittel aus dem Total Quality Management. Bei dieser Methode wird abhängig vom aktuellen Entwicklungsstand eine Selbstbewertung vorgenommen, die von einer Vielzahl der Beteiligten immer identisch eingeschätzt wird [Bu06].

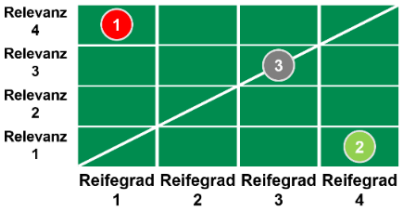


Abb. 2: Potentialmatrix mit Reifegraden und Relevanz (Eigene Darstellung)

Die Eintragungen aus der Potentialmatrix (Reifegrad, Relevanz) sind Grundlage für den kommenden Schritt, bei dem das Potential aufgedeckt wird und eine Priorisierung für die weitere Digitale Transformation vorgenommen werden kann.

4.3 Potential erkennen

Zur Offenlegung des Potentials ($Potential_i \in \mathbb{N}$) der Digitalen Transformation dienen die Relevanzen ($Relevanz_i \in \mathbb{N}$) und Reifegrade ($Grad_i \in \mathbb{N}$) je Element i:

$$Potential_i = Relevanz_i - Grad_i = \begin{cases} > 0, \text{ positiv} & \text{Verbesserung notwendig} \\ < 0, \text{ negativ} & \text{für Relevanz zu hoher Reifegrad} \\ = 0, \text{ neutral} & \text{Relevanz entspricht Reifegrad} \end{cases}$$

Interpretiert werden die Ergebnisse des Potentials nach einer bipolaren Skala. Je höher der Wert ausgeprägt ist, umso mehr Umsetzungspotential weist sich aus. Bei einem positiven Ergebnis (vgl. Abb. 2, 1 rot) ist deutliches Potential für eine weitere Digitale Transformation zu erkennen. Für dieses Element muss ein Vorgehen geplant werden, wie ein höherer Reifegrad erreicht werden kann. Bei einem negativen Ergebnis (vgl. Abb. 2, 2 grün) wurde beim vorherigen Vorgehen der Aufwand falsch eingesetzt. An dieser Stelle wird der digitale Transformationswille erkannt, in dem ein Element durch einen so hohen Reifegrad beschrieben wird. Zukünftig sollte die Priorität auf andere Elemente der Digitalen Transformation mit höherem Potential-Wert eingesetzt werden. Bei einem Potentialwert $Potential_i = 0$ entspricht der Reifegrad der gegebenen Relevanz des Elements i – aus aktueller Sicht wird der Zielzustand erfüllt.

Die analysierte Verwaltung hat ihre Werte offengelegt und kann priorisieren, welche Elemente weiter digital transformiert werden. Für jedes Element und jede Verwaltungsleistung muss nun ein entsprechendes Vorgehen für die einzelne Verwaltungseinheit geplant und umgesetzt werden. Dies schafft die Voraussetzung für zielgerichtete Innovation in der Verwaltung und ist folglich von entscheidender Bedeutung bei der Wettbewerbsfähigkeit [FB18].

4.4 Aktuelle Stände vergleichen

Der Vorteil der föderalen Struktur ist, dass einzelne Verwaltungen eine Vorreiterrolle einnehmen und ihre Erfahrungen anderen bereitstellen können. Allgemein sollte für das Innovationsmanagement in der Verwaltung mehr Raum gegeben werden und ein Austausch bspw. in Innovationslaboren zwischen unterschiedlichen Gruppen forciert werden. Hier kann Beteiligten von Leuchtturmprojekten oder welchen, die eine Rolle als Aggregator einnehmen, der Raum zum Erfahrungsaustausch mit anderen Verwaltungen (Vorreitern, Nachzüglern) geboten werden. Mit den erfassten Reifegraden des Referenzmodells ergibt sich ein umfassender Blick auf die Verwaltungsleistung. Eine zentrale Datenhaltung dieses Modells ist der Erfolgsfaktor dieser Strategie – umgesetzt wird nach wie vor lokal.

Um den Informationsfluss über alle Instanzen zu unterstützen, können die digitalen Vorreiter aus den zentral gelagerten Daten herausgefiltert werden:

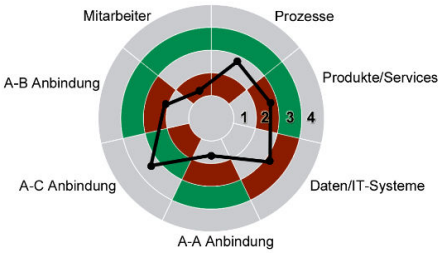


Abb. 3: Instanz einer Verwaltungseinheit mit Ist-, Ziel- und Vergleichszustand (Eig. Darstellung)

Die aufgenommenen Reifegrade können als Ist-Zustände (Abb. 3, rot), die Relevanzen als Ziel-Zustände (Abb. 3, grün) eingetragen werden. Durch die Meldung an eine zentrale Einheit ist es möglich, alle Meldungen zu einem Zeitpunkt zu kumulieren und diese auch als Vergleichszustand einzutragen (Abb. 3, schwarz). Der Vergleichszustand wird durch den einfachen Mittelwert aller mit sich selbst verglichenen Verwaltungen und Verwaltungsleistungen je Element der Digitalen Transformation errechnet.

Dieses Modell bietet einen transparenten Blick auf die Organisation, die von allen Beteiligten einheitlich interpretiert werden kann [Lu02]. Daher bietet es eine ausgezeichnete Grundlage für die Kommunizierbarkeit der gesetzten und erreichten Ziele. Die Sichten auf dieses Modell können entweder eine gesamte Verwaltung oder eine Verwaltungsleistung über mehrere Verwaltungen kumuliert sein. Vergleichszustände können visuell angepasst werden, so dass diese auf die jeweilige Sicht zur Verdeutlichung dienen.

5 Schlussbetrachtung

Zusammenfassend wurde die Digitale Transformation im öffentlichen Sektor in Form eines Referenzmodells aufgezeigt. Die übergeordneten Kategorien der Digitalen Transformation je Element werden je Verwaltung und Verwaltungsleistung in vier Reifegraden

aufgenommen und nach ihrer Relevanz bewertet. Deutlich wird mit den entsprechenden Kriterien, in welchem Reifegrad die Elemente eingebracht werden. Bei einer höheren Relevanz als der Reifegrad muss ein Vorgehen zur Erreichung eines höheren Reifegrades geplant werden, um das Verbesserungspotential auszuschöpfen. Die Offenlegung der Reifegrade im Ist- und Ziel-Zustand hilft, Vorreiter und Nachzügler über alle föderalen Stufen zur Lenkung der Umsetzung der OZG zu verfolgen.

Die aufgeworfenen Forschungsfragen dienten als Leitfaden das Referenzmodell zu erstellen. Bei kritischer Betrachtung werden Referenzmodelle nach einem induktiven-deduktiven Zyklus entwickelt. Bis zu diesem Forschungszeitpunkt konnte nur eine deduktive Strategie nach Fettke angewendet werden [FB18]. Als konsequente Fortsetzung dieses Ansatzes sollen Reifegrade der öffentlichen Verwaltung aufgenommen werden, um den Fortschritt der Umsetzung von IT-Strategievorhaben wie dem OZG festzustellen.

Bei kritischer Betrachtung zeigt sich bei dem aktuellen Ansatz, dass Abhängigkeiten zwischen Prozessen bisher nicht ausreichend gewürdigt werden konnten. Grundlage für jede zu erbringende Verwaltungsleistung muss bspw. eine verlässliche Onlineautorisierung sein, diese sollte bei der Digitalen Transformation vorrangig betrachtet werden.

Das Referenzmodell ermöglicht über die OZG-Umsetzung hinaus eine ständige Verwendung zur Harmonisierung über alle föderalen Stufen hinweg und dient der Erreichung höherer Reifestufen. Geachtet werden sollte darauf, dass die Digitalisierung nur so weit getrieben wird, wie es der Institution Vorteile bringt – eine Digitalisierung zum Selbstzweck ist nicht erstrebenswert [AF18].

Literaturverzeichnis

[AF18] Appelfeller, W.; Feldmann, C.; Die Digitale Transformation des Unternehmens. SpringerGabler, Berlin, 2018

[Ba20] Barthel, C. (Hrsg.); Managementmethoden in der Verwaltung, SpringerGabler, 2020

[BF18] Burgelman, R. A.; Floyd, S. W.; Laamanen, T.; Mantere, S.; Vaara, E.; Whittington, R.: Strategy processes and practices: Dialogues and intersections; Strategic Management Journal, Nr. 39, Ausgabe 3, S. 531-558, März 2018

[BI19] Bundesministerium des Innern, Programmmanagement Digitalisierung Phase 2, it-planungsrat.de/SharedDocs/Downloads/DE/Entscheidungen/28_Sitzung/TOP03_Anlage1_Programmmanagement_Phase_2.pdf?blob=publicationFile&v=9, 22.04.2020

[BI20] Bundesministerium des Innern, Pressemittteilung; 0kh.de/qgibi20, 24.04.2020

[Bu06] Burkhard, Carina. (2007). TQM-Trend-Matrix - Methode zur progn. Analyse TQM.

[FB18] Fettke, P., Brocke, J. v.: Referenzmodell - Enzyklopaedie der Wirtschaftsinf. (2018)

[FT17] Fricke, H., Thiesen, T.: Mittelstand im Wandel, 2017; https://kommunikation-mittelstand.digital/content/uploads/2017/01/Leitfaden_Ermittlung-digitaler-Reifegrad.pdf

[Gr11] Grande, M.: 100 Minuten für Anforderungsmanagement. Vieweg+Teubner, 2011

[GG20] Gramlich, L, Gluchowski, P., Horsch, A., Schäfer, K., Waschbusch, G. (Hrsg.) (2020). Gabler Banklexikon. Springer Gabler, Berlin. 2020

[He19] Hess, T.: Digitale Transformation strategisch steuern, Springer, Wiesbaden, 2019

[HM08] Hoermann, K.; Mueller, M.; Dittmann, L.; Zimmer, J.: Automotive Spice in Practice; rockynook Computing, Santa Barbara, Kalifornien, USA, 2008

[Kn07] Kneuper, R.: CMMI, Verbesserung von Software- und Systementwicklungsprozessen mit Maturity Model Integration (CMMI-DEV). dpunkt.Verlag, Heidelberg, 2007

[Ku05] Kunau G, Junginger M, Herrmann T, Krcmar H (2005): Ein Referenzmodell für das Service Engineering mit multiperspektivischem Ansatz. In: Herrmann T, Kleinbeck U, Krcmar: Konzepte für das Service Engineering. Heidelberg, Physika-Verlag. S. 187-216

[LS16] Laudon, J; Schoder, D.; Martin, E.; Knebel-Heil, H. (2016). Wirtschaftsinformatik - eine Einführung. Pearson Deutschland, Hallbergmoos

[Lu02] Ludewig, J., (2002). Modelle im Software Engineering – eine Einführung und Kritik. In: Glinz, M. & Müller-Luschnat, G. (Hrsg.), Modellierung 2002, Modellierung in der Praxis – Modellierung für die Praxis. Bonn: Gesellschaft für Informatik e. V. (S. 7-22).

[Ma20] Reifegradmodell Digitale Geschäftsprozesse; https://www.mach.de/news/Reife-digitaler-Geschaeftsprozesse-messen, 23.04.2020

[Me19] Mergel, I.: Digitale Transformation als Reformvorhaben der deutschen öffentlichen Verwaltung; dms – der moderne staat – Zeitschrift für Public Policy, Recht und Management, 12. Jg., Heft 1/2019, S. 162-171

[ME19a] Mergel, Ines, Noella, Edelmann & Haug, Nathalie (2019). Defining digital transformation: Results from a discourse analysis of expert interviews

[Oz20] informationsplattform.ozg-umsetzung.de

[Ro20] ROI Management Consulting AG, Messung und Bewertung der Digitalisierungs-Reifegrade; https://www.roi.de/effizienzsteigerung-durch-digitalisierung-industrie-40/reifegrad-der-digitalisierung/, München; Zuletzt geprüft: 23.04.2020

[RR20] Reinhard, H.; eRentz, R.; Sommerfeld, T.: Digitalisierung steuerbar machen, Controlling & Management Review 2/2020, S. 46-51, 2020

[SW20] Scheppler, B.; Weber, C: Robotic Process Automation. Informatik Spektrum, 2020

[TS19] Thuis, P.; Stuive, R.: Planning and control of business processes; In: Business Administration, Kapitel 6, Seite 37, Ausgabe November/2019

[VD15] VDA QMC Working Group 13 / Automotive SIG: Automotive SPICE Process Assessment / Reference Model. Veröffentlicht: 16.07.2015

[WD18] Wirtz B.W., Daiser P. E-Government. In: Voigt R. Handbuch Staat (S.981ff). Springer

[WS18] Wolf, T.; Strohschen, J.-H.: Digitalisierung: Definition und Reife, Informatik Spektrum 41 1/2018, Springer Verlag, Berlin, 2018

Vergleichende Analyse der Komplexität musikalischer Dimensionen von Klavierwerken europäischer Komponisten anhand von MIDI-Files

Mihail Miller[1]

Abstract: Die vorliegende Arbeit analysiert mithilfe von objektiven Metriken Musikwerke namhafter europäischer Komponisten, die für Klavier und Cembalo geschrieben wurden, explorativ in Bezug auf die Komplexität ihrer musikalischen Dimensionen. Die für diese Analyse notwendigen Messungen wurden an 311 annotierten mehrspurigen MIDI-Files vorgenommen. Das Korpus wurde deskriptiv-statistisch untersucht. Abschließend wurden die Ergebnisse auf ihre Plausibilität und ihre Aussagekraft im Bezug auf das zeitgeistige Musikverständnis der erfassten Zeiträume geprüft.

Keywords: Komplexität; MIDI-Files; Analyse; quantitativ; Klavier; Cembalo; Komponisten

1 Einleitung

Den Musikwerken europäischer Komponisten werden zur Beschreibung ihres Stils von Fachkreisen charakteristische Merkmale zugewiesen. Auch auf epochaler Ebene werden unterschiedliche Eigenschaften musikalischer Strömungen beschrieben. So schreiben Fachkreise der (Wiener) Klassik[2] im Vergleich zum Barock eine einfachere Harmonik zu. Solche oder ähnliche Attribuierungen lassen sich jedoch bisher nur qualitativ aufstellen und sind nur intersubjektiv nachprüfbar. Die Motivation der vorliegenden Arbeit besteht darin, die Komplexität von 311 diachron eingeordneten Klavier- bzw. Cembalowerken namhafter Komponisten europäischer Kunstmusik mithilfe von objektiven Metriken zu berechnen und deskriptiv-statistisch zu untersuchen. Weiterhin sollen Korrelationen der Metriken berechnet werden, um auf eventuelle Interdependenzen schließen zu können. Die Evaluation der Metriken auf ihr Wahrheitsgehalt im Abgleich mit musikwissenschaftlichen Analysen zu den untersuchten Werken ist nicht Teil dieser Arbeit.

[1] Universität Leipzig, Wirtschaftsinformatik, Nürnberger Str. 5, 04103 Leipzig, mihail.miller@yahoo.de

[2] Allgemein wird Musik, die vor der Möglichkeit der elektroakustischen Verstärkung im europäischen Raum entstand, vielfach unter dem Begriff „klassische Musik“ zusammengefasst. Im Folgenden soll der Begriff jedoch in seiner eigentlichen Bedeutung als epochale Bezeichnung der (Wiener) Klassik (etwa 1780 - 1820) verstanden werden.

2 Verwandte Arbeiten

Das Vorgehen dieser Studie orientiert sich an der Arbeit von Madsen und Widmer von 2015, bei der die Verwendbarkeit von Komplexitätsmaßen zur Erkennung der Melodiespur in mehrspurigen MIDI-Files untersucht wurde. Der Informationsgehalt der MIDI-Files wurde hierbei auf Tonklassen (pitch class), Tonintervalle (melodic interval) und auf den zeitlichen Abstand zwischen zwei Tönen (inter-onset interval) untersucht [MW15]. Die Klassifikation der musikalischen Dimensionen stütze sich dabei wesentlich auf die Arbeiten von Conklin [Co06]. Zur Approximation der Komplexität dieser Dimensionen wurde das Entropiemaß von Claude E. Shannon (1948) genutzt, welches den mittleren Informationsgehalt einer Nachricht angibt. Dabei besteht die Grundannahme, dass ein höherer Informationsgehalt verarbeitungsintensiver und somit komplexer ist. So setzte Youngblood 1958 bei seiner Studie zur Determinierung des musikalischen Stils das Shannonsche Entropiemaß frühzeitig im Kontext von Tonfolgen ein, unter der Annahme, dass Redundanz ein werkprägendes Stilmittel sei [Yo58]. Einen anderen Ansatz wählte Simonton 1984 bei seinen Untersuchungen zur Entwicklung der melodischen Originalität von musikalischen Themen. Dabei wurde ein Korpus von 15.618 Themen auf die Wahrscheinlichkeit des Auftretens eines Tonintervalls im Vergleich zum gesamten betrachteten Zeitraum untersucht [Si84].

3 Korpus

Für die Analyse von Musikwerken eignet sich das MIDI-Format aus zweierlei Hinsicht:

1. Die Notation des Musikstücks liegt bereits maschinenlesbar vor; eine Anwedung von verlustbehafteten Methoden wie OMR (Optical Music Recognition) zur Extraktion der Informationen aus den originalverlegten Notenblättern ist somit nicht notwendig.
2. Das Format ist weitverbreitet; Partituren im MIDI-Format sind leicht aufzufinden.

Während in der Theorie sich im MIDI-Format exakte Informationen zur Komposition des Musikstücks abbilden lassen, wird von diesen Eigenschaften sehr heterogener Gebrauch gemacht. Infolgedessen mussten für den Einschluss eines MIDI-Files in die Untersuchung Kriterien festgelegt werden, die verlässliche Aussagen zu den Musikwerken zulassen:

1. Das MIDI-File ist rhythmisch exakt sequenziert (nicht live eingespielt) und enthält korrekte Informationen zur Zeitsignatur und zum Tempo.
2. Das MIDI-File ist vom Format-Typ 1 und somit synchron mehrspurig [MI]. Damit soll die melodische und rhythmische Komplexität besser approximiert werden können.

Als Ausgangsbasis für das Korpus dienten die MIDI-Archive „Classical Disklavier-I(-VII)“, die auf der Seite „Disklavier™ World“ [3] zu finden sind. Die ZIP-Archive wurden extrahiert und anschließend mithilfe der Python-Bibliothek Mido[4] auf grundlegende Eigenschaften (Typ, Zeitsignatur, Tempo, Spieldauer) untersucht. Anschließend wurde eine Heuristik eingeführt, um eine erste Filterung der MIDI-Files nach den oben beschriebenen Kriterien vorzunehmen. Der Positiv-Test zum Einschluss eines MIDI-Files in die Untersuchung bestand in der automatisierten Prüfung, ob die MIDI-Datei vom Typ 1 ist sowie eine von den Default-Werten abweichende Angabe der Zeitsignatur (4/4 Takt) und des Tempos (120 BPM) besitzt [MI]. Diese Heuristik könnte zum fälschlichen Ausschluss von MIDI-Files führen, deren Zeitsignatur und Tempo tatsächlich den Default-Werten entsprechen. Relevanter war jedoch die Filterung fälschlich positiv kategorisierter MIDI-Files, also solcher, die trotz Einschluss durch die beschriebene Heuristik, die notwendigen Kriterien nicht erfüllen. Dafür wurden die verbliebenen MIDI-Files in den gefilterten Archiven nochmals unter manueller Einsicht in einer DAW[5] geprüft. Gründe für den Ausschluss fälschlich positiv kategorisierter Files aus der Untersuchung waren diverser Natur: Zum einen enthielten einige MIDI-Files trotz ihrer Festlegung als Typ 1 in der Tat nur eine Spur, wiederum andere effektiv nur eine Notenspur, da die restlichen Spuren missbräuchlich zur Angabe von Metainformationen zum Komponisten oder zum Autor des MIDI-Files genutzt wurden. Weiterhin waren mehrere MIDI-Files trotz der Abweichung von den Default-Werten bei der Zeitsignatur und dem Tempo nicht sequenziert, sondern wurden live eingespielt, obwohl es bei live eingespielten MIDI-Files typischerweise der Fall ist, dass die Default-Werte beibehalten werden. Eine Kombination der Ausschluss-Kriterien war selbstverständlich ebenfalls möglich. Die „False Positives“ wurden als solche gekennzeichnet. Anschließend wurden die Partituren auf ihr redundantes Vorkommen geprüft. Dies erfolgte Anhand des Titels, der in den MIDI-Archiven „Classical V“ und „Classical VI“ zumeist eine Werknummer enthielt. Mithilfe der beschriebenen Vorgehensweise konnten 308 Titel aus den Archiven „Classical V“ und „Classical VI“ in die Untersuchung eingeschlossen werden. Für die Archive „Classical I“, „Classical II“ und „Classical VII“ wurde eine leicht abgewandelte Heuristik angewandt, da die enthaltenen MIDI-Files bis auf drei Ausnahmen eine Clock Rate von 512820 µs angaben, welches nach [GB03] der Standardwert der offiziellen Disklavier™ MIDI-Files ist. Auch diese MIDI-Files erlauben keine verlässlichen Aussagen zur Komposition des Musikstücks, da auch sie größtenteils live eingespielt wurden und bereits aufgrund der identischen Clock Rate die tatsächliche Information zum Tempo, die für die Bestimmung der Stücklänge notwendig ist, nicht enthalten. Zudem waren die MIDI-Files in den Archiven „Classical I“ und „Classical III“ alle vom Typ 0 und mussten vollständig ausgeschlossen werden, während sich im Archiv „Classical VII“ drei weitere MIDI-Files befanden, welche in die Untersuchung eingeschlossen werden konnten.

[3] http://www.kuhmann.com/Yamaha.htm. Zuletzt zugegriffen: 17. Juli 2020. Auf der Webseite sind Informationen zu den Titeln der ZIP-Archive enthalten, welche unter anderem darüber Auskunft geben sollen, ob die MIDI-Files vorwiegend sequenziert oder live eingespielt wurden. Diese ohnehin äußerst wagen Angaben stellten sich bei der Evaluation der MIDI-Files als unverlässlich dar

[4] https://pypi.org/project/mido/. Zuletzt zugegriffen: 19. Juli 2020.

[5] DAW = Digital Audio Workstation; Integrierte Software zum Zwecke der Musikproduktion

Für die übriggebliebenen 311 MIDI-Files wurde anhand des Titels nach dem Kompositionsjahr in der Petrucci Music Library (IMSLP[6]) gesucht, falls dieses fehlte wurde das Jahr der Erstveröffentlichung notiert. Falls ein Jahreszeitraum vorzufinden war, wurde zunächst dieses notiert. In einer weiteren Spalte, wurde das Jahr für die Analyse notiert, welches je nach Verfügbarkeit das letzte Kompositionsjahr oder das Jahr der Erstveröffentlichung war.

Die Titel in den Archiven „Classical II“ und „Classical IV“ waren für die eindeutige Identifizierung der Werke nicht ausreichend, weshalb sie bereits vor der beschriebenen Evaluation der Archive ausschieden. Die Evaluation der MIDI-Files und Annotation der Jahre wurde für das jeweilige Archiv in einem CSV-File vorgenommen. Für die Erfüllung einer Eigenschaft wurde jeweils eine 1 angegeben, für die Nichterfüllung eine 0.

4 Methodik

Der Hauptansatz für die Messung der Komplexität der MIDI-Files bestand wie zuvor erwähnt analog zu Madsen und Widmer [MW15] in dem Entropiemaß nach Shannon.

Dieses ist gegeben durch:

$$E(x) = -\sum_{x \in X} p(x)\, log_2 p(x)$$

Das Maß gibt die negierte Summe des Produkts der Wahrscheinlichkeiten jeder Ausprägung einer bestimmten Zufallsvariable mit dem Logarithmus der jeweiligen Wahrscheinlichkeit an. Analog zur Methodik von Madsen und Widmer wurde die Formel auf die drei musikalischen Dimensionen nach Conklin [Co06] angewandt:

- *Tonklasse*: X ist die Gesamtheit aller Töne unabhängig von ihrer Oktave
- *Tonintervall*: X ist die Gesamtheit aller tonalen Abstände aufeinander folgender Noten
- *Zeitintervall*[7]: X ist die Gesamtheit aller zeitlichen Abstände aufeinander folgender Noten

Der zweite von Madsen und Widmer vorgestellte Ansatz ist die LZW-Kompression, mit der Annahme, dass redundante Strukturen sich stärker komprimieren lassen und somit geringere Komplexität indizieren [MW15]. Der Einsatz dieser Methode entfällt aufgrund der starken Streuung der Spieldauer der untersuchten Werke, welche die Vergleichbarkeit des Komprimierungsgrads verhindern würde. Auch für eine Untersuchung einheitlicher kürzerer Werkausschnitte wäre Kompression als Komplexitätsindikator nicht geeignet, da sie für größere Datenstrukturen optimiert ist.

Neben den Entropiemaßen wurde zusätzlich die „tonale Bestimmtheit“ der Werke gemessen. Diese experimentelle Metrik basiert auf der Funktion `tonalCertainty()` der genutzten

[6] International Music Score Library Project. https://imslp.org/

[7] Auch: Inter-Onset Interval (IOI) != Notendauer (Pausen fließen bei der Notendauer nicht mit in die Berechnung ein, sind aber für die Berechnung der Rhythmik, als Wechselspiel zwischen Ruhe- und Bewegtheitsphasen, essenziell)

Python-Bibliothek music21 und wird in Abhängigkeit der alternativen Interpretationen der Dur-Moll-Tonart des zu analysierenden Musikstücks gemessen [mu20]. Je mehr alternative Tonarten berechnet werden, desto ungewisser ist das Maß.

Die Berechnungen der Tonklassenentropie (*Hpc*) und der „tonalen Bestimmtheit“ (*k*) wurden sowohl für die vollständige Sequenz als auch (trotz hoher Laufzeitkomplexität) für alle 16 Takte ausgeführt. Für die zweite Berechnung wurde das arithmetische Mittel gebildet. Der Hintergrund für letzteres besteht in der bereits erwähnten heterogenen Werklänge und dient dem Zweck der besseren Vergleichbarkeit der Ergebnisse. Besonders für die „tonale Bestimmtheit“ ist die Berechnung des Mittelwerts aller 16 Takte des Stücks von Interesse. Somit soll die Frequenz der Modulation angenähert werden. Sobald Tonarten frequentiert gewechselt werden oder es sich gar um ein chromatisches Stück handelt, könnte von einer hohen harmonischen Komplexität ausgegangen werden.

Während *Hpc* und *k* mit der vollständigen Partitur berechnet werden konnten, da es sich um die gesamtheitliche Betrachtung der Harmoniestruktur handelt, wurden die Berechnungen der Tonintervallentropie (*Hi*) sowie der Zeitintervallentropie (*Hioi*) pro MIDI-Spur vorgenommen. Analog wurde die Entropie für die vollständige Spur als auch das arithmetische Mittel der Entropie aller 16 Takte berechnet. Für die nachfolgende Auswertung wurde die Entropie der Spur mit dem höchsten Entropiewert festgehalten. Um die Berechnung von *Hi* zu ermöglichen, mussten zunächst alle Tonabstände in ihrer komponierten Abfolge vorliegen. Da auch einzelne MIDI-Spuren polyphon sein konnten und die Tonabstände sich nur zwischen zwei aufeinander folgenden Noten bestimmen lassen, wurde eine Heuristik implementiert, bei der zunächst die höchsten Noten der MIDI-Spur bestimmt wurden. Mithilfe dieser Notenlisten, die jeweils für die gesamte Spur, als auch für alle 16 Takte der Spur bestimmt wurden, konnte *Hi* für beide betrachteten Objekte berechnet werden. Die zeitlichen Abstände, die für die Berechnung von *Hioi* notwendig sind, konnten anhand der Subtraktion der Spielzeit jeder Note von der Zeit ihrer nachfolgenden bestimmt werden.

Die berechneten Ergebnisse wurden mitsamt der Angabe zum Titel, Jahr und Länge der Komposition zeilenweise in ein CSV-File geschrieben, welches als Grundlage für die deskriptiv-statistische Auswertung dient. Aufgrund von Fehlern bei der Konvertierung der MIDI-Files in Streams, die mithilfe der Bibliothek music21 behandelt werden können, reduzierte sich die Stichprobe für die weiterführenden Analysen auf 291 Werke.

5 Deskriptiv-statistische Analyse der Merkmalsausprägungen

Die folgende Grafik zeigt die Verteilung der Werke im Korpus pro Komponist und Jahr:

Anhand dieser Darstellung lässt sich feststellen, dass vorallem die Diachronie der Klavier- bzw. Cembalowerke von J. S. Bach, W. A. Mozart, L. v. Beethoven und F. Chopin in dem Korpus nachvollzogen werden könnte. Eine diachrone Analyse der Komplexitätsentwicklung im gesamten Korpus würde geringe Aussagekraft besitzen.

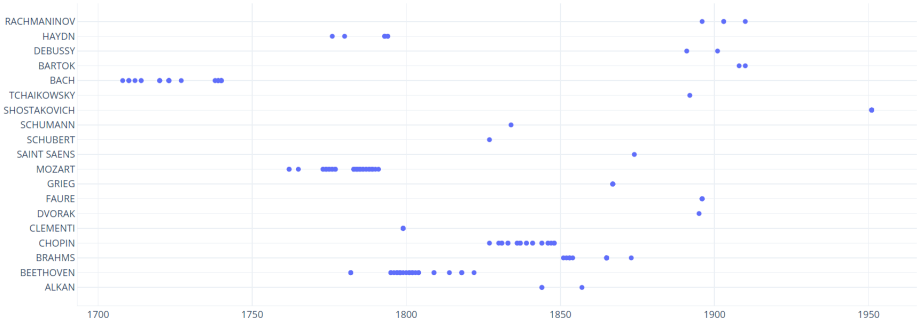


Abb. 1: Werke im Korpus pro Komponist und Jahr

Ein weiterer Aspekt, der bei der Betrachtung der folgenden Ausführungen beachtet werden sollte, ist die Spieldauer der Werke. Die für das vollständige Stück berechnete Komplexität, ist stets unter Einbezug der Spieldauer zu betrachten. Die folgende Grafik zeigt die Verteilung der Spieldauer (in Sekunden) der Werke im Korpus je Komponist:

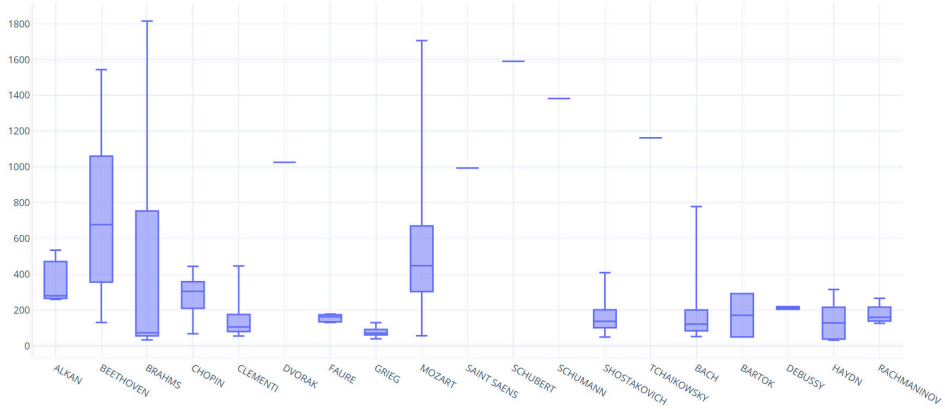


Abb. 2: Spieldauer der Kompositionen pro Komponist

Damit lässt sich erkennen, dass die längste Komposition im Korpus von J. Brahms für eine 30 minütige Spieldauer verfasst wurde. Die größten Spannweiten und Interquartilsabstände der Kompositionsspieldauer sind im Korpus bei L. v. Beethoven, J. Brahms und W. A. Mozart vorzufinden. Hiermit lässt sich aufzeigen, dass die im Korpus vorliegenden MIDI-Files stark heterogen in Bezug auf einzelne Sätze oder vollständige Partituren der Werke vorliegen.

5.1 Harmonische Komplexität der Werke

Zunächst soll die Streuung der harmonischen Komplexität der Werke betrachtet werden, die durch *Hpc* und die tonale Bestimmtheit *k* approximiert wurde. Die folgenden Grafiken zeigen die Streuung der beiden Metriken für das vollständige Stück pro Komponist.

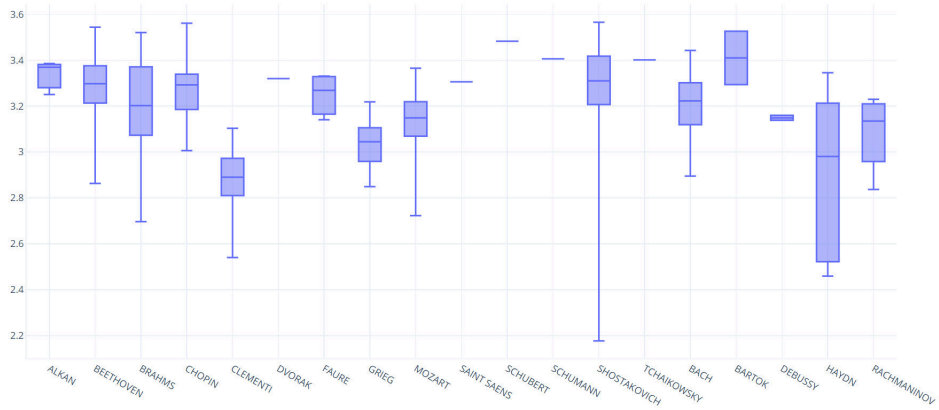


Abb. 3: *Hpc* für die vollständigen Partituren pro Komponist

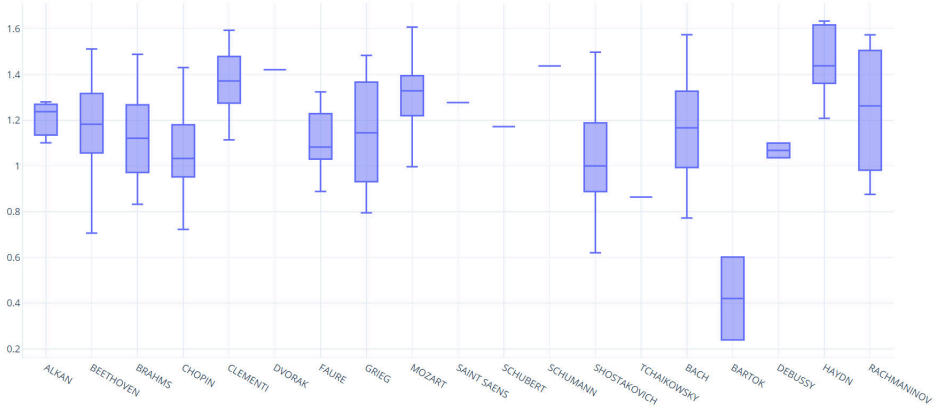


Abb. 4: *k* für die vollständigen Partituren nach Komponist

Die erste Auffälligkeit bei der Betrachtung beider Grafiken besteht darin, dass die beiden Metriken der harmonischen Komplexität sich in einem tendenziell inversen Zusammenhang zueinander verhalten, wie es zu vermuten war. Die Korrelation der beiden Metriken für das vollständige Stück ergibt einen Koeffizient von r=-0.3385, welcher eine schwache negative Korrelation aufzeigt.

Für den Mittelwert der Entropie aller 16 Takte des Stücks ist die Korrelation der beiden

Metriken ebenfalls schwach negativ mit einem Wert von r=-0.2938. Die Metriken für das vollständige Stück korrelieren mit der analogen Metrik für den Mittelwert aller 16 Takte mit r=0.8874 für *Hpc* und für *k* mit r=0.6254 positiv. Somit lässt sich diskutieren, ob die Inkaufnahme der hohen Laufzeitkomplexität für die Berechnung des Mittelwerts aller 16 Takte lohnend ist. Besonders auffällig bei dieser Untersuchung ist die hohe Spannweite der Merkmalausprägungen zur harmonischen Komplexität für die Werke von Shostakovich.

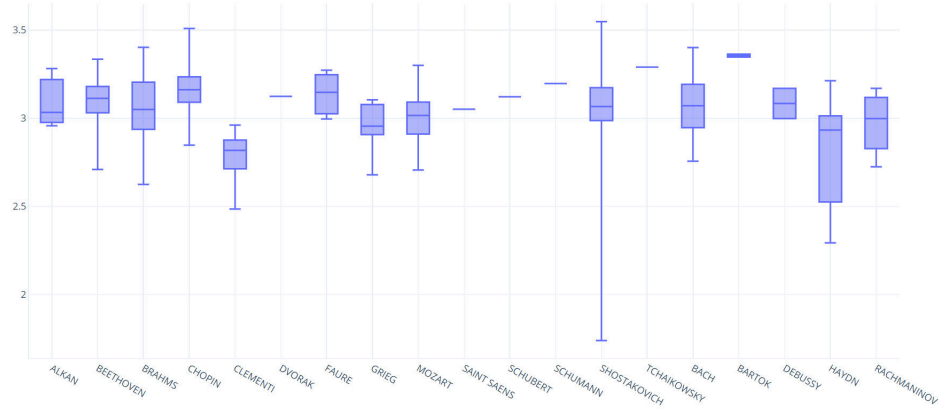


Abb. 5: mittlere *Hpc* aller 16 Takte der vollständigen Partituren nach Komponist

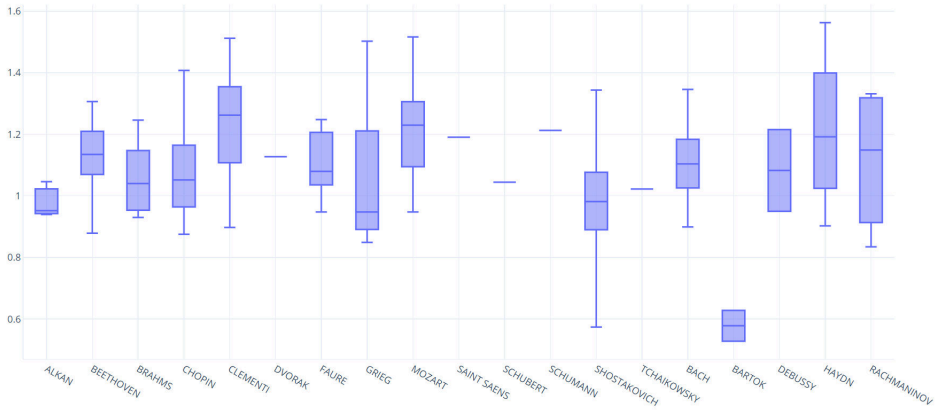


Abb. 6: mittlere *k* aller 16 Takte der vollständigen Partituren nach Komponist

5.2 Melodische Komplexität der Werke

Die melodische Komplexität der Werke wurde mit *Hi* approximiert. Die folgenden Grafiken zeigen die Streuung dieser Metrik für das vollständige Stück und den maximalen Mittelwert aller 16 Takte einer Spur pro Komponist auf.

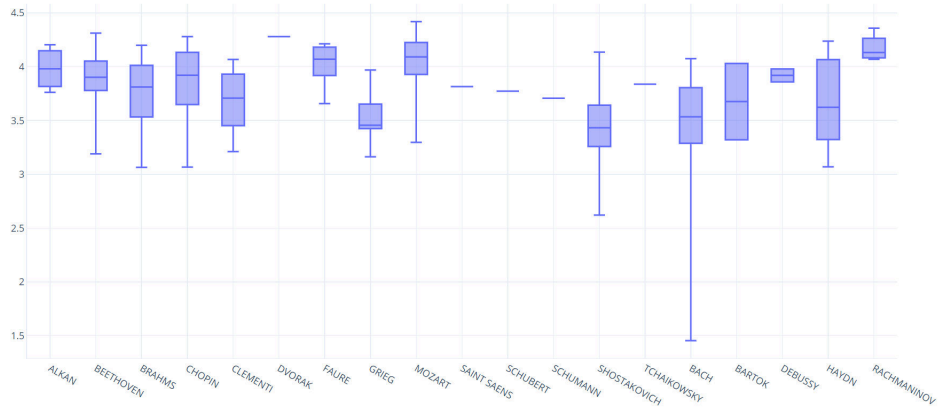


Abb. 7: maximale *Hi* der vollständigen Partituspuren pro Komponist

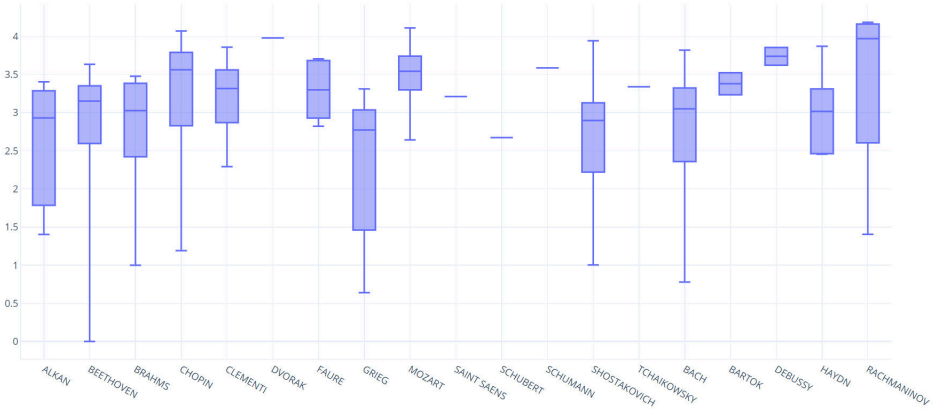


Abb. 8: maximale mittlere *Hi* aller 16 Takte der Partituspuren pro Komponist

Die Metriken zur Bestimmung der melodischen Komplexität korrelieren mit r=0.563 moderat positiv. Das arithmetische Mittel für die Gesamtheit der berechneten Werte beträgt für das vollständige Stück ø=3.728 und liegt damit über dem Wert der zweiten Metrik (ø=3.017). Auffällig ist vorallem die Differenz der Interquartilsabstände der beiden Metriken für die Komponisten Grieg, Rachmaninov und Alkan. Auch die Spannweite der Metriken für die Komponisten L. v. Beethoven, J. Brahms und F. Chopin ist bemerkenswert unterschiedlich.

5.3 Zeitintervall-Komplexität der Werke

Die Zeitintervall-Komplexität der Werke wurde mit *Hioi* approximiert. Die folgenden Grafiken zeigen die Streuung dieser Metrik für das vollständige Stück und den maximalen Mittelwert aller 16 Takte einer Spur pro Komponist aufgezeigt.

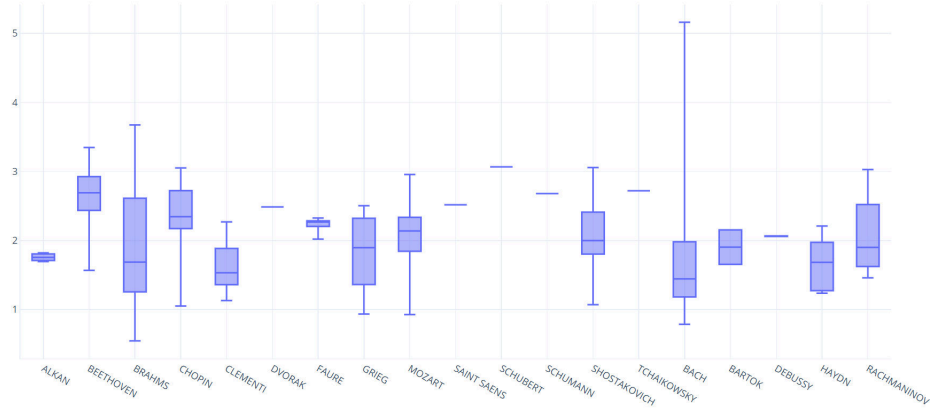


Abb. 9: maximale *Hioi* der vollständigen Partiturspuren pro Komponist

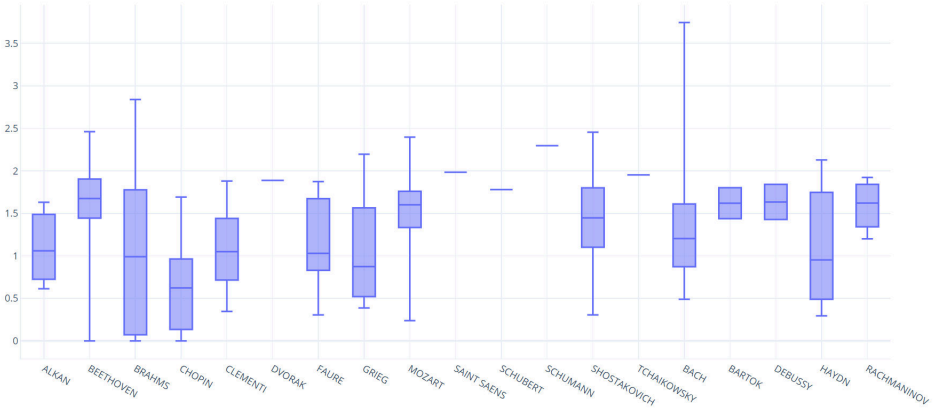


Abb. 10: maximale mittlere *Hioi* aller 16 Takte der Partiturspuren pro Komponist

Die Metriken zur Bestimmung der Zeitintervall-Komplexität korrelieren mit r=0.594 ebenfalls moderat positiv. Das arithmetische Mittel für die Gesamtheit der berechneten Werte beträgt für das vollständige Stück ø=2.111 und liegt damit über dem Wert der zweiten Metrik (ø=1.353). Auffällig ist vorallem die Spannweite der Metriken bei den Werken von J. S. Bach.

5.4 Korrelationsanalyse zwischen den musikalischen Dimensionen

Für das vollständige Stück ist die Korrelation zwischen *Hpc* und *Hi* mit einem Koeffizient von r=0.0511 vernachlässigbar. *Hi* und *Hioi* korrelieren mit r=-0.2764 schwach negativ. *Hioi* und *Hpc* hingegen korrelieren mit r=0.3025 schwach positiv. Für die Berechnung der Mittelwerte aller 16 Takte ist die Korrelation zwischen *Hpc* und *Hi* mit einem Koeffizient von r=0.0828 vernachlässigbar. *Hi* und *Hioi* korrelieren mit r=0.1031 ebenfalls nicht signifikant positiv. *Hioi* und *Hpc* korrelieren mit r=0.1888 schwach positiv.

6 Fazit und Ausblick

Das Entropiemaß nach Claude E. Shannon ermöglichte einen Einblick in die Verteilung der Komplexität und Gewichtung unterschiedlicher musikalischer Dimensionen der Klaviermusik europäischer Komponisten im Zeitraum 1700-1950. Durch die zusätzliche Betrachtung der negativen tonalen Bestimmtheit konnten zudem Abweichungen vom Dur-Moll-System aufzeigt werden, die indikativ für einen befremdlichen Klang eines Musikstücks sein können. Problematisch zeigte sich hingegen die Datenakquise, da ein den Ansprüchen dieser Arbeit genügender Korpus nur äußerst zeitaufwendig konstruiert werden konnte. Die Datenqualität der Analysen leidet somit an dem geringen Stichprobenumfang. Grenzen dieser Arbeit liegen in ihrem stark explorativen Ansatz. Die Limitationen der Studie liegen weitestgehend in dem geringen Stichprobenumfang sowie der schwer nachprüfbaren Korrektheit der analysierten Partituren. Hinzu kommt die heterogene Spieldauer der Kompositionen. Weiterhin bleibt anzumerken, dass das für die Tonintervallentropie *Hi* und die Zeitintervallentropie *Hioi* die werkglobal bewegteste Spur bestimmt wurde, jedoch die lokale Komplexität, wie sie von Madsen und Widmer [MW15] ebenfalls postuliert wurde, für die Untersuchung interessanter gewesen wäre. Dabei sollte für jedes Zeitfenster die bewegteste Spur in die Analyse einfließen, da diese in der Theorie, die höchste Aufmerksamkeit des Hörers genießt. Jedoch hätte dies zu einer noch höheren Laufzeitkomplexität geführt, welche in ihrer aktuellen Implementierung ohnehin bereits die Extraktion der Merkmalsausprägungen erschwerte.

Zukünftig wäre auch die Analyse längerer struktureller Redundanzen innerhalb eines Werks von Interesse, wofür aktuell eine geeignete Metrik fehlte. Weiterhin wäre die Betrachtung des Auftretens von musikalischen Strukturen im gesamten Zeitraum von Interesse, wie Sie für Tonintervalle bei Simonton [Si84] erfolgte. Demnach müssten besonders selten auftretende musikalische Strukturen die Komplexität eines Werks zusätzlich zu einer geringen werksinternen Redundanz steigern, da sie für den Hörer befremdlicher und somit komplexer erscheinen würden. Weiterhin könnte eine Korrelationsanalyse der Entropiemaße mit dem Schwierigkeitsgrad nach G. Henle zum Zwecke der Erforschung einer Interdependenz zwischen der wahrgenommenen Komplexität eines Musikstücks

und dessen Spielschwierigkeit durchgeführt werden sowie eine Studie zum Vergleich der Ergebnisse mit qualitativen musikwissenschaftlichen Analysen zu Einzelwerken.[8]

Literaturverzeichnis

[Co06] Conklin, D.: Melodic analysis with segment classes. Machine Learning, 65(2):349–360, 2006.

[GB03] Goebl, W.; Bresin, R.: A complexity-based approach to melody track identification in MIDI files. 2003.

[MI] Standard MIDI-File Format Spec. 1.1.

[mu20] tonalCertainty(). https://web.mit.edu/music21/doc/moduleReference/moduleKey.html. 2020-05-10.

[MW15] Madsen, S.; Widmer, G.: A complexity-based approach to melody track identification in MIDI files. 2015.

[Si84] Simonton, D. K.: Melodic Structure and Note Transition Probabilities: A Content Analysis of 15,618 Classical Themes. Psychology of Music, 12(1):3–16, 1984.

[Yo58] Youngblood, J. E.: Style as Information. Journal of Music Theory, 2(1):24–35, 1958.

[8] Das Skript für die Extraktion der Merkmalausprägungen und weitere Ergebnisartefakte wie die Evaluation der MIDI-Archive sind unter folgender URL verfügbar: https://github.com/MihailMiller/music-complexity

Sicherheit

Asynchronous and Decentral Group Management in Messengers with Delegated Proof of Stake

Andreas Hellenbrand[1]

Abstract: Mobile messaging applications are used widely for group communication using group chats. Most messenger platforms rely on their centralized infrastructure to maintain the group states. This can imply privacy issues and allow potential misuse by the messenger providers. To resolve this privacy implications, a decentral approach can be implemented. The decentral protocol presented in this work is based on the Delegated Proof of Stake consensus protocol and uses a blockchain to store the groups state. The main focus of this work is the optimization of the protocol to be able to deal with the asynchronous environment of mobile applications.

Keywords: Blockchain; DPoS; Group Chats

1 Introduction

Group Chats are an essential feature of mobile messenger applications, such as WhatsApp, Signal and Telegram. In order to add or remove members from a group, management is needed. Most messenger systems rely on centralized management by storing the groups state gr on the system server. [2]

$$gr = (id, \mathcal{M}, \mathcal{M}^*, info) \tag{1}$$

Table 1 defines the components of the four tuple introduced in equation 1.

id	:= Unique group identifier
$\mathcal{M}$	:= Set of members/users
$\mathcal{M}^*$	:= Set of admins with $\mathcal{M}^* \subseteq \mathcal{M}$
$info$	:= Meta information (e.g. name, icon, description, ...)

Tab. 1: Group state

Signal recently published a paper that introduced an end-to-end encrypted, centralized group management [CPZ19], ditching their previously used decentral system [Ma14]. Other systems store the group information unencrypted, as they distribute group messages from

[1] Hochschule RheinMain, Wiesbaden, Germany andeas.hellenbrand@student.hs-rm.de

[2] The notation of equation 1 was introduced by Rösler et al. in their analysis of group messaging protocols [RMS17].

cc by sa

the server [Te; Wh17]. In Signal group messages are processed like direct messages, such that the server does not need to know the groups members.

The unencrypted, central storage of group information violates the user's privacy. Even in Signals encrypted handling of group information, the existence of a group is disclosed to the service and may allow the server to return incorrect information about the group [CPZ19].

The goal of this work is to provide a decentral group management system, that can be used in the asynchronous environment of mobile messengers, where clients are not always connected and the users only open the application from time to time.

1.1 Delegated Proof of Stake

Delegated Proof of Stake (DPoS) is a consensus algorithm used for the blockchains of EOS and BitShare [Bi18; EO18]. Compared to Proof of Work systems like Bitcoin [Na09], DPoS does not rely on a competition to find hashes which requires a high computational effort. Instead all stakeholders (e.g. the coin holders) can elect delegates, who then can build and distribute new blocks. These blocks contain transactions and votes and are authenticated by the signature of a currently elected delegate. Every x blocks, the votes are recounted and the set of delegates is updated if needed [Bi].

The order in which the delegates sign new blocks is defined by a fixed or randomized schedule and all delegates who share a valid block in their assigned time slot are rewarded with new tokens [EO18].

1.2 Related Work

In [He19] a group management system based on the DPoS protocol was presented. Instead of relying on a central service to maintain and distribute the state gr, group members share messages between each other to manage the group.

In this work, several requirements for a group management system where defined.

Same State All members m in a group with state gr use the same local state gr_m.

$$\forall m \in \mathcal{M} \quad (gr_m = gr)$$

Confidentiality by Group Management Only admins $m \in \mathcal{M}^*$ should be able to update the group state gr (e.g. add or remove members).

Privacy Only the members $m \in \mathcal{M}$ of a group should know that the group exists.

Each group is defined by its own blockchain that is distributed between the members. The members of the group elect delegates, which are able to sign suggestions. Suggestions, such as *add* or *remove* member and *info*, can be shared by all members and are confirmed if an elected delegate shares a new block, including the suggestions. If no delegate builds such a block, the suggestion is ignored and not applied to the group. Suggestions can be seen as the transactions of traditional DPoS systems, beside being optional to confirm.

[He19] describes several shortcomings regarding the chosen voting process to elect delegates and the handling of forks. Members can start *Voting Windows* in which every member can share votes if they get online during this time frame. This implies that members which did not get online are excluded from the vote, as a synchronous component is introduced in the asynchronous environment of messenger applications. After collecting all votes, the members need to confirm to each other, that they computed the same election result. This introduces a second step and increases the protocols complexity. Forks are resolved by dropping the branch that does not get continued, this leads to information loss and violates the *same state* requirement for the time the fork exists.

The presented protocol also does not take delivery issues like delayed or lost messages into account. As these are expected in an asynchronous mobile environment, the group management protocol needs to be resilient against delivery issues.

This initial work also contains a security analysis of the protocol and an introduction to alternative solutions that are used by messenger applications.

2 Group Management with DPoS

The following section introduces an updated version of the protocol with focus on the new voting and forking system.

2.1 Notation

Table 2 shows some additional notations, which will be used in the following section.

More on Merkle trees can be found in [Me88].

2.2 Protocol

The blockchain $\mathcal{B}_{gr}$ that is distributed between the members $\mathcal{M}_{gr}$ describes the current state of a group $gr = (id, \mathcal{M}_{gr}, \mathcal{M}^*_{gr}, info)$. If a new block B_S with a suggestion S is added to $\mathcal{B}_{gr}$, the group state is updated according to the action a of S ($gr' = Update(gr, a)$).

$$\mathcal{B}_{gr} + B_S \rightarrow gr' = Update(gr, a) \tag{2}$$

$[x_1, ..., x_n]$	List of the items x_1 to x_n
$\mathcal{B}_{gr}$	The Blockchain of the group gr
B_{-1}	The latest block of a Blockchain $\mathcal{B}_{gr}$
$hash(x) \rightarrow$ byte[]	Hash for x
$sig(u_{id}, x) \rightarrow$ byte[]	Signature for x from signer u_{id}
$verify(u_{id}, sig, data) \rightarrow$ boolean	Verifies a Signature
$MerkleRoot([x_1, ..., x_n]) \rightarrow$ byte[]	Merkle root of a Merkle tree over the list $[x_1, ... , x_n]$
$MerklePath(x_i, [x_1, ..., x_n]) \rightarrow$ [byte[]]	Merkle Verification Path to the leaf x_i with $1 \leq i \leq n$ of a Merkle tree over the list $[x_1, ..., x_n]$
$verify(x_i, mvp) \rightarrow$ boolean	Verifies a Merkle path

Tab. 2: Additional Notation

2.2.1 Suggestion

A suggestion S can be shared by any member $m \in \mathcal{M}$ at any time.

$$S = (u_{id}, action, value, b_{id}, b_{hash}, sig) \tag{3}$$

Table 3 defines the components of the suggestion tuple introduced in equation 3.

	Definition	**Comment**
u_{id}	$:= m \in \mathcal{M}$	Sender
$action$	$:= add \parallel remove \parallel info$	The action to perform
$value$	$:= u_{id} \parallel info$	The user or new *info*
b_{id}, b_{hash}	:= Block Reference	Block id and the blocks hash
sig	$:= sig(S \setminus sig)$	Signature

Tab. 3: Suggestion

Each suggestion includes a block reference (b_{id}, b_{hash}), which links it to a specific block in $\mathcal{B}_{gr}$. This prevents the suggestions to be included in a different blockchain and allows the members to check the assumptions the sender makes about the current groups state gr. This mechanism is known as Transactions as Proof of Stake (TaPOS)[EO18].

If a suggestion is applied, its action updates the group according to table 4.

(action, value)	**Update**
(add, u_{id})	$\mathcal{M}' = \mathcal{M} \bigcup \{u_{id}\}$
$(remove, u_{id})$	$\mathcal{M}' = \mathcal{M} \setminus \{u_{id}\}$
$(info, i)$	$info' = i$

Tab. 4: Suggestion actions

A suggestion S is valid for the blockchain $\mathcal{B}$ with the latest block B_{-1} and can be confirmed

if the conditions in table 5 hold. Suggestion can expire if they reference a block, that is to far down the chain.

$a = remove \wedge value \in \mathcal{M}$	Only remove members
$a = add \wedge value \notin \mathcal{M}$	Only add users that are not already a member
$\exists b \in \mathcal{B} \quad (hash(b) = b_{hash} \wedge b.b_{id} = b_{id})$	Referenced block is in $\mathcal{B}$
$b_{id} \geq B_{-1}.id - n$	Referenced block is not older than n blocks
$verify(u_{id}, sig, S \setminus sig)$	Valid Signature

Tab. 5: Valid Suggestion

2.2.2 Block

A suggestion block B_S can be shared by the delegates $m \in \mathcal{M}^*$. As soon as a delegate gets online and receives a suggestion S he likes to confirm, he can sign and share a block, as long as S is not already confirmed by a block from another delegate. Unlike a traditional DPoS systems, a specific order does not need to be followed by the delegates. This allows the group management to work in the asynchronous messenger environment as it is not requierd for them to be online at a specific time.

$$B_S = (b_{id}, pbh, S, sig, u_{id}, MVP, VMR, nVMR, votes) \tag{4}$$

Table 6 defines the components of B_S introduced in equation 4.

	Definition	**Comment**
b_{id}	$:= B_{-1}.b_{id} + 1$	Latest $b_{id} + 1$
pbh	$:= hash(B_{-1})$	Hash of the previous/latest block
S	:= Suggestion	Confirmed suggestion
u_{id}	$:= m \in \mathcal{M}^*$	Delegate / Signer
MVP	$:= MerklePath(u_{id}, \mathcal{M}^*)$	Delegate proof
VMR	$:= MerkleRoot(\mathcal{M}^*)$	Delegate proof
$nVMR$	$:= MerkleRoot(\mathcal{M}^{*'})$	Merkle root of $\mathcal{M}^*$ with the new *votes*
$votes$	:= [vote]	List of new votes
sig	$:= sig(u_{id}, Block \setminus sig)$	Block signature

Tab. 6: Block

Delegate Proof When building and sharing a new block, the block signer u_{id} needs to proof that he is indeed a currently elected delegate ($u_{id} \in \mathcal{M}^*$). To do so he creates a *delegate proof* with the Merkle root *VRM* and a Merkle verification path *MVP* from the Merkle tree over $\mathcal{M}^*$. The *VRM* can be used to verify that the block signer used the correct set of delegates and by providing a valid *MVP* from his u_{id} to the *VRM*, he can proof that $u_{id} \in \mathcal{M}^*$.

The *nVRM* is used to update $\mathcal{M}^*$ if the new votes change the election result. This process is explained in section 2.3.

Confirmation Blocks S can also be empty. Such a block is called *Confirmation block*, as it confirms the current group state *gr* without updating it. These blocks are shared by the delegate that first gets online after a time x since the last block and no new suggestions are available for confirmation, to add new votes to the blockchain.

Block Verification After a block B is received, each member validates the block in relation to the latest/previous block $B_{-1} \in \mathcal{B}$. The block is accepted if the conditions from table 7 hold:

Condition	Comment
$b_{id} = B_{-1}.b_{id} + 1$	Latest $b_{id} + 1$
$pbh = hash(B_{-1})$	Latest hash matches
$Valid(S)$	Valid suggestion
$verify(u_{id}, sig, B \setminus sig))$	Valid signature
$verify(u_{id}, MVP)$	Valid Merkle path
$VMR = B_{-1}.\text{nVRM}$	Merkle root matches the latest *nVMR*
$nVMR = MerkleRoot(\mathcal{M}^{*'})$	Next Merkle root is for the update $\mathcal{M}^*$
$\forall v \in votes \quad (Valid(v))$	All votes are valid

Tab. 7: Block verification

2.3 Elections

Votes can be sent by every member $m \in \mathcal{M}$ at every time.

$$Vote = (u_{id}, \mathcal{V}_{u_{id}}, sig, b_{id}, b_{hash}) \tag{5}$$

Table 8 defines the components of a *Vote* introduced in definition 5.

	Definition	Comment
u_{id}	$:= m \in \mathcal{M}$	Sender of the Vote
$\mathcal{V}_{u_{id}}$	$:= [v_1, \ldots, v_n]$ with $v_i \in \mathcal{M}$	The votes (List of users)
sig	$:= sig(u_{id}, Vote \setminus sig)$	Signature
b_{id}, b_{hash}	$:= B_{-1}.b_{id}, hash(B_{-1})$	Block reference

Tab. 8: Vote

Like suggestions, votes also include a block reference as TaPOS.

A shared vote must be added to the next block a delegate builds. Unlike suggestions, votes must not be ignored. For each block B_S that includes votes, the signing delegate needs to compute the new election result. First the last votes are retrieved from the blockchain, including the new block ($\mathcal{B} + B_S$). Each users votes $\mathcal{V}_i = [v_1, \ldots, v_n]$ are stored in the set $\mathcal{V} = [\mathcal{V}_{u_1}, \ldots, \mathcal{V}_{u_m}]$.

A vote V is valid for the blockchain $\mathcal{B}$ with the latest block B_{-1} and can be confirmed if the conditions in table 9 hold.

Condition	**Comment**
$\exists b \in \mathcal{B} \quad (hash(b) = b_{hash} \wedge b.b_{id} = b_{id})$	Referenced Block is in $\mathcal{B}$
$b_{id} \geq B_{-1} - n$	Referenced block is not older than n blocks
$verify(u_{id}, sig, Vote \setminus sig)$	Valid Signature
$\forall v \in \mathcal{V}_{u_{id}} \quad (v \in \mathcal{M})$	Voted users are in $\mathcal{M}$

Tab. 9: Valid Vote

In the next step, for each member u_{id} the received votes are computed using equation 6. Each user has one vote. If he chooses to vote for multiple users, the vote is distributed equally between his choices ($\frac{1}{|\mathcal{V}_i|}$).

$$x_{u_{id}} = \sum_{i=0}^{|\mathcal{V}|} \begin{cases} \frac{1}{|\mathcal{V}_i|} & \text{, if } u_{id} \in \mathcal{V}_i \\ 0 & \text{, else} \end{cases} \qquad \text{for all } u_{id} \in \mathcal{M} \tag{6}$$

nVRM of the new block is set to the Merkle root of the list of u_{id}'s of the $|\sqrt{\mathcal{M}}|$ members with the highest votes $x_{u_{id}}$. If $nVRM \neq VRM$ this step updates the set of delegates $\mathcal{M}^*$.

2.4 Forks

Forks can occur if delegates share new blocks that are linked to the same previous block. In this case the blockchain opens into multiple branches. Most systems resolve forks by only using the longest chain, e. g. the branch that gets continued first. This approach is not suitable for group management, as it can temporarily break the *same state* requirement as members are on different branches.

Forks can be prevented if messages get a *Server-Side-Timestamp* (SST). Members then can compare which block was shared first and only use this block. In figure 1 the block 1339 with timestamp 13:45 gets chosen over the block with 13:46.

The SST is attached to every message m by the messaging services server ($m' = (m, SST)$). It's assumed that the connection between the users and the server is protected.

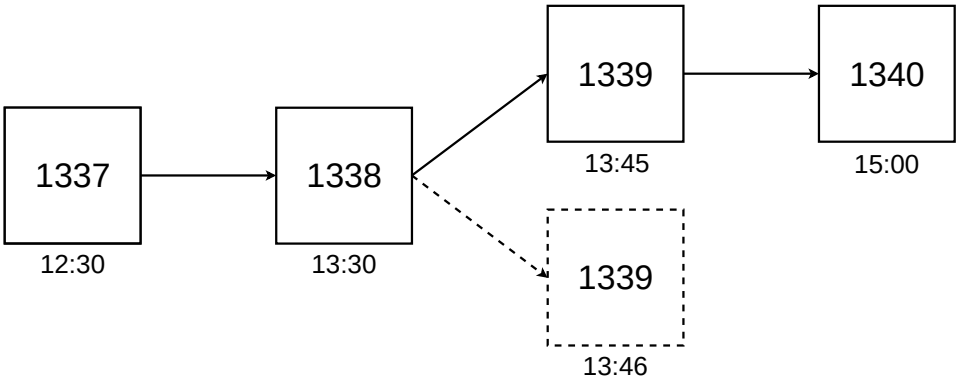


Fig. 1: Fork with *Server-Side-Timestamp*

3 Asynchronous Messaging

In the mobile environment of messenger applications it can occur that messages are not received or received in the incorrect order. A distributed group management system needs to compensate these cases.

The correct recpetion of messages is only critical for block messages, as only these messages directly affect the state. Suggestion and vote messages can be either confirmed by a delegate that received the message or need to be resent.

A lost or delayed block can be detected if another block with a b_{id} greater than the latest local $b_{id} + 1$ is received. Such blocks need to be stored in a local cache. If the missing block is received (due to a delay), the cached block can be added to the chain after the delayed block. If no block is received after a time X, the member sends a $Sync(b_{id})$ to a random set of members $M^{'} \subseteq \mathcal{M}$, which reply with the blocks $\{b_{id}, ..., b_{latest}\} \subset \mathcal{B}$. The requesting member then compares the responses and rebuilds his local blockchain accordingly.

4 Example

Figure 2 shows several messages to a group, and how the votes, suggestions and blocks affect the group state.

Incoming messages to the group $(u_{id}, type, content)$ can be of type Suggestion S, Vote V or Block B. For simplication, some fields of the messages are left out. New blocks are appended to the blockchain and the state is updated accordingly. The state contains the group name, the member list and the election results $(u_{id} : recieved\ votes \quad (own\ votes))$. The elected delegates are color coded together with the *VMR* and *nVRM*.

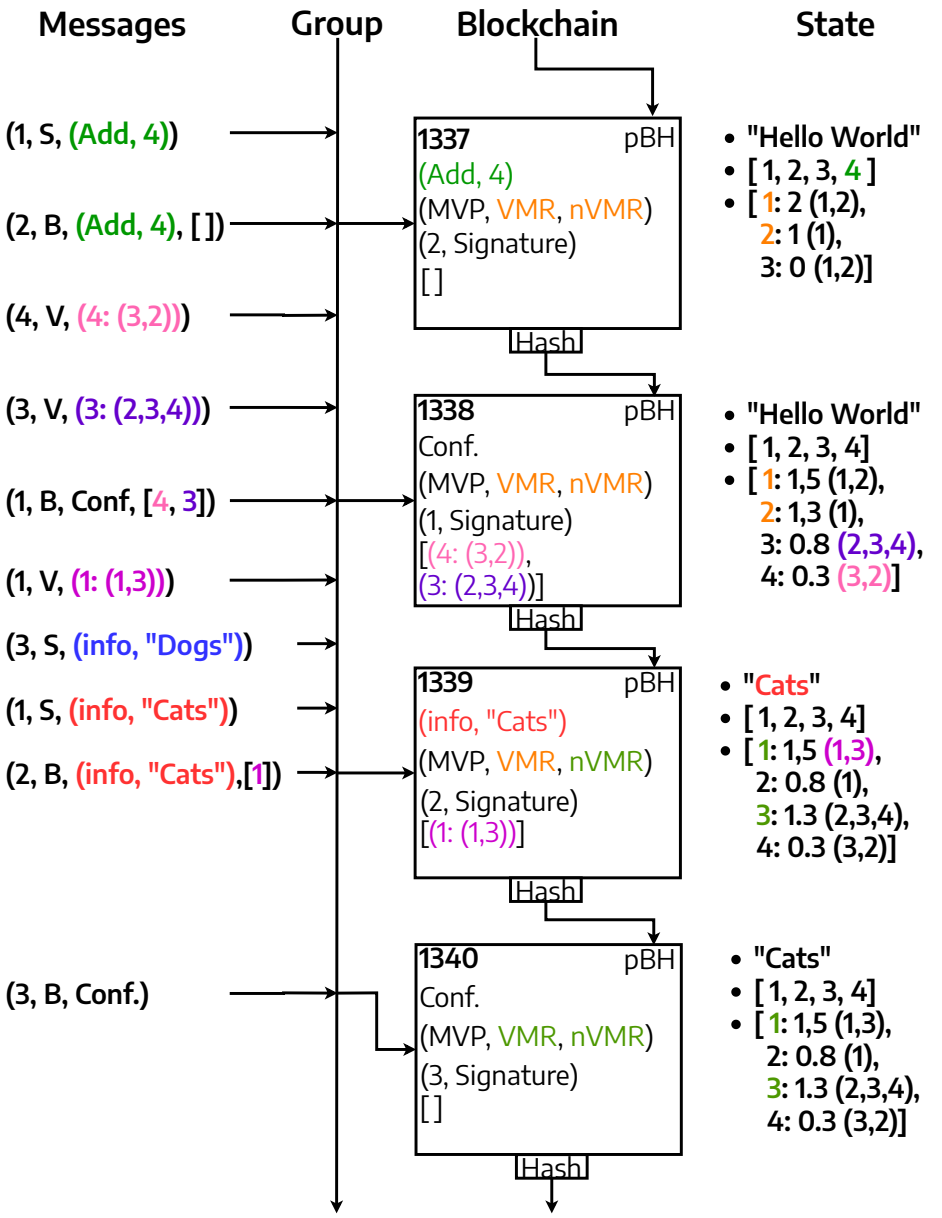


Fig. 2: Example Group

5 Security Considerations

As [He19] covers the security analysis of the initial protocol, this work only focuses on the parts affected by the updates introduced above.

Two adversaries are described, *malicious server* and *malicious member*. *Malicious user* and *network attacker* are not covered, as the attack surface for these adversaries is not affected by the updated protocol.

5.1 Malicious Server

A malicious server attacks the connection between client and server or obtains control over the server. This adversary can modify and forge all communication that is only protected on the transport layer or drop messages.

The server-side-timestamp, used for fork prevention (2.4), can be altered by such an adversary. As the message content is assumed to be end-to-end encrypted, the malicious server is not able to target a specific group, as it cannot be distinguished between one-to-one and group management messages. The server can either run a general attack and set invalid timestamps on all messages from an user (this can be prevented by using Signals Sealed Sender technology [Lu18]) or on all message towards a specific user. On the later, the user can be tricked to add the wrong block. With the receival of the next management message referencing the latest block, this can be detected and the user can request a sync (3) to recover from the attack.

5.2 Malicious Member

A malicious user is a user with knowledge of the protocol and that is a member of the group he wants to attack.

[He19] described an attack where this adversary could use the forking mechanism to drop and override the latest block. This involved creating a fork and directly extending the own, malicious, branch. In the updated protocol this attack is no longer possible, as forks can not be triggered by a delegate.

Malicious member can still send different messages for blocks, votes and suggestions to different members of the group. As new blocks cannot be added after such a forged block is received, the member would request a sync. As the different blocks send by the attacker are visible trough commparing the received chains, this attack can be detected and the group can react trough a rollback to the latest valid state. As for votes and suggestions, this attack is not an issue, as only the version of the message added to the blockchain is relevant.

6 Results and Conclusion

The updated protocol is implemented and tested using a dotnet core application, that simulates a group chat. Multiple instances connect to a RabbitMQ message server [3], which forwards the messages and acts as the messenger service. Asynchronicity is simulated by random sleep times of the clients. Delivery issues are also simulated by the clients through randomly dropping or delaying incoming messages. To keep the group *alive*, random suggestions or votes are sent if the client gets online/wakes up.

In this simulated environment the described protocol was able to maintain a stable group. The updated protocol solves the described shortcomings and introduces a mechanism to deal with message delivery issues. However this results allow no assumptions about real world performance, as the results strongly depend on the parameters chosen for the simulation. E.g. the range of random sleep times together with the probability of sending and confirming suggestions affect the average block times. The probability assigned to each suggestion type, such as adding or removing members, affects if the group grows or stays stable in size. The amount of dropped and delayed messages affects the group stability as clients often needs to request syncs from other members, which might also be in an uncertain state.

The new voting system is less complex and more resilient to delivery issues. It also provides a better representation of the users votes, as they do not need to be online at a specific time in order to share their vote and the set of delegates is directly updated if needed.

While an external *clock* resolves the forking issue, the SST might not be a suitable solution, as every message needs to be touched. Due to the end-to-end encryption, the server cannot only *tag* block messages. This increases the load on the delivery system. This could be resolved by a more sophisticated system, where delegates can anonymously request a signed timestamp from the service. This timestamp can than be included in the block before sending.

7 Outlook

In further work, the properties of the proposed protocol can be compared to existing solutions, to figure out how the decentral approach performs regarding scalability, stability and security of group chats.

Finally the protocol can be integrated in an open source messenger like Signal, to test it under real world conditions.

[3] `https://www.rabbitmq.com/`

References

[Bi] BitShares: Delegated Proof of Stake (DPOS), Accessed July 18, 2020, URL: https://docs.bitshares.org/en/master/technology/dpos.html.

[Bi18] BitShares: BitShares Whitepaper, Accessed July 18, 2020, 2018, URL: https://github.com/bitshares-foundation/bitshares.foundation/blob/master/download/articles/BitSharesBlockchain.pdf.

[CPZ19] Chase, M.; Perrin, T.; Zaverucha, G.: The Signal Private Group System and Anonymous Credentials Supporting Efficient Verifiable Encryption, Cryptology ePrint Archive, Report 2019/1416, 2019, URL: https://eprint.iacr.org/2019/1416.

[EO18] EOS: TechnicalWhitePaper, Accessed July 18, 2020, 2018, URL: https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md.

[He19] Hellenbrand, A.: Decentral Group Management in Messengers with Delegated Proof of Stake, 2019, URL: https://github.com/AndHell/DPoSGroupchatManagement/blob/master/BachelorThesis/thesis.pdf.

[Lu18] Lund, J.: Technology preview: Sealed sender for Signal, Accessed Juni 18, 2020, 2018, URL: https://signal.org/blog/sealed-sender/.

[Ma14] Marlinspike, M.: Private Group Messaging, Accessed July 18, 2020, 2014, URL: https://signal.org/blog/private-groups/.

[Me88] Merkle, R. C.: A Digital Signature Based on a Conventional Encryption Function. In (Pomerance, C., ed.): Advances in Cryptology — CRYPTO '87. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 369–378, 1988, ISBN: 978-3-540-48184-3.

[Na09] Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system, 2009, URL: http://www.bitcoin.org/bitcoin.pdf.

[RMS17] Rösler, P.; Mainka, C.; Schwenk, J.: More is Less: On the End-to-End Security of Group Chats in Signal, WhatsApp, and Threema, Cryptology ePrint Archive, Report 2017/713, 2017, URL: https://eprint.iacr.org/2017/713.

[Te] Telegram: Telegram FAQ, Accessed July 18, 2020, URL: https://telegram.org/faq.

[Wh17] WhatsApp: WhatsApp Security Whitepaper, Accessed July 18, 2020, 2017, URL: https://www.whatsapp.com/security/WhatsApp-Security-Whitepaper.pdf.

Games

Vergleich verschiedener Algorithmen für ein N-Spieler schachähnliches Brettspiel

Hannes Dröse[1]

Abstract: Die Implementierung von Computergegnern für N-Spieler-Spiele (mit mehr als zwei Spielern) stellt eine besondere Herausforderung dar. Für die Lösung gibt es eine Vielzahl verschiedener Algorithmen und Varianten. Daher werden in diesem Paper ein paar typische Algorithmen mit einander verglichen. Dabei handelt es sich um den Max^{N} (mit verschiedenen Pruning Strategien), den paranoiden Minimax und den Hypermax-Algorithmus. Alle Algorithmen werden implementiert und ihre Effektivität wird anhand des Brettspiels "Chamäleon Schach" auf die Probe gestellt.

Keywords: AI; Computergegner; Brettspiel; N-Spieler; MaxN; Paranoid-Minimax; Hypermax; Pruning

1 Einleitung

1982 hat Wolfgang Großkopf, der Großvater des Autors, ein Spiel namens "Chamäleon" entwickelt. Es handelt sich dabei um eine interessante Schach-Variante. Das Spiel ist mehrmals verlegt worden, u.a. 1990 und 1992 vom VSK Verlag. 1991 belegte es sogar den siebten Platz im bekannten Wettbewerb "Spiel des Jahres".

Nun ist eine Neuauflage in digitaler Form unter dem Namen "Chamäleon Schach" geplant, die ebenfalls einen möglichst starken Computergegner enthalten soll. Das Spiel kann nicht nur zu zweit, sondern auch zu dritt oder zu viert gespielt werden. Dadurch entstehen besondere Herausforderungen für die Implementierung des Computergegners. Deswegen werden in diesem Paper verschiedene Algorithmen zur Lösung dieser Aufgabe präsentiert, analysiert und miteinander vergleichen.

2 "Chamäleon Schach"

2.1 Spielprinzip

Zum besseren Verständnis wird hier kurz das Spielprinzip von "Chamäleon Schach" erklärt.

[1] Fachhochschule Erfurt, Altonaer Str. 25, 99085 Erfurt, Deutschland hannes.droese@fh-erfurt.de

cc by sa

Gespielt wird auf einem speziellen Schachbrett mit 8x8 Feldern. Anders als beim klassischen Schach haben die Felder aber vier unterschiedliche Farben (rot, grün, gelb und blau). Die Farben sind nach einem vorgegebenen Muster über das Brett verteilt.

Die Spielfiguren sind sozusagen Chamäleons (also gestaltwandelnde Echsen). Denn je nach der Farbe des Feldes, auf dem sich eine Figur befindet, hat die Figur eine andere Schach-Rolle. So ist eine Figur auf einem roten Feld vielleicht ein Turm, auf einem grünen Feld aber ein Springer. Es kommen vier verschiedene Schach-Rollen vor: Springer, Königin, Läufer und Turm. Die Zuordnung von Feldfarbe zu Rolle ist je nach Figur unterschiedlich.

Jeder Spieler startet mit vier Figuren. Wer am Zug ist, wählt eine seiner Figuren aus und bewegt sie entsprechend der Rolle (z.B. ein Turm kann nur horizontal oder vertikal ziehen). Wird ein Feld erreicht, auf dem eine gegnerische Figur steht, kann diese geschlagen werden. Es besteht aber kein Schlagzwang. Danach ist der nächste Spieler an der Reihe.

Ziel ist es alle gegnerischen Figuren zu schlagen und damit der einzige Überlebende zu sein.

2.2 Spielanalyse

"Chamäleon Schach" ist ein Spiel mit perfekter Information, da es keine Elemente gibt, die nicht allen Spielern bekannt sind. Alle Informationen zum Spiel können mit einem Blick auf das Spielbrett gewonnen werden. [vgl. LR57, s. 58f]

Es gibt immer einen klaren Gewinner, daher machen Kooperationen wenig Sinn. Das Spiel ist strikt kompetitiv und damit ein Nullsummenspiel bzw. allgemeiner gefasst: ein Spiel mit konstanter Summe. [vgl. LR57, s. 164]

Nach der bisherigen Analyse wirkt dieses Brettspiel dem klassischen Schach sehr ähnlich. Für Spiele dieser Art gibt es eine klare Strategie zur Erstellung eines Computergegners: der Minimax-Algorithmus mit Alpha-Beta-Pruning. Die Tatsache, dass "Chamäleon Schach" aber nicht nur zu zweit, sondern auch zu dritt oder zu viert gespielt werden kann, verändert die Sachlage. Der Minmax-Algorithmus funktioniert nur bei Zwei-Spieler-Spielen. Für Mehrspieler-Spiele gibt es verschiedene Ansätze und keinen klar überlegenen Algorithmus, sodass in so einem Fall immer abgewogen werden muss, welcher Algorithmus infrage kommt. Ein direkter Vergleich der Algorithmen gegeneinander ist also nötig. [vgl. St03, s. 2f]

3 Algorithmen

3.1 Grundprinzip

Das Grundprinzip für Computergegner in Brettspielen ist eigentlich immer das gleiche. Es gibt eine heuristische Bewertungsfunktion, mittels der Spielzustände bewertet werden

können. Der Computer generiert also, von der aktuellen Situation ausgehend, die möglichen Züge und das aus dem jeweiligen Zug resultierende Spielbrett. Nun werden diese Spielzustände mit der Bewertungsfunktion bewertet und der Computer wählt den besten Zug aus und führt ihn durch. [vgl. St03, s. 26]

Diese heuristischen Bewertungsfunktionen sind immer sehr individuell vom Spiel abhängig. Sie versuchen, möglichst präzise festzustellen, welcher Spieler gerade die Oberhand hat. [vgl. St03, s. 27] Für dieses Paper nehmen wir die Bewertungsfunktion als gegeben an. Es sollen nur die darauf aufbauenden Algorithmen miteinander verglichen werden.

Die Effektivität der Bewertung kann gesteigert werden, indem nicht nur ein, sondern mehrere Züge in die Zukunft geschaut wird. Das heißt, zu jedem möglichen Zug werden wiederum die darauf möglichen Züge generiert. Es entsteht eine Baumstruktur mit der aktuellen Spielsituation als Wurzel. Bewertet werden nur die Blatt-Knoten des Baumes. Anschließend werden diese Bewertungen mittels einer Entscheidungsregel (die vom konkreten Algorithmus abhängig ist) den Baum aufwärts propagiert. Die Bewertungen auf der ersten Ebene (das sind die möglichen Züge, die der Computer ausführen kann) werden dadurch wesentlich besser und "vorausschauender" bewertet. [vgl. St03, s. 26f]

Die Tiefe des Baums, also wieviele Züge in die Zukunft geblickt wird, hängt von der verfügbaren Rechenzeit ab. Dabei ist es üblich, den Spielbaum iterativ mehrmals hintereinander zu generieren. Jedes mal mit einer größeren Tiefe. Die dabei gefundenen Bewertungen werden jeweils aktualisiert. Das geschieht so lange, wie noch Rechenzeit zur Verfügung steht. Dieser Prozess wird als Iterative Deepening bezeichnet. Die Bewertungen werden umso intelligenter, je tiefer der Baum aufgebaut wird. Daher lässt sich mittels der verfügbaren Rechenzeit auch das Schwierigkeits- bzw. Intelligenzlevel der Algorithmen regeln. [vgl. St03, s. 91f]

3.2 Minimax mit Alpha-Beta-Pruning

In Zwei-Spieler-Spielen ist dies der effektivste Algorithmus und kann als Standardlösung angesehen werden [vgl. St03, s. 3f]. Eine gute Beschreibung und Herleitung geben Knuth und Moore [KM75].

Einer der Spieler wird als Max, der andere als Min bezeichnet. Die Bewertungsfunktion liefert ein skalares Ergebnis. Je höher die Bewertung, desto besser für Spieler Max, je niedriger, desto besser für Spieler Min. Da wir von strikt kompetitiven Spielen sprechen, in denen Koalitionen keinem etwas nützen, ist diese Betrachtung zulässig. Bei der Auswahl des besten Zuges wählt Spieler Max daher stets den am höchsten und Spieler Min den am niedrigsten bewerteten Zug.

Was diesen Algorithmus so stark macht, ist, dass eine sehr effektive Form des Prunings (dt. Beschneiden) auf den Spielbaum angewendet werden kann: Das Alpha-Beta-Pruning.

Im Worst Case hat der Algorithmus eine Laufzeit von $O(b^d)$ (das gilt für alle hier betrachteten Algorithmen). Wobei b der branching factor, also die Anzahl möglicher Züge, die ein Spieler ausführen kann, ist. d ist die Tiefe (engl. depth), also wieviele Züge in die Zukunft geschaut wird. Tatsächlich muss aber nicht in allen Fällen der komplette Spielbaum aufgebaut, durchsucht und bewertet werden. Mit Pruning-Verfahren werden Zweige des Baumes, bei denen bereits absehbar ist, dass sie keine bessere Bewertung liefern werden, beschnitten.

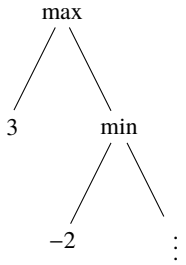


Abb. 1: Alpha-Beta-Pruning. Bereits wenn Min die -2 entdeckt, ist klar, dass Max sich niemals für den rechten Zweig entscheiden wird.

Betrachten wir dazu Abbildung 1. Die Analyse des Spielbaums erfolgt immer über eine Tiefensuche. Nehmen wir an, Spieler Max hat auf dem zuerst bewerteten Zweig (links) eine 3 als Bewertung erhalten. Spieler Max würde sich also für diesen Zweig entscheiden, wenn keine höhere Bewertung gefunden wird. Nehmen wir nun an, dass auf dem nächsten Zweig Spieler Min (ein Zug weiter in der Zukunft) eine -2 als Bewertung erhalten hat. Da Spieler Min immer die kleinstmögliche Bewertung bevorzugt, wird er auf jeden Fall den Zweig mit der -2 oder sogar eine noch kleinere Bewertung wählen. Da Spieler Max aber eine Ebene darüber steht und auf dem linken Zweig bereits ein 3 bekommen kann, ist klar, dass Spieler Max niemals den rechten Zweig wählen wird, sondern stets den linken. Wir können also den kompletten rechten Zweig ignorieren, sobald wir die -2 gefunden haben. Das ist das Alpha-Beta-Pruning.

Der Name leitet sich davon ab, dass in einer Variable namens α der beste bisher gefundene Wert für Spieler Max und in β der bisher beste Wert für Spieler Min gespeichert wird. α wird mit $-\infty$ initialisiert und nun von Spieler Max maximiert. Bei β erfolgt die Initialisierung entsprechend mit ∞ und die Variable wird nun minimiert. In dem Moment, wo $\alpha \geq \beta$ ist, kann beschnitten werden. Denn Spieler Max würde α ja weiter maximieren, Spieler Min hat aber auf einem anderen Zweig schon eine niedrigere (für ihn bessere) Bewertung gefunden und umgekehrt.

Durch dieses Verfahren wird eine Laufzeit von $O(b^{\frac{d}{2}})$ im Best Case und $O(b^{\frac{3d}{4}})$ im Average Case erreicht. [vgl. St03, s. 43; zitiert nach Pe84]

Leider reicht ab drei Spielern keine skalare Bewertung der Spielzustände mehr aus. Damit funktioniert dieser Algorithmus für Spiele ab drei Spielern nicht mehr. Sturtevant und

Korf [SK00] geben allerdings eine Lösung dazu: nämlich den paranoiden Ansatz. Wir nehmen also an, dass der Spieler, der am Zug ist, Max ist. Alle seine Gegner bilden eine Koalition gegen ihn und sind Spieler Min. Nun kann wieder der Minimax-Algorithmus mit Alpha-Beta-Pruning verwendet werden.

Sie geben allerdings zu bedenken, dass diese Annahme zu fehlerhaftem Spielverhalten führen kann, da ja in der Realität gar keine Koalition zwischen den gegnerischen Spielern besteht.

In Sturtevants Versuchen u.a. mit dem Spiel Chinese Checkers [St03, s. 118] ist aber der paranoide Ansatz den anderen Algorithmen häufig überlegen gewesen. Unter anderem weil durch die höhere Suchtiefe die Bewertung der Züge effektiver und "weitsichtiger" ist.

3.3 Max^{N} mit Immediate und Shallow Pruning

Der Max^{N}-Algorithmus ist von Luckhardt und Irani [LI86] vorgestellt worden. Der Minimax-Algorithmus wird dabei auf N-Spieler erweitert. Die Bewertungsfunktion liefert einen Ergebnisvektor mit je einem Eintrag pro Spieler. Die Spieler wählen nun immer jeweils den Zug mit der höchsten Bewertung an ihrer Stelle im Vektor. Es herrscht aber kein direkter Zusammenhang zwischen den Werten im Vektor vor. Dadurch ist das Pruning in der bisherigen Form nicht mehr möglich.

Es gibt dennoch einige Möglichkeiten des Prunings im Max^{N}. Keine ist so effektiv wie das Alpha-Beta-Pruning und alle sind an spezielle Bedingungen geknüpft. [vgl. SK00]

3.3.1 Immediate Pruning

Diese Art des Prunings kann angewandt werden, wenn es eine maximale Bewertung gibt. Eine Bewertung also, die nicht mehr übertroffen werden kann. Sobald ein Blatt im Spielbaum mit der Maximalbewertung gefunden worden ist, können alle weiteren Zweige ignoriert werden, weil sich der Wert nicht mehr verbessern kann. [vgl. SK00]

Im absolut besten Fall sprechen wir dann von einer Laufzeit von $O(b^{\frac{n-1}{n}})$. In den meisten Spielen, in denen es so einen Höchstwert gibt, kommt dieser aber eher gegen Ende des Spieles vor. Zu Beginn des Spieles nützt dieses Verfahren also faktisch gar nichts.

3.3.2 Shallow Pruning

Auch diese Art des Prunings ist von Sturtevant und Korf [SK00] beschrieben worden. Die Bedingung ist hier, dass es eine maximale Summe der Elemente im Ergebnisvektor gibt.

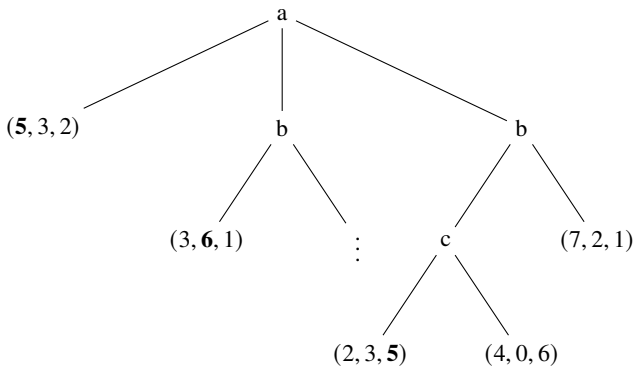


Abb. 2: Shallow Pruning. Die $max\Sigma = 10$ daher wird beim ersten Blatt unter b klar, dass a sich stets für den linken Zweig entscheiden wird.

Betrachten wir dazu Abbildung 2. Spieler a ist am Zug und der erste Wert im Ergebnisvektor gehört zu a. Im linken Zweig kann also eine 5 erreicht werden. Die Summe des Ergebnisvektors ist stets 10. In dem Moment, wo Spieler b im mittleren Zweig die Bewertung $(3, 6, 1)$ erhält, ist klar, dass Spieler a in diesem Zweig maximal noch eine 4 erreichen kann. 6 von 10 Punkten sind ja schon an Spieler b gegangen und dieser wird das Ergebnis für sich eventuell sogar noch weiter maximieren. Daher wird sich Spieler a stets für den linken, anstatt für den mittleren Zweig entscheiden und der mittlere Zweig muss nicht weiter betrachtet werden.

Diese Art des Prunings kann nur zwischen Vater- und Kindknoten durchgeführt werden. Wenn Spieler c im rechten Zweig die Bewertung $(4, 0, 6)$ findet, könnte man ja vermuten, dass der komplette Zweig ignoriert werden kann. Ähnlich wie es im mittleren Zweig der Fall gewesen ist. Auf den zweiten Blick sehen wir aber, dass sich Spieler b im normalen Max^{N} ohne Pruning für den Zweig ganz rechts entscheiden würde. Dadurch bekommt Spieler a eine Bewertung von $(7, 2, 1)$ und wird sich für den rechten Zweig entscheiden. Durch das Pruning zwischen Großvater- und Enkelknoten, hätten wir diese Bewertung allerdings nicht gefunden.

Dadurch beschränkt sich das Pruning nur auf unmittelbar verwandte Knoten und die Effektivität sinkt. Im Best Case tendiert diese Lösung gegen $O(b^{\frac{d}{2}})$. [vgl. St03, s. 52f]

3.3.3 Immediate und Shallow Pruning-Bedingung erzeugen

Immediate und Shallow Pruning sind an zwei Bedingungen geknüpft: es muss einen Maximalwert in der Bewertung geben und eine maximale Summe. Praktischerweise können diese Vorrausetzungen in jeder Implementierung des Max^{N}-Algorithmus geschaffen werden.

Die Idee dazu hat bereits Sturtevant [St03, s. 73] gehabt. Dazu ist folgende Modifikation notwendig:

Angenommen unsere Bewertungsfunktion liefert uns einen Bewertungsvektor $\vec{v} \in {\mathbb{R}_0}^N$. Dann kann dieser zu einem Vektor $\vec{v}^*$ umgewandelt werden, der die Bedingungen erfüllt. Dazu müssen lediglich die Elemente von $\vec{v}$ jeweils durch die Summe von $\vec{v}$ geteilt werden. Also: $v_i^* = \frac{v_i}{\Sigma \vec{v}}$.

Dadurch ist die Summe von $\vec{v}^*$ stets 1. Der höchstmögliche Wert, den ein Element in $\vec{v}^*$ erreichen kann, ist ebenfalls 1. Damit sind beide Vorraussetzungen für Immediate und Shallow Pruning gegeben.

Anmerkung: $\vec{v}$ darf kein Nullvektor sein. Außerdem: wenn Elemente in $\vec{v}$ auch negative Werte annehmen können, dann sollte zunächst zu allen Elementen der Betrag des kleinsten Elements addiert werden. Dadurch gibt es keine negativen Werte im Vektor mehr. Das Verhältnis der Elemente zueinander wird dadurch allerdings minimal verzerrt. Die meisten Bewertungsfunktionen liefern aber ohnehin nur positive Bewertungen.

3.4 Hypermax

Der Hypermax-Algorithmus ist von Mikael Fridenfalk [Fr14] vorgestellt worden. Das Ziel ist es gewesen, eine Variante des Alpha-Beta-Prunings für N-Spieler-Spiele zu entwickeln.

Der entscheidende Punkt für das Alpha-Beta-Pruning ist die Eigenschaft des Nullsummenspiels. Eine positive Bewertung ist für Spieler Max im gleichen Maße gut, wie sie für Spieler Min schlecht ist. Formal ist die Pruning-Bedingung, wie bereits beschrieben, als $\alpha \geq \beta$ definiert. Umgeformt könnte die Pruning-Bedingung auch so geschrieben werden: $a_1 \quad a_2 \geq 0$, wobei $a_1 = \alpha$ und $a_2 = -\beta$ ist. Mit anderen Worten: sobald die Summe der besten bisher gefundenen Werte der Spieler größer oder gleich der konstanten Summe des Spieles ist (in einem Nullsummenspiel ist die Summe also 0), kann der aktuell betrachtete Zweig im Spielbaum keine Verbesserung mehr bringen. Da es sich nun mal um ein Spiel konstanter Summe handelt, ist klar, dass das Übersteigen dieser Summe nicht zulässig ist.

Fridenfalk führt nun fort, dass Alpha-Beta-Pruning möglich wäre, wenn es sich auch beim N-Spieler-Spiel um ein Nullsummenspiel handeln würde. Dazu gibt er eine Methode, dies zu erreichen. $\vec{v}$ ist wieder das ursprüngliche Ergebnis unserer Bewertungsfunktion und $\vec{v}^*$ die modifizierte Version. Die Umwandlung in ein Nullsummenspiel (er bezeichnet dies als eine Transformation in einen Zero-Space-Vektor) erfolgt, indem von allen Elementen in $\vec{v}$ der Durchschnitt aller Elemente in $\vec{v}$ abgezogen wird. Also: $v_i^* = v_i - avg(\vec{v})$.

Der Algorithmus funktioniert nun so, dass in einem Vektor (den Fridenfalk konsequent $\vec{\alpha}$ nennt) die jeweils bisher besten Ergebnisse für die jeweiligen Spieler gespeichert werden. Wann immer eine bessere Bewertung gefunden wird, wird das zum Spieler gehörenden

Element in $\vec{\alpha}$ aktualisiert (maximiert). Sobald nun die Summe von $\vec{\alpha}$ größer oder gleich 0 ist, ist die Pruning-Bedingung erreicht und der aktuelle Zweig wird beschnitten.

Er gibt aber auch zu bedenken, dass durch die Transformation in den Zero-Space die Verhältnisse der Bewertungen zueinander geringfügig verzerrt werden. Außerdem können durch das Pruning Zweige beschnitten werden, obwohl sie eigentlich auf einer tieferen Ebene eine bessere Bewertung liefern würden. Das Problem scheint ähnlich gelagert zu sein, wie der Grund, warum das Shallow Pruning im Max^N nur zwischen Vater und Kindknoten möglich ist, ohne das Ergebnis zu verändern. In dieser Richtung wären weitere Analysen nötig.

Dennoch sagt Fridenfalk, dass der Hypermax eine gute Approximation des ursprünglichen Max^N liefern und durch das effektivere Pruning weiter in die Zukunft blicken kann. Die Argumentation ist also ähnlich zum paranoiden Ansatz. Nicht die Genauigkeit der Berechnung, sondern die höhere Suchtiefe sollen den entscheidenden Vorteil liefern.

Betrachten wir nun, wie sich die Algorithmen gegeneinander behaupten können.

4 Vergleich der Algorithmen

4.1 Vorbereitung

Für den Vergleich sind die folgenden Algorithmen implementiert worden:

- Max^N (ohne Pruning)
- Max^NIS (Max^N mit Immediate und Shallow Pruning)
- Paranoid (paranoider Minimax mit Alpha-Beta-Pruning)
- Hypermax

“Chamäleon Schach” kann zu viert gespielt werden, daher treten alle vier Algorithmen gleichzeitig gegeneinander an. Es macht durchaus einen Unterschied, welcher Spieler der Startspieler ist bzw. welcher als zweites, drittes oder viertes zieht. Daher sind insgesamt 24 Spiele gespielt worden, in allen möglichen Anordnung der vier Spieler um das Spielbrett (4! = 24 verschiedene Anordnungen).

Die Algorithmen verwenden das Iterative Deepening und haben eine Sekunde Zeit, den optimalen Zug zu berechnen. In der App später werden sie ebenfalls eine Sekunde Zeit haben. Die Algorithmen sind in Node.js implementiert worden und der Test ist auf einem MacBook Pro 13"von 2018 mit einem Intel Core i5 Prozessor und 16GB RAM durchgeführt worden.

In den Originalregeln ist kein Unentschieden vorgesehen gewesen. Im Spiel mit realen Menschen ist es bisher auch noch nie zu einem Unentschieden gekommen. Bei den

Algorithmen kann es aber vorkommen, dass sie gegen Ende des Spieles einen zyklischen Ablauf finden. Sie “tanzen” dann mehr oder weniger die ganze Zeit umeinander, ohne dass einer den anderen schlagen und das Spiel entscheiden könnte. Daher sind die Spiele auf 100 Züge begrenzt worden (ein typisches Spiel dauert zwischen 40 und 60 Zügen). Wenn der 100. Zug erreicht ist, wird ein Unentschieden für alle noch lebenden Spieler erklärt.

4.2 Auswertung

Algorithmus	wins	draws	losses	depthAvg	depthMedian
Max^N	8	0	16	5.02	4.08
Max^NIS	10	1	13	7.38	4.08
Hypermax	2	0	22	5.49	4.04
Paranoid	3	1	20	6.39	4.2

Tab. 1: Ergebnisse der 24 Spiele der verschiedenen Algorithmen gegeneinander.

Die Tabelle zeigt die Ergebnisse der 24 Spiele. Es wird zu jedem Algorithmus angegeben, wie häufig dieser gewonnen, unentschieden gespielt oder verloren hat. Bei jeder Berechnung ist die vom Algorithmus erreichte Suchtiefe gespeichert worden. Die Berechnung wird sofort abgebrochen, wenn die vorgegebene Zeit von einer Sekunde erreicht ist. Daher wird die Suchtiefe mit Kommastellen angegeben (3.5 würde bedeuten, dass die Hälfte der Züge mit einer Suchtiefe von 3 und die andere Hälfte mit 4 bewertet worden ist). Die Spalte “depthAvg” enthält den Durchschnitt aller erreichten Suchtiefen je Algorithmus; “depthMedian” den Median derselbigen.

Zunächst fällt auf, dass die durchschnittliche Suchtiefe stark von ihrem Median abweicht. Das liegt daran, dass gegen Ende des Spieles, wenn nur noch sehr wenige Figuren übrig sind, sehr viel tiefer gesucht werden kann. Algorithmen, die sehr häufig gewonnen haben, werden dadurch wesentlich mehr hohe Ausreißer in der Suchtiefe haben. Daher ist der Median besser zum Vergleichen geeignet.

Der Paranoid hat im Median die höchste Suchtiefe erreicht. Dies ist nicht verwunderlich, da das Alpha-Beta-Pruning ja das effektivste Pruning-Verfahren darstellt. Der Hypermax hat tatsächlich die geringste Suchtiefe erreicht. Dies lässt sich dadurch erklären, dass der Hypermax fast alle Spiele verloren hat und daher fast ausschließlich Werte aus der rechenaufwendigen Anfangsphase des Spieles vorliegen. Wahrscheinlich ist es der recht hohen Effizienz des Prunings vom Hypermax zuzuschreiben, dass seine Ergebnisse überhaupt in der Nähe der anderen Algorithmen liegen.

Die Max^N-Algorithmen haben im Median die exakt gleiche Suchtiefe erreicht. In Bezug auf den Durchschnitt liegt der Max^{-N}IS allerdings deutlich höher. Das liegt daran, dass Immediate

und Shallow Pruning erst gegen Ende des Spieles ihre volle Effektivität entwickeln. Dadurch werden besonders am Ende eines Spieles extrem hohe Suchtiefen erreicht. Im Hauptteil des Spielverlaufes hat das Pruning hingegen kaum einen Effekt.

Der Hypermax hat fast jedes Spiel verloren. Fridenfalk führt an, dass der Hypermax nur eine Approximation des Max^N ist und die konkreten Bewertungen der beiden Algorithmen durchaus unterschiedlich ausfallen können. Diese Unterschiede sind für "Chamäleon Schach" so signifikant, dass der Hypermax keine Chance gegen die anderen Algorithmen hat.

Auch der Paranoid hat, trotz höherer Suchtiefe, fast alle Spiele verloren. Sturtevant hat postuliert, dass die paranoide Betrachtung zu einer fehlerhaften Spieleweise führen kann, da die angenommen Koalition zwischen den Gegnern gar nicht existiert. Dieser Fehler potenziert sich außerdem mit steigender Suchtiefe. [vgl. St03, s. 133] Es ist also gut möglich, dass der Paranoid bei geringen Suchtiefen (oder mit weniger Rechenzeit) besser abschneidet. In diesem Versuch hat der Paranoid allerdings keine Chance gegen die Max^N-Algorithmen gehabt.

Die Max^N-Algorithmen schneiden in diesem Versuch am besten ab, mit deutlichem Vorsprung. Der Max^NIS ist seinem Bruder etwas überlegen. Im Hauptteil des Spieles hat das Pruning, wie erwähnt, kaum einen Effekt. Gegen Ende des Spieles erreicht der Max^NIS aber höhere Suchtiefen und spielt damit tatsächlich etwas besser als der Max^N.

5 Fazit

In diesem Paper sind verschiedene Algorithmen für strikt kompetitive N-Spieler-Brettspiele mit perfekter Information miteinander verglichen worden.

Hypermax und Paranoid, welche die beiden Algorithmen mit dem effektivsten Pruning-Verfahren sind, haben in diesem Versuch am schlechtesten abgeschnitten. Der Hypermax hat außerdem nur eine mittelmäßige Suchtiefe erreicht, was allerdings an der Menge der verlorenen Spiele liegen kann. Beide Algorithmen liefern keine optimale Bewertung ab, sollen aber durch besseres Pruning und höhere Suchtiefe diesen Nachteil ausgleichen. Die erreichten Suchtiefen haben bei allen Algorithmen sehr nah beieinander gelegen. In Zukunft könnte untersucht werden, wie sich verschiedene Rechenzeiten auf die Suchtiefe und die Effektivität der Spielweise auswirken. Insgesamt scheint die besondere Beschaffenheit von Mehrspieler-Spielen aber dafür zu sorgen, dass ein effizienteres Pruning nicht mit einer effektiveren Spielweise gleichzusetzen ist.

Die klaren Gewinner sind die "klassischen" Max^N-Algorithmen gewesen. Mit Immediate und Shallow Pruning kann die Suchtiefe und die Effektivität der Bewertungen vor allem gegen Ende des Spieles gesteigert werden. Für "Chamäleon Schach" ist damit der "beste" Algorithmus für den Computergegner gefunden: der Max^N mit Immediate und Shallow Pruning.

In Sturtevants Versuchen u.a. mit Chinese Checkers ist der paranoide Ansatz meist der stärkste gewesen. Für “Chamäleon Schach” hat sich diese Erkenntnis nicht bestätigt. N-Spieler-Spiele bleiben damit ein schwieriges Problem, zu dem es keine Standardlösung gibt. Stattdessen müssen die verschiedenen Algorithmen jeweils mit dem entsprechenden Spiel getestet und verglichen werden.

Literatur

[Fr14] Fridenfalk, M.: N-Person Minimax and Alpha-Beta Pruning. In: NICOGRAPH International 2014. S. 43–52, 2014.

[KM75] Knuth, D. E.; Moore, R. W.: An Analysis of Alpha-Beta Pruning. Artificial Intelligence 6/4, S. 293–326, 1975.

[LI86] Luckhart, C. A.; Irani, K. B.: An Algorithmic Solution of N-Person Games. In: AAAI. Bd. 86, S. 158–162, 1986.

[LR57] Luce, R. D.; Raiffa, H.: Games and Decisions: Introduction and Critical Survey. John Wiley, New York, 1957.

[Pe84] Pearl, J.: Heuristics Addison-Wesley. Reading, MA/, 1984.

[SK00] Sturtevant, N. R.; Korf, R. E.: On Pruning Techniques for Multi-Player Games. AAAI-2000 49/, S. 201–207, 2000.

[St03] Sturtevant, N. R.: Multi-Player Games: Algorithms and Approaches, Diss., University of California, 2003.

Developing a game AI for Murus Gallicus

Philip Wilson[1], Andrej Savinov[2], Annabella Kadavanich[3]

Abstract: The development of game AIs has been a popular challenge in the last years. One of the best game agents, *AlphaZero*, was developed by DeepMind in 2017 and superseded by *MuZero* in 2019. Both agents are based on algorithms that perfectly learn to play any game within not even a day, given they are fed the game's rules [Da18]. The development of such game AIs does not necessarily require big computation centers like the ones Google has. In this work, we show how to develop and implement a *Murus Gallicus* game AI using mainly GOFAI (Good Old-Fashioned Artificial Intelligence) methods. We start with a comparison between different search tree algorithms, including MiniMax, NegaMax, NegaScout (principal variation search) and show how transposition tables can be used for optimization. Furthermore, we demonstrate the advantages of a dynamic value function and time management while searching for the best move. Lastly, we evaluate the application of Evolutionary Learning (EL), explaining how we trained specific parameters.

Keywords: Artificial Intelligence, Game AI, Evolutionary Learning

1 Motivation

This project was part of a course at the Technische Universität Berlin called "Projekt KI: Symbolische Künstliche Intelligenz"(AI Project: Symbolic Artificial Intelligence). This means that we had no choice in the methods we implemented except for EL. Even though Deep Reinforcement Learning methods are currently the best at playing board games and even simple video games, it does not mean we should forget the old school methods that went out of favor in the 1980s. *MuZero* still gets the same score as a random agent for some Atari games like "Montezumas Revenge"where a combination of reinforcement learning and symbolic representations may be needed to beat these games. In this project, we demonstrate how to implement an AI for the game called *Murus Gallicus* (MG). MG is a board game based on the walls of stone built by the Gauls against Roman aggressors in the Gallic Wars. The game was invented in 2009 by Phil Leduc [Ph19]. We examine the advanced version of MG, which also includes catapults in addition to walls and towers.

[1] TU Berlin, Straße des 17. Juni, 12309 Berlin, Deutschland, philipollaswilson@protonmail.com
[2] TU Berlin, Straße des 17. Juni, 12309 Berlin, Deutschland, andrej.savinov@campus.tu-berlin.de
[3] TU Berlin, Straße des 17. Juni, 12309 Berlin, Deutschland, a.kadavanich@campus.tu-berlin.de

MG is played on a 7 x 8 board. There are two players, each has 16 stones of their color, where light represents the Romans and dark the Gauls. A tower consists of two stones of the same color stacked on top of each other, while a single stone is called a wall. A catapult consists of three stones of the same color, stacked on top of each other. At the start of the game each player possesses a row of towers. The Romans start the game. On their turn, a player may move a tower, capture with a tower or fire a catapult. Walls cannot be moved. Towers can move to any straight direction, orthogonal or diagonal. Both stones are removed from the starting square and placed, one each, on the next two squares of the chosen direction. Destination squares may not be occupied by adversarial pieces or friendly catapults. If a destination square is occupied by a friendly wall it becomes a tower, and if occupied by a friendly tower it becomes a catapult. Towers can capture an adjacent adversarial wall by sacrificing one of its stones. Both stones are then removed from the board. Towers can downgrade an adjacent adversarial catapult to a tower by sacrificing one of its stones. Two stones are removed in total, one of the attacking tower's stones and one of the defending catapult's stones. Similarly, towers can downgrade a catapult to a wall, by sacrificing two stones in total. Catapults can throw one of their stones two or three spaces away in all five forward directions into an empty or opponent occupied cell. The player who places a wall on the opponents home row wins the game. If a player cannot make a valid move anymore he loses the game.

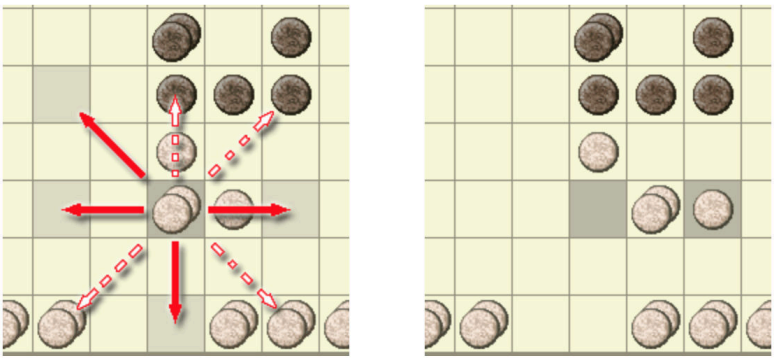


Fig. 1: Exemplary game state (right) showing in which directions a tower can move (left) Source: http://www.iggamecenter.com/info/de/murusgallicus.html, last visited on 15.07.2020

Even though an empirical analysis was not performed, our intuition is that the state and action spaces are smaller than in chess. Unconstrained by the board all possible actions are: a tower can move in eight directions or capture any adjacent enemy wall or downgrade a catapult, and a catapult can fire towards six different locations. Every one of these moves, except the tower move, are guaranteed to reduce the action space, as pieces are either lost or walls are created which have no moves. As there are only seven types of states a square can

have (empty or wall, tower or catapult of either colour) compared to 13 states in chess, and as the board is 7x8 and not 8x8, we also believe the state space to be smaller.

2 Tree search algorithms

This paper is part of an university project at TU Berlin, consisting of different student groups who developed their own game AI. In total seven groups participated. During three milestones (MS), and one pre-milestone, all game AIs competed against each other in a contest simulation environment. During each contest each move had a time limit, therefore the AI that made the smartest cutoffs (using alpha-beta pruning [BM01]) and could search the game tree the deepest, generally won the contest. Therefore, this chapter is dedicated to introducing and comparing different search tree algorithms.

2.1 Minimax

MiniMax is the standard algorithm for two player perfect-information games such as MG. It searches forward to a fixed depth in the game tree, limited by the amount of time available per move. Then, a heuristic evaluation function is applied, which takes a board position and returns a number that indicates how favorable that position is for one player relative to the other. One player, called MAX, seeks to maximize this number, whereas the other player, called MIN, seeks to minimize it. Finally, it recursively computes the values for the interior nodes in the tree according to the maximum rule. The value of a node where it is MAX's turn is the maximum of the values of its children, while the value of the node where MIN is to move is the minimum of the values of its children [Po89].

2.2 Iterative Deepening Negamax

NegaMax [Al90] search is a variant form of the MiniMax search algorithm. We have chosen it in order to simplify the implementation of the MiniMax algorithm. NegaMax takes advantage of the zero-sum property of MG. In other words, the value of a position for MAX is the negation of the value for MIN. Instead of MAX selecting the move with the maximum-valued successor and MIN selecting the move with the minimum-valued successor, they both look for a move that maximizes the negation of the value resulting from the evaluation function.

2.3 NegaScout: Principal Variation Search

NegaScout is a directional search algorithm for computing the MiniMax value of a node in a tree. It is said to be faster than NegaMax with alpha-beta pruning [BM99]. This is true also for our MG AI (see *Fig. 2*). In our implementation we use a null window to reduce the amount of nodes that need to be computed. We used move sorting and only searched the best move with a full window, all other moves were searched using null windows. Though this can be risky, theory shows [Ma86] that if the null window value is bigger than the alpha value and smaller than beta then the move could possibly have a better value and therefore should be searched using a bigger window. If such a situation appeared, another NegaScout computation using a bigger window was calculated.

2.4 Transposition Tables with zobrist hashing

It is possible to arrive at the same game state through different combinations of moves [BUH70]. To avoid calculating the value of the state more than once, the values are stored in *Transposition Tables* (TT) [BTW00]. To increase the search speed during lookups, we also implemented zobrist hashing [JF16]. The depth, at which each value was discovered, is also stored in order to give priority to states whose value was calculated with a larger subtree (the ones with lower depth). When the table fills up, less-used states are removed to make room for new ones.

Fig. 2 shows a comparison of the computed nodes using NegaMax, NegaScout without TT and NegaScout with TT. The x-axis describes the tree depth that was used and the y-axis the amount of nodes that were computed. Taking the average amount of nodes for several different fen strings, we found that NegaScout generates on average almost twice as many nodes as NegaMax and it only takes less than half of the time doing so. Using NegaScout with TT we could decrease our computation time and therefore reach another speed up.

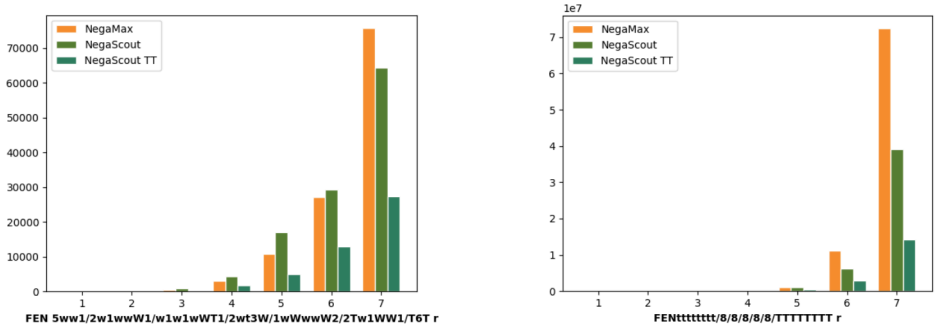


Fig. 2: Comparison of NegaMax, NegaScout without TT & NegaScout with TT for 2 FEN Strings

Comparing the computation time, using NegaScout is 1.5 times faster than NegaMax in average. Furthermore, we noticed that on average NegaScout with TT is 2.9 times faster than NegaMax. Also the direct comparison of NegaScout with TT and NegaScout without TT showed, that **not using TT** almost doubled the computation time. These calculations are based on measuring the time and amount of computed nodes for all three algorithms for approximately 20 different FEN strings and then computing the arithmetic mean.

2.5 Value Function

Besides optimizing the search tree algorithm, our most challenging task while developing the game AI was to design a good value function [RA10]. In our implementation we started with a basic value function, only consisting of a *Piece-Square Table* [CP20] (PST) and then progressively added the following five features:

1) Amount of towers > amount of walls, 2) amount of catapults > amount of walls, 3) mobility 4) Dominance of the middle row and lastly, 5) giving or taking points, depending on whether the amount of our towers is larger than the amount of enemy towers. We adjusted the parameters by adding weights that could easily be changed. With this dynamic value function we were able to win almost all games against the other groups at the second milestone. Furthermore, we had implemented a better defense strategy that forced our AI to build a row of defensive towers in the middle of the board using our PST. Once this position was achieved we deactivated the use of PST and played more aggressively towards the winning line of the opponent. Since each game only consisted of 120s we did not re-activate the PST in case the defense border was destroyed as there was no time to rebuild it. In addition, we implemented a panic mode that got activated once our time limit dropped below a certain threshold. In panic mode the search depth was decreased and a more aggressive play-style was activated.

2.6 Summary

We examined different features in order to develop a good MG AI. We started with very simple features and core functions, e.g. the implementation of a move generator and bitboards in MS1. Our value function at that time was only the PST and we used NegaMax with alpha-beta pruning as our main search tree algorithm. Our time management was static and we would always search depth five without any panic mode. Moving forward to MS2, we implemented NegaScout with a null window, we adapted our static time management to a dynamic one, added a panic mode and variable search depths. We also started implementing TT with Zobrist Hashing which improved our computation time Most important for MS2 was also the change of our value function. Instead of only having two PSTs we added six tables, three for each player and one table for each material (Materials: wall, tower, catapult). In addition, our value function was changed to be a linear combination of seven parameters.

The analysis of the contests in MS1 and MS2 showed that all AIs played without any variations, leading to two adversarial AIs always playing the same game against each other repetitivly. To create a more dynamic value function, some type of learning would be helpful. Therefore, we tried using Machine Learning (ML) techniques of our choice to improve the performance of our AI. We implemented Evolutionary Learning (EL) since there was no MG training data available. In the next chapter we will explain, why we chose this technique, how we trained our AI, which challenges we faced and how our AI performed in the last MS3 contest.

3 Evolutionary Learning

EL [Ri01] is inspired by the evolutionary processes happening in nature. Therefore, every evolutionary algorithm needs a fitness function. A function used to compare organisms in order to decide which ones survive and which ones will pass their genes on to the next generation. As we had no datasets to train this fitness function with, we had to establish the fitness function by means of playing numerous games between the organisms in each generation (SPC) regardless of its impracticality [Da09].

3.1 Parameter Adjustments

The first parameter we had to decide on was the population size. Our initial population size for each epoch was ten. In the following, game AIs are referred to as so called *Individuals*. Furthermore, we decided to only train one parameter at a time. We chose to start with the material values (wall, tower, catapult), since other values were more complex to train (e.g. PSTs). Every Individual in the population had different material values in order to ensure variation. Our contest AI from MS2 was also part of the population. We chose Round-Robin (RR) as the tournament mode. We also considered K.O. tournaments as they finish faster than RR and we hoped to speed up our training.

3.1.1 Crossover

We set the probability for a crossover to 10-15%. When we used these low values the improvement of our population was very slow. Since we only had three weeks of training time, until the next MS contest, we increased the value to 80%. This allowed us to create more variance and speed up our training time.

The crossover was implemented in the following way: First, we determined the current best Individual based on the win rate. This way of selection is called *elitism* [TD94]. We perform a crossover on all Individuals, except our MS2 AI and the selected current best Individual. The new material values were calculated as follows: we randomly chose four Individuals of the whole population and select the best Individual from this group (again

using the win rate). Next, this step was repeated. In this way, we selected two Individuals. This type of selection is called *tournament selection* [Ge20]. It is possible – though rare – that the same Individual is selected twice in this process. If this happens, we randomly re-selected another AI. Next, we created a new Individual object by summing up the material parameter of both Individuals from the tournament selection and dividing this number by two. We did this separately for the wall, the tower and the catapult parameter. This tournament selection process is repeated until a new population of eight Individuals is created (assuming that the best Individual of the population was not our MS2 AI). Then we added the best Individual and our MS2 AI to the new population and therefore had a new population of ten Individuals again, used for training the next epoch.

3.1.2 Mutation

Our mutation process was similar to the crossover (see 3.1.1). The probability was set to 50%. Normally in EL this parameter is very low (< 5%) [Go89] but as we only had limited time and needed to speed up the training process we increased the probability.

Again, we implemented a variation of elitism selection. For each Individual – except the Individual selected by elitism and our MS2 contest AI – we randomly choose the material value for mutation. We then overwrote this value by adding a random ranged offset to the current value of the Individual. This step was repeated until all Individuals mutated.

3.2 Material Parameter

We spent about two weeks setting up our training environment (implementing the classes, controlling our memory overflows, optimizing the parameter and the crossover, see 3.1.1 & 3.1.2) and in parallel, started training the weight parameters that are multiplied with the PST. We noticed that after approximately 20 generations our population would start to **converge** and no further improvements were made. Since this happened continuously for different populations we decided to move on and train other parameters.

In summary, we did find three Individuals that were able to win against our MS2 AI. We also tested these three Individuals in a local contest against the MS2 AI's from the other groups. We had slightly better scores than in MS2. In total, the results were not satisfying and also did not mirror the effort and time we had spent on the training of the parameter.

3.3 Piece-Square Table

We also decided to try training the PST parameter. Instead of just training the material factor of the PST, we trained every index of the PST list. Mutation and crossover rate was the same as in section 3.1.2 and 3.1.1. As we could not see any significant progress after one week, we decided to stop training these parameters and moved on and implement *Monte Carlo Tree Search* (MCTS) (see 3.4). We assume that the amount of parameters (3 x 56 indices, 3 PST per player) was too high and therefore much longer training time and a better GPU infrastructure would have been needed in order to effectively train the PST parameter. The value range of each index of the PST that we tried to train was also very large. Therefore we had to limit the random mutation offset to be in between the range of $[-200, 200]$. This limitation caused less variance and maybe prevented us from training more efficient PST, but was a necessary trade-off regarding the lack of having a GPU cluster infrastructure for the training. Furthermore, the probability of finding a good PST is very low as there are dependence between the list parameters.

3.4 Monte Carlo Tree Search (MCTS)

We had expected to have more success with our SPC, so we decided to try an algorithm which could improve performance without relying on a dataset. The MCTS is based on playouts. In each playout a MG game is played until the very end by selecting moves randomly. The final result of this game is used to weight the nodes so that better nodes are more likely to be chosen in future playouts [JKR17]. Due to time limitation, we only used MCTS for the first couple of moves as an *opening book* [NHI06] and then continued with the NegaScout algorithm.

3.5 Summary

Overall the results were not as good as we would have expected. As there was no training data we wanted to find out if SPC could lead to sufficient results. With a training period of only four weeks – including setup and implementation – we discovered that for MG only minimal improvement was possible. Furthermore, isolating and training parameters and then putting them back together lead to strange results sometimes. We believe this was caused by dependencies among the parameters. In total, none of the other groups was able to significantly increase their game AI performance using EL.

Yet, EL has the potential to be used to increase the performance of game AIs. Studies of Omid *et al.* show that by using EL for chess, good value functions can be derived [Da14]. In contrast to our work, they had an initial population of size 100, crossover rate was only 75% and mutation had a probability of 0.5%. They evaluated 200 generations. We believe they had better results with EL than we did, because the initial evaluation function of the population was derived from actuals moves of grandmaster-level games. As future work,

we would spend more time on MCTS. Assuming there were MG datasets, Koppel *et al.* [Da09] suggests to train a certain amount of AIs with the help of a fitness function. This fitness function was based upon the MG datasets. SPC was then performed on these AIs to determine the fittest one of them. This paper shows that SPC can be useful, if the initial population has been selected or trained before hand (and not random as we did).

4 Performance & Contest Results

In this chapter we will compare our performance during all three MS contests. We will mainly compare our results against the average performance of the other groups as a benchmark. We will not only measure our AI by counting the games that were won, but also have a look at the amount of rounds we were able to play and the minimum and maximum computation time per move. All together we will then try to measure the quality of our MG AI.

4.1 Overall Contest Performance

Overall our AI continuously improved during all three milestones. *Fig. 3 – left,* shows the amount of won games in percent (y-axis) relative to the total amount of played games for each MS (x-axis). Our AI did improve about 30% in between MS1 and MS2. On MS3 we had several different contest settings (Tournament mode, time per move, loading / no loading of opening books) and therefore a direct comparison is not possible.

Fig. 3 – right, shows the average amount of moves per game. In MS2 we were able to play five more moves per game on average compared to the other groups. As shown in chapter 4.1 the amount of won games was also improved in MS2. Therefore we conclude, that not only did we play longer games, but also win more games and therefore improved the performance of our AI.

4.2 Minimum & Maximum Time per Move

We also compared our minimum and maximum time per move with the average time of the other groups. *Fig. 4* shows the minimum (left) and maximum (right) computation time (y-axis) per MS (x-axis). We see, that our minimum time was 0 ms for every milestone contest. In contrast, our maximum computation time was below average which means we could have used our time more efficiently. In MS1 and MS2 we used approximately 5 ms per move, in MS3 we were able to triple the move selection time to 15,162 ms. Further improvements for our AI would therefore be the implementation of a more efficient usage of the computation time.

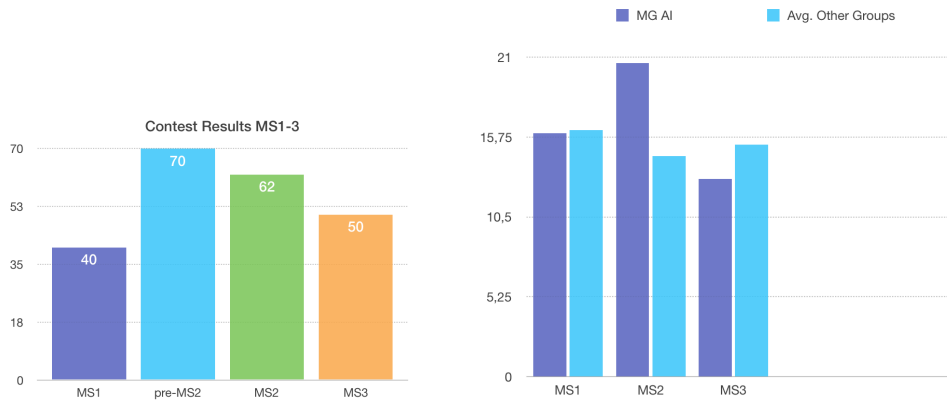


Fig. 3: Left: Average of won games in percent. Right: Amount of moves per game, MS1-3.

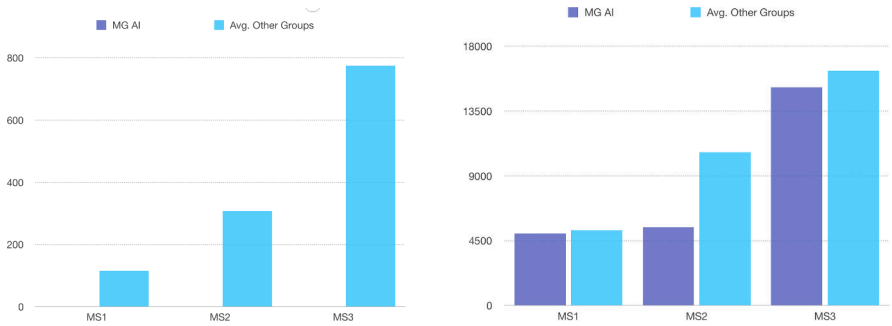


Fig. 4: Minimum (left) & maximum (right) time per move, MS1-3.

5 Conclusion

As shown in section 4.1 our AI was always performing on average or better than the other groups. The AIs we submitted at each MS were always competitive – they were able to win at least several games in each tournament. All the features that we implemented showed visible performance improvements. We reduced the amount of computed nodes using a combination of NegaScout and TT. Also, we could show in 4.2, how we adapted our time management to efficiently use the available time in each MS. For further work and improvements we would suggest speeding up the computation time in order to search deeper in the game tree. This could be realized by optimal selection of TT entries. For example, recognizing which positions are more likely to appear again. In addition, we would debug and complete our MCTS implementation in order to create more efficient opening book. Regarding the dynamic time management, we would try using a heuristic by using the branching factor to determine if we are able to search deeper.

Overall, the project showed us how important, but also difficult, it is to design an efficient value function. We also noticed that dependencies between different features can become very inefficient if there is a lack of computational power or time. The strongest game AIs, like AlphaZero, do not depend on manual value functions as they operate on zero game knowledge. It is important to find algorithms that work without data sets because we can not assume their availability in real life scenarios. Therefore, using AlphaZero on MG could be a promising alternative to the approach we followed in this project.

6 Acknowledgement

We want to thank our supervisor **Dr.-Ing. Stefan Fricke** for his great support and for giving us the highest possible flexibility during this project by letting us implement a wide range of different AI algorithms.

References

[Al90] AlthÃ¶fer, I.: An incremental negamax algorithm. Artificial Intelligence 43/1, pp. 57–65, 1990, ISSN: 0004-3702, URL: http://www.sciencedirect.com/science/article/pii/000437029090070G.

[BM01] Björnsson, Y.; Marsland, T. A.: Multi-cut alpha-beta-pruning in game-tree search. Theoretical Computer Science 252/, pp. 177–196, Feb. 2001.

[BM99] Björnsson, Y.; Marsland, T.: Multi-cut Pruning in Alpha-Beta Search. In (van den Herik, H. J.; Iida, H., eds.): Computers and Games. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 15–24, 1999, ISBN: 978-3-540-48957-3.

[BTW00] Baxter, J.; Tridgell, A.; Weaver, L.: Learning to Play Chess Using Temporal Differences. Machine Learning 40/3, pp. 243–263, Sept. 2000, ISSN: 1573-0565, URL: https://doi.org/10.1023/A:1007634325138.

[BUH70] Breuker, D.; Uiterwijk, J.; Herik, H.: Replacement Schemes for Transposition Tables. ICCA Journal 17/, Feb. 1970.

[CP20] (CPW), C. P. W.: Piece-Square Tables, https://www.chessprogramming.org/Piece-Square_Tables, [Online; accessed 11-May-2020], 2020.

[Da09] David, O. E.; van den Herik, H. J.; Koppel, M.; Netanyahu, N. S.: Simulating Human Grandmasters: Evolution and Coevolution of Evaluation Functions. 2009.

[Da14] David, O. E.; van den Herik, H. J.; Koppel, M.; Netanyahu, N. S.: Genetic Algorithms for Evolving Computer Chess Programs. IEEE Transactions on Evolutionary Computation 18/5, pp. 779–789, Oct. 2014, ISSN: 1089-778X.

[Da18] David, S.; Thomas, H.; Julian, S.; Demis, H.: AlphaZero: Shedding new light on chess, shogi, and Go, https://deepmind.com/blog/article/alphazero-shedding-new-light-grand-games-chess-shogi-and-go, [Online; accessed 06-May-2020], 6.12.2018.

[Ge20] GeeksforGeeks: Tournament Selection (GA), https://www.geeksforgeeks.org/tournament-selection-ga/, [Online; accessed 11-May-2020], 2020.

[Go89] Goldberg, D. E.: Zen and the Art of Genetic Algorithms. In: Proceedings of the Third International Conference on Genetic Algorithms. Morgan Kaufmann Publishers Inc., George Mason University, USA, pp. 80–85, 1989, ISBN: 1558600063.

[JF16] Jinnai, Y.; Fukunaga, A.: Abstract Zobrist Hashing: An Efficient Work Distribution Method for Parallel Best-First Search. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence. AAAI'16, AAAI Press, Phoenix, Arizona, pp. 717–723, 2016.

[JKR17] James, S.; Konidaris, G.; Rosman, B.: An Analysis of Monte Carlo Tree Search. In. Feb. 2017.

[Ma86] Marsland, T. A.: A Review of Game-Tree Pruning. J. Int. Comput. Games Assoc. 9/, pp. 3–19, 1986.

[NHI06] NAGASHIMA, J.; HASHIMOTO, T.; Iida, H.: SELF-PLAYING-BASED OPENING BOOK TUNING. New Mathematics and Natural Computation (NMNC) 02/, pp. 183–194, July 2006.

[Ph19] Phil, L.: Murus Gallicus, https://sites.google.com/site/theowlsnest02/home/murus-gallicus, [Online; accessed 11-May-2020], 2019.

[Po89] Polak, E.: Basics of Minimax Algorithms. In (Clarke, F. H.; Dem'yanov, V. F.; Giannessi, F., eds.): Nonsmooth Optimization and Related Topics. Springer US, Boston, MA, pp. 343–369, 1989, ISBN: 978-1-4757-6019-4, URL: https://doi.org/10.1007/978-1-4757-6019-4_20.

[RA10] Raoof, O.; Al-raweshidy, H.: Theory of Games: an Introduction. In. Sept. 2010, ISBN: 978-953-307-132-9.

[Ri01] Riechmann, T.: Genetic algorithm learning and evolutionary games. Journal of Economic Dynamics and Control 25/6, Computing, economic dynamics, and finance, pp. 1019–1037, 2001, ISSN: 0165-1889, URL: http://www.sciencedirect.com/science/article/pii/S016518890000066X.

[TD94] Thierens, D.; DE, G.: Elitist recombination: an integrated selection recombination GA. In. 508–512 vol.1, July 1994, ISBN: 0-7803-1899-4.

Text Mining

Semantic Code Search with Neural Bag-of-Words and Graph Convolutional Networks

Anna Abad Sieper[1], Omar Amarkhel[2], Savina Diez[3], Dominic Petrak[4]

Abstract: Software developers are often confronted with tasks for which there are widespread solution patterns. Searching for solutions using natural language queries often leads to unsatisfying results. Github, Microsoft Research and Weights & Biases created the CodeSearchNet Challenge [Hu19] to address this problem. Its goal is to develop code search approaches that return the code that best matches a natural language query. In this paper, we investigate two different approaches in this context. First, a Neural Bag-of-Words encoder using TF-IDF weighting and second, a Graph Convolutional Network which includes the call hierarchy in a target method's representation. In our experiments we were able to improve the Neural Bag-of-Words models, whose results were published in the CodeSearchNet Challenge. Our Neural Bag-of-Words encoder improves the MRR by 4.38% for Python and 4.98% for Java. The Graph Convolutional Network did not improve the results over of the Neural Bag-of-Words model.

Keywords: Semantic Code Search; Graph Convolutional Network; Neural Bag-of-Words; CodeSearchNet Challenge

1 Introduction

This paper introduces two novel approaches towards the development of a search engine for code. This challenge - referred to as semantic code search - has applications in supporting software development because it faciliates an easier on-demand reuse of code. Github, Microsoft Research and Weights & Biases have established the CodeSearchNet Challenge [Hu19] for this purpose. The participants are asked to design their own approaches to retrieve a suitable code function from a corpus. The provided data corpus includes code functions from open source Github repositories and their docstrings. It contains code in the programming languages Python, Java, Ruby, Go and PHP.
Semantic similarities between search queries and code functions are not always based exclusively on the same keywords, for example when looking at synonyms such as "picture" and "image" and declensions such as "load" and "loading". Instead of using simple keyword matching, semantic relationships between words should be learned.
One common approach called *Bag-of-Words* is to be developed. Bag-of-Words models map

[1] Hochschule RheinMain, Wiesbaden, Germany a.abadsieper@gmail.com
[2] Hochschule RheinMain, Wiesbaden, Germany omar.m.amarkhel@gmail.com
[3] Hochschule RheinMain, Wiesbaden, Germany savinadz@gmail.com
[4] Hochschule RheinMain, Wiesbaden, Germany dominicpetrak@googlemail.com
https://github.blog/2019-09-26-introducing-the-codesearchnet-challenge/

cc by sa

search queries and code functions each to a vector that marks the keywords. By means of neural networks, similarities between vectors belonging to each other (code function and corresponding docstring) are to be established. Ultimately, syntactically different words that are, however, in context with each other are to be identified. For example, it could be learnt that queries containing the word “database” match functions containing the word “db”.
Our second approach deals with the inclusion of code that is called within code functions. For this purpose, caller graphs are used. They are assembled based on the corresponding Github repositories and consumed by a Graph Convolutional Network. With the help of this technique, the features of a function are to be extended by the features of its callees. Ultimately, functions are to be characterized more extensively in this way.
We make the following contributions:

- A Graph Convolutional Network trained on caller graphs is used for semantic code search.
- A Neural Bag-of-Words method with weighting schemes for identifying semantic similarity between search query and code function is used to retrieve code.
- The models are evaluated quantitatively and compared to baseline models from the CodeSearchNet Challenge. Our findings are that TF-IDF weighting and Byte Pair Encoding bring substantial improvement to the Neural Bag-of-Words approach.

Starting with related approaches, the CodeSearchNet Challenge will be presented and explained in more detail. Then the data corpus used is presented. In the following chapter we present our approaches to graph-based data processing and to identifying similarities between docstring and code using the Bag-of-Words method. Finally, the results of our experiments with the presented methods are shown and a brief conclusion is given.

2 Foundations

This work addresses the CodeSearchNet Challenge and the neural learning of graph representations. A short overview of related work is given below.

2.1 CodeSearchNet Challenge

Semantic code search is the task of finding relevant code given a natural language query. It requires bridging the gap between natural language and the language used in the program code. Unlike natural language, code is not created to be read. Code contains purely syntactic tokens and different identifiers for the same content. The goal is to relate identifiers in code, such as “db”, to terms like “database” in natural language.

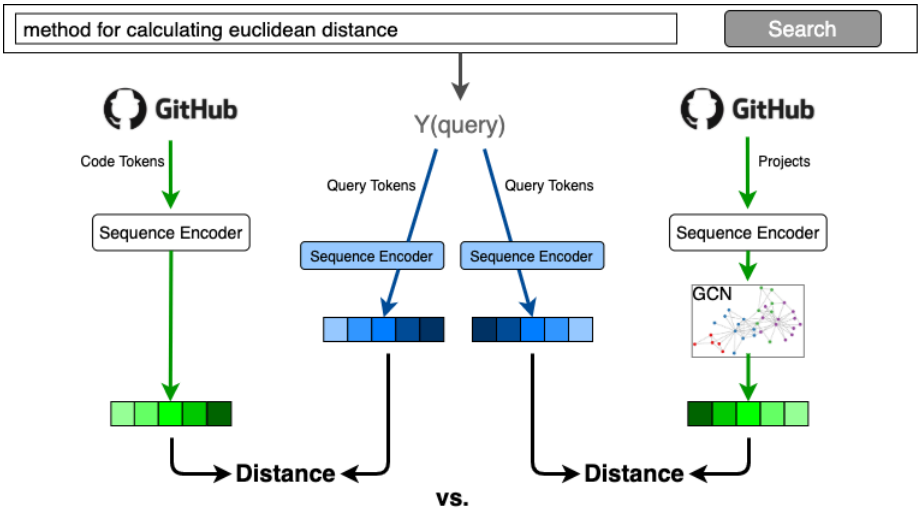


Fig. 1: We suggest two approaches towards code search (left/right). In both approaches, queries (blue) and the respective code (green) from Github are processed by an encoder. Then the respective pairs are matched based on the distance (bottom).

With today's standardized code documentation, both the code and natural language descriptions can be extracted for each documented method of a software project, resulting in data records containing pairs of code and language. This forms an interesting basis for learning code-language similarity measures at the large. To encourage researchers and practitioners to further explore this interesting task, the CodeSearchNet Challenge was created and presented by Github, Microsoft Research and Weights & Biases as a competition.

The CodeSearchNet Challenge provides an extensive corpus of training, test and validation data. Overall it consists of 2 million (comment, code) pairs crawled from different open source libraries, covering six widely used programming languages: Python, JavaScript, Java, Ruby, Go and PHP. The dataset is provided as json lines files, each containing json objects of (comment, code) pairs in a preprocessed manner [Hu19]. This enables participants to directly reuse them for building representations and training, testing and validation of machine learning approaches like neural networks.

2.2 Related Works on Retrieving Code

There exist various approaches on retrieving code. In this section, some related works are mentioned that dealt with a similar topic.
In "Semantics-Based Code Search" by Reiss et. al [Re09], relevant source code snippets are suggested based on the specification formulated by the user. This specification must at best include semantics and signature and should be formulated as precisely as possible.

The initial suggestions are selected by keyword matching. To have a more robust system with regard to synonyms, our approach learns semantic connections between terms. Wan et. al's approach in their paper "Multi-Modal Attention Network Learning for Semantic Source Code Retrieval" [Wa19] uses Abstract Syntax Trees and Control Flow Graphs for code embeddings. Furthermore, the system works with an LSTM and a Gated Graph Neural Network. An abstract syntax tree is used to decompose code fragments syntactically in order to better extract the semantics. In contrast, we only use the code tokens. The approach of Kanade et. al, discussed in "Pre-trained Contextual Embedding of Source Code" [Ka19], proposes CuBERT (Code Understanding BERT) as a further development of the BERT framework. Here Word2Vec embeddings for code are pre-trained. As in this paper, in Sachdev et. al's paper "Retrieval on Source Code: A Neural Code Search" [Sa18], word embeddings are used in combination with TF-IDF weighting. Furthermore, a high-dimensional vector similarity search is performed. Additionally, a supervision layer and a custom ranking system have been developed. In Cambronero et. al's work "When Deep Learning Met Code Search" [CKC], the authors used a Sequence-to-Sequence Gated Recurrent Unit Network and an LSTM for code retrieval.

2.3 Graph Convolutional Network (GCN)

The key to semantic code search is to understand the complete functionality of a method. However, methods are usually not functionally complete but may outsource vast parts of their functionality to other methods, so that the internal dependencies of methods become relevant. To represent the functional range of a method, it is common practice to represent software programs as caller graphs [NGV08]. A Graph Convolutional Network can be applied to the resulting graph-structured data. Our key idea is to utilize this caller graph to learn more complete representations of methods for semantic code search.

A GCN is a folding neural network for learning on graph structured data. The most prominent approach for this task is provided by Kipf et al. [KW16] and offers a modified architecture for spectral graph convolution on undirected graphs. Each layer can be described as follows:

$$H^{l\ 1} = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^l W^l) \tag{1}$$

In 1, $\hat{A} = A \quad I$ is the graph's adjacency matrix with added self-connections and I as identity matrix. $\hat{D}$ is the diagonal node degree matrix of $\hat{A}$. W^l is a layer-specific trainable weight-matrix. Input parameters for $H^{l\ 1}$ are H^l and A, with $H^0 = X$ and X as feature matrix. $\sigma(\cdot)$ is a non-linear activation function (in our case ReLU, Rectified Linear Unit). The GCN proposed by Kipf et al. shows a significant increase in efficiency and improvement in the classification of nodes and graphs, compared to earlier work in this field [KW16].

3 Approach

The general approach towards semantic code search is to encode a search query and code function tokens obtaining embeddings. Then the resulting embeddings are compared, obtaining a similarity. The following notation is defined:

- Given a search query q and a target function f, a sample is designated (q, f).
- The vocabulary that is created using the training samples and should contain all relevant words is referred to as V. It consists of the preprocessed natural language tokens and code tokens.
- A word (or token) is denoted by t. This can belong to a query or a method. A list of relevant tokens from a query or a method, also referred to as a document below, is denoted by $d = [t \mid t \in d \text{ and } t \in V]$.
- The number of occurrences of a word t in document d is represented as $\#(t, d)$.
- The embedding vector of a document d is a real-valued vector and denoted by e_d.

Given a query q, we rank functions f by the similarity between e_q and e_f.

3.1 Bag-of-Words

As a baseline a *Bag-of-Words* encoder is used. For this purpose, a vocabulary is generated from the most frequently occurring words in the entire corpus. Each document gets an embedding $e_d \in \mathbb{N}^V$, where $e_d[i] = \#(t_i, d)$ and $t_i = V[i]$. Finally, the distances of an encoded query to the encoded code functions are measured by the angular metric. The smaller the distance, the better the function is ranked. In the following, the implemented model is referred to as BoW (Bag-of-Words) encoder.
The data from the CodeSearchNet Challenge is preprocessed by splitting at special characters, camelcase splitting as well as lowercasing and stopword filtering. The stop word list contains the keywords of the respective programming language. For example, the document d = [def, connect_to_db(portNumber)] is transformed to d' = [connect, to, db, port, number].

In further experiments the preprocessing is extended by a Byte Pair Encoding (BPE). For this purpose, the vocabulary from the training data is created and stored using a Sentencepiece model. Before creating the embeddings, the tokens are preprocessed using this model.

In order to distinguish relevant words from unimportant ones, words can also be weighted using the TF-IDF (Term Frequency-Inverse Document Frequency) scheme. The weight of

https://github.com/google/sentencepiece

a word in the entire corpus tfidf(t,d) is thus calculated as shown below and normalized to euclidean norm [De20].

$$\text{tfidf}(t, d) = \#(t, d) \cdot log \frac{\#d \quad 1}{\sum \mathbf{1}_{t \in D} \quad 1} \tag{2}$$

The idea of this formula is to combine the frequency of a token t in the document d and the relative frequency of the documents in which t occurs into a total weight.
The vocabulary is created from all tokens in all documents. The embedding e_d contains the TF-IDF values in the order of the words in the vocabulary. Frequent tokens get low weights anyway, so a stopword list is omitted. The encoder operating with the TF-IDF weights is referred to as BoW TF-IDF encoder.

3.2 Neural Bag-of-Words

To improve the results of the BoW TF-IDF encoder, we aimed for a fuzzy matching (e.g. image matches picture), which we implemented with a Neural Bag-of-Words encoder (NBoW). Each token $t \in V$ gets an embedding $e_{\mathrm{t}} \in \mathbb{R}^{\mathrm{k}}$ of k dimensions. The embedding of a docstring or a method is calculated by averaging its tokens' embeddings. In addition, TF-IDF weights are used to weight the embeddings accordingly:

$$e_{\mathrm{d}} = \frac{1}{\#d} \sum_{t \in d} \text{tfidf}(t, d) \cdot e_{\mathrm{t}} \tag{3}$$

The embeddings e_t are trained such that matching code and queries are considered similar while unmatching code and queries are not. The Triplet Margin Loss [VM16] is used for this in training. It takes a docstring embedding as anchor e_a, its corresponding code embedding as positive example e_p and the code embedding of a random other sample as negative example e_n. A margin γ is also used, which is set to 1 by default.

$$L(a, p, n) = max\{||e_a - e_p||_2 - ||e_a - e_n||_2 \quad \gamma, 0\} \tag{4}$$

Basically, the loss is zero if e_a and e_p have a short distance, but e_a and e_n have a large distance.

3.3 GCN-based Neural Bag-of-Words

As a novel approach towards code representation we use Graph Convolutional Networks. To represent the functionality of a method as accurately as possible, it is important to understand all other methods called in the method under consideration. To view internal

dependencies, we consider a method as a node in a caller graph [NGV08]. For learning on this kind of data we use the modified framework of spectral graph convolutions by T. Kipf and M. Welling [KW16]. Since this approach can only work on undirected graphs, we also consider caller graphs as undirected graphs.

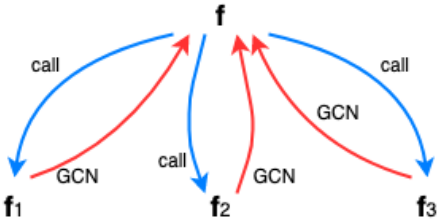


Fig. 2: The blue edges show which methods are called by f. In order to be able to display the functionality of f as accurately as possible, f must be enriched with the information of all called methods. The GCN (red edges) aggregates the features of the called methods up to f.

To train the GCN, the method's code is extracted from the caller graphs. To apply the propagation rule from Equation 1, the code is entered as H. The caller graph's adjacency matrix is used as A. As loss function, the Triplet Margin Loss is derived from Equation 4 is used, where it takes the output of the GCN as anchor, its corresponding docstring embedding as positive example and the docstring embedding of a random sample as negative example. In this way, the net tries to approximate the natural language description of a method and its code. We trained the GCN on 4.292 caller graphs (see Table 1), 100 iterations each, using a learning rate of 0.004 and Adam optimizer. To generate the embeddings, the Neural Bag-of-Words from Subsection 3.2 has been used.

4 Caller Graph Generation

We generated caller graphs for 4.197 GitHub repositories corresponding to the CodeSearchNet Challenge's Java subset (Fig. 3 shows a small visualized subgraph). 3.787 caller graphs are used for training, 189 for testing and 221 for validation of our GCN model (see also Table 1). The distribution follows the CodeSearchNet Challenge's original split.

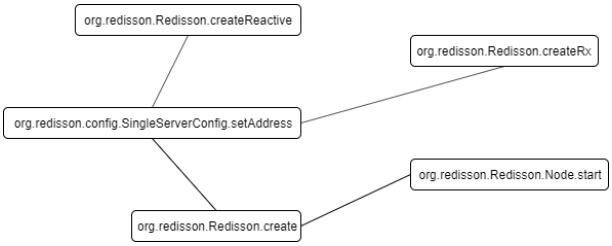


Fig. 3: Section of the caller graph generated from a repository called *redisson*. Rectangles symbolize methods and edges symbolize calls. We consider caller graphs as undirected graphs.

We reused the (comment, code) pairs provided in the CodeSearchNet Challenge's Java subset. Additionally, we extracted code fragments from the corresponding GitHub repositories to ensure that the caller graphs are as deep as possible to get the most benefit from using a GCN. Note that by doing this, we did not filter for the quality criteria used by GitHub for creating the CodeSearchNet corpus (e.g. removing test methods) [Hu19]. The additional data only serves to enhance the CodeSearchNet Challenge's methods. No additional (comment, code) pairs have been added for training.

Because the reused (comment, code) pairs are provided in a preprocessed way, the additionally extracted code fragments had to be preprocessed, too. This has been done by applying camelcase splitting, stopword filtering and lowercasing. We observed a huge amount of non-latin characters in Unicode representation in both the provided (comment, code) pairs and the additionally extracted code fragments and therefore transliterated all data into ASCII representations using *Unidecode* , a Python library.

4.1 Pipeline

Our caller graph generation pipeline consists of the following steps:

1. **Repository crawling** – The repositories are cloned in the version of the commit sha contained in the CodeSearchNet Challenge. (comment, code) pairs of one project are always of the same revision.
2. **Code and comment extraction** – Additional code and comments are extracted from the crawled repository using the *TreeSitter* Java parser.
3. **Generate individual caller graphs** – For each method, a caller graph is generated using *Doxygen*, a software documentation tool. Each caller graph contains the fully qualified method name and its full call hierarchy, representing each referenced method as a node.
4. **Caller graph assembly** – Combining the separate caller graphs of one repository's methods results in a complete representation of its call hierarchy. Additionally, each node is enriched by the corresponding code and comment.

Finally, a vocabulary is extracted containing all code and docstring tokens over all assembled caller graphs.

Overall, we assembled 4.197 caller graphs (see Table 1) containing 1.954.908 methods. Table 2 shows the composition in more detail. It should again be noted that only the docstrings of the 213.203 methods from the CodeSearchNet Challenge have been used to train the GCN. Self-extracted docstrings have not been used.

https://pypi.org/project/Unidecode/
https://tree-sitter.github.io/tree-sitter/

	Train	Test	Valid	Overall
Full CodeSearchNet Dataset	4.292	239	238	4.769
Usable in GCN	3.787	189	221	4.197

Tab. 1: Table showing how many repositories are provided in the full CodeSearchNet Challenge's Java subset and how many could actually made usable for the GCN.

	Train	Test	Valid	Overall
Methods in caller graph	**1.798.287**	**64.674**	**91.947**	**1.954.908**
Methods in caller graph (provided by CodeSearchNet Challenge)	198.092	7.605	7.506	213.203
Methods in caller graph (self-extracted)	1.600.195	57.069	84.441	1.741.705
Methods in caller graph without docstring	1.182.338	44.028	70.733	1.297.099
Methods in caller graph with docstring ((comment, code) pairs)	615.949	20.646	21.214	657.809

Tab. 2: Table showing the final composition of our generated caller graphs under two aspects: Contained in the CodeSearchNet Challenge / self-extracted and without docstring / with docstring.

4.2 Problems during Caller Graph Generation

During the generation of the caller graphs using our pipeline, several problems occured in steps 2 and 3, primarily in the context of encoding processed java files. The usage of non-standard names led to invalid caller graphs that could not be used. Overall, this resulted in a loss of 305 repositories (see Table 1).

5 Experiments

Faced with the CodeSearchNet Challenge [Hu19], we conducted several experiments. The challenge is about finding the correct code function among 1000 different code snippets using a given docstring. For this purpose, we implemented different Bag-of-Words models as well as Neural Bag-of-Words models and compared them to each other. We also conducted experiments with the Graph Convolutional Network regarding the challenge. To compare the results to the CodeSearchNet Challenge baseline, we used the test det of the CodeSearchNet Corpus for the evaluation, although the GCN approach could only use it partially for the evaluation (see Table 1).

5.1 BoW vs. Neural BoW

The Bag-of-Words and Neural Bag-of-Words approaches are based on training on the train set of the CodeSearchNet Corpus [Hu19]. For experiments with the Bag-of-Words encoder a vocabulary is necessary, which has been created from the docstring tokens and code tokens of the train set. The preprocessing of the tokens is explained in section 3.1. In our experiments, we created the vocabulary only from tokens that occurred at least twice in

the corpus. Ignoring stopwords, this resulted in a vocabulary size of 87,582 for Java and 140,002 for Python. The BoW encoder thus received an MRR of 31.55% for Java and 34.67% for Python, see Table 3. To improve our results, we gave the Bag-of-Words encoder TF-IDF weighting (BoW TF-IDF), which resulted in a significant increase of 20% for Java and Python. In this case, ignoring stopwords has no longer been necessary and therefore the vocabulary size was 87,633 for Java and 140,053 for Python. At that time we already had comparable results to the Neural Bag-of-Words results of the CodeSearchNet Challenge (NBoW CSNC [Hu19]).

Another experiment was the creation of the vocabulary using Byte-Pair Encoding (BoW BPE), which in combination with TF-IDF brought an improvement of one to two percent. Here the size of the model vocabulary generated by training was set to 5000 and the tokens were encoded using the model. Afterwards, the TF-IDF algorithm was applied. With the BPE encoding of the tokens the vocabulary size for Java was 19,781 and for Python 15,706.

Our next experiment consisted of a Neural Bag-of-Words encoder with an embedding dimension of 700, a learning rate of 0.0003, a batch size of 512 and Adam optimizer. With TF-IDF weighting and the byte-pair encoded vocabulary (NBoW TF-IDF BPE), we achieved our best result with an MRR of 56.38% for Java and 62.47% for Python, an improvement of 4.98% for Java and 4.38% for Python over the currently published Neural BoW model of the CodeSearchNet Challenge [Hu19]. Nevertheless, the CodeSearchNet Challenge baseline remains best with their attention-based model (Self-Att CSNC in table 3). Yet, including TF-IDF weighting may be an interesting approach for other neural models, too.

Encoder	MRR in percentage	
Text and Code	Java	Python
SelfAtt CSNC [Hu19]	58.66	69.22
NBoW CSNC [Hu19]	51.40	58.09
BoW	31.71	34.85
BoW BPE	29.46	28.94
BoW TF-IDF	51.52	58.38
BoW TF-IDF BPE	53.83	59.43
NBoW	51.24	55.49
NBoW TF-IDF	54.59	60.94
NBoW TF-IDF BPE	**56.38**	**62.47**

Tab. 3: Mean Reciprocal Rank (MRR) on the test set of CodeSearchNet Corpus in comparison to the baseline results of the CodeSearchNet Challenge (CSNC) [Hu19]. This is the evaluation of the CSNC training task, where the models try to rank the correct code snippet among 999 distractor snippets, when the documentation comment is given as a query.

That the matching of the Neural BoW compared to BoW has become more fuzzy and therefore better can be seen more clearly in the following example: The sample with the docstring tokens "*move bytes left or right of an offset* ." has the following code tokens:

```
public void move ( final int i from , final int i position ) {
     if ( i position = = 0 ) return ;
     final int to = i from + i position ;
     final int size = i position > 0 ? buffer . length - to : buffer . length - i from ;
     system . arraycopy ( buffer , i from , buffer , to , size ) ;
}
```

Listing 1: Code tokens of a Java function that shifts bytes left or right from an offset

The tokens *left* and *right* of the docstring do not appear in the code, but instead the token *position*. With the BoW TF-IDF BPE encoder the code was ranked at position 43. This is due to the fact that the tokens *left*, *right* and *position* are evaluated in the vocabulary as completely different tokens, which have no relation to each other. Only the dot and the token *move* count as real matches. In contrast, the NBoW TF-IDF BPE encoder ranked the code at position 5, which shows that semantically similar tokens like *position*, *left* and *right* have been learned while training.

5.2 GCN

To measure the quality of the GCN-based approach, we exclusively used the Java data sets from the CodeSearchNet Challenge. A caller graph was generated from each project, according to the process from Section 4, and trained on it. The evaluation is also based

Encoder	MRR in percentage
Text and Code	Java (GCN)
BoW	29.90
BoW BPE	28.73
BoW TF-IDF	49.27
BoW TF-IDF BPE	52.46
NBoW	48.81
NBoW TF-IDF	54.49
NBoW TF-IDF BPE	**55.01**
GCN	27.15

Tab. 4: Comparison with GCN on Java test data

exclusively on the Java test data from the CodeSearchNet Challenge. The exact procedure of the evaluation and calculation of the result is described in subsection 5.1.

Table 4 shows the results of the GCN-based approach compared to the previous approach and the CodeSearchNet Challenge results. Unfortunately, at 27,15%, the GCN result is the worst score in this comparison.

6 Conclusion

In this paper, two approaches on the task of semantic code search have been presented. The first one is based on a Neural Bag-of-Words encoder. Different variations have been tried out with TF-IDF weighting and byte-pair encoding. The best model, which used TF-IDF and byte-pair encoding, achieved an MRR of 56.38% on the Java data set and an MRR of 62.47% on the Python data set. This exceeds the Neural Bag-of-Words approach of the Code-Search-Net Challenge [Hu19], but the attention based-model of the Code-Search-Net Challenge could not be exceeded so far. The other approach dealt with semantic code search based on functional dependencies. A caller graph has been created for the search in a repository. This served as input for the GCN. The usage of the GCN, however, showed no improvement.

References

[CKC] Cambronero, J.; Kim, S.; Chandra, S.: When Deep Learning Met Code Search./, pp. 964–974.

[De20] Developers, S.: Feature extraction Tfidf-Term-Weighting — scikit-learn 0.22.2 documentation, 2020, URL: https://scikit-learn.org/stable/modules/feature_extraction.html#tfidf-term-weighting.

[Hu19] Husain, H.; Wu, H.-H.; Gazit, T.; Allamanis, M.; Brockschmidt, M.: CodeSearchNet Challenge: Evaluating the State of Semantic Code Search, 2019, arXiv: 1909.09436 [cs.LG].

[Ka19] Kanade, A.; Maniatis, P.; Balakrishnan, G.; Shi, K.: Pre-trained Contextual Embedding of Source Code./, pp. 1–22, 2019, arXiv: 2001.00059, URL: http://arxiv.org/abs/2001.00059.

[KW16] Kipf, T. N.; Welling, M.: Semi-Supervised Classification with Graph Convolutional Networks, 2016, arXiv: 1609.02907 [cs.LG].

[NGV08] Narayan, G.; Gopinath, K.; Varadarajan, S.: Structure and Interpretation of Computer Programs. 2008 2nd IFIP/IEEE International Symposium on Theoretical Aspects of Software Engineering/, June 2008, URL: http://dx.doi.org/10.1109/TASE.2008.40.

[Re09] Reiss, S. P.: Semantics-based code search. Proceedings - International Conference on Software Engineering/, pp. 243–253, 2009, ISSN: 02705257.

[Sa18] Sachdev, S. S.; Li, H. H.; Luan, S. S.; Kim, S. S.; Sen, K. K.; Chandra, S. S.: Retrieval on source code: A neural code search. MAPL 2018 - Proceedings of the 2nd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages, co-located with PLDI 2018/, pp. 31–41, 2018.

[VM16] Vassileios Balntas Edgar Riba, D. P.; Mikolajczyk, K.: Learning local feature descriptors with triplets and shallow convolutional neural networks. In (Richard C. Wilson, E. R. H.; Smith, W. A. P., eds.): Proceedings of the British Machine Vision Conference (BMVC). BMVA Press, pp. 119.1–119.11, Sept. 2016, ISBN: 1-901725-59-6, URL: https://dx.doi.org/10.5244/C.30.119.

[Wa19] Wan, Y.; Shu, J.; Sui, Y.; Xu, G.; Zhao, Z.; Wu, J.; Yu, P.: Multi-modal attention network learning for semantic source code retrieval. Proceedings - 2019 34th IEEE/ACM International Conference on Automated Software Engineering, ASE 2019/, pp. 13–25, 2019.

Quantitative comparison of polarity lexicons in sentiment analysis tasks: Using a lexicon overlap score for similarity measurement between lexicons

Felix J. M. Welter[1]

Abstract: Sentiment classification is either based on sentiment lexicons or machine learning. For the construction and improvement of sentiment lexicons, several approaches and algorithms have been designed. The resulting lexicons are commonly benchmarked in different tasks and compared by their respective performance. However, this measure depends on the application domain. This work proposes a method for context-independent comparison of sentiment lexicons. Three scoring methods for similarity measurement of lexicons are explained. Furthermore, exemplarily applications of the scores are shown, including lexicon similarity analysis before and after expansion via a Distributional Thesaurus and clustering of lexicons. Adaptability and limitations of the lexicon overlap score and the demonstrated applications are discussed.

Keywords: sentiment analysis, sentiment classification, semantic orientation, sentiment lexicons, polarity lexicons, lexicon similarity, lexicon distance, lexicon overlap score

1 Introduction

Sentiment analysis is a field of Natural Language Processing (NLP) and focuses on the extraction of polarity and opinion targets in texts of any size. With the growing amount of possibilities for individuals to share content online, opinion-rich resources become available. Since they are heavily used by customers for decision making and often contain patterns and signals about customer behavior, organizations have a reasonable interest in employing advanced text mining techniques to extract valuable information. [PL08] For deeper insights, sentiment analysis can be combined with other NLP techniques such as text summarization. [To01] Application domains of sentiment analysis include, but are not limited to, political speeches [GST16], web advertising [AA16], [Qi10], and social media [VS16].

This work revolves around the subtask of sentiment classification, which involves the detection of the semantic polarity of a piece of text. Classification is most often either binary (positive or negative), ternary (positive, neutral or negative) or 5-scale ordinal (often visualized by using stars). [RFN17] It should be noted that the task of sentiment classification is generally specific to a context. The polarity of words varies between topics, domains and also changes historically. [Re05], [Ha16] Further challenges are imposed by misspellings [Mu13], negations [Po10], emoticons [Re05], multi word

[1] Universität Hamburg, Vogt-Kölln-Straße 30, 22527 Hamburg, felix.welter@studium.uni-hamburg.de

cc by sa

expressions [Sa02], [WWH05], idioms [Jo18], [NSW94] and language specific characteristics [ACS08].

For sentiment classification, one can generally use two approaches. While they are mostly used independently, they can also be combined. [Zh11] The first approach is based on a lexicon. This involves the creation of a list of words where each word has a semantic orientation assigned. The polarity of a piece of text is then determined by the contained words and the corresponding scores. The second approach is based on machine learning classifiers, e.g. Naïve Bayes, Support Vector Machines or Neural Networks. Features, such as contained words, part-of-speech tags, and punctuation, are extracted from the text and the classifier is trained to predict the polarity. [VS16]

The following paper focuses on the first approach. Current methods for lexicon creation contribute to an increase in lexicon size. Furthermore, the construction of lexicons becomes highly automated. [Ku16], [Ta14], [TL02], [Ve10], [YS14] This is excellent since most applications should construct a domain specific sentiment lexicon and the manual creation or construction involving crowd work is expensive. [MT10] However, this poses the challenge that many lexicons are available, which are only different in the construction algorithm, the underlying dataset and the performance in the target domain. A trivial method to compare lexicons without these context factors is missing.

This work presents a possibility to compare different sentiment lexicons solely based on their content. Three different but closely related scores are described. Lexicon size, common words and the similarity of the semantic orientation of each word are used for computation. No external resources are required, and the score is language independent. Example applications of the proposed lexicon overlap score are given.

2 Related Work

Since human-made sentiment lexicons yield poor results, [PLV02] research in lexicon construction for sentiment classification designed different algorithms for automatic lexicon construction. Hatzivassiloglou and McKeown [HM97] extracted words of the same polarity by searching for adjectives connected by the word *and*. Adjectives connected by *but* were supposedly of opposite polarity. Retrieving a multitude of these colocations from a corpus and clustering found words into two groups enabled them to construct a positive and a negative word list.

Turney [Tu02] designed a method where phrases containing adjectives were scored based on their cooccurrence with the words *excellent* and *poor*. For both cases the Point Wise Mutual Information (PMI) is calculated and the semantic polarity of a phrase is defined by the difference between the PMI with *excellent* and the PMI with *good*.

More recent work has put a focus on graph-based approaches. Relations between words or phrases are captured based on linguistic resources like WordNet [Pr10] or inferred

from corpora. Semantic orientation is then propagated between words which results in a polarity score for each word in the graph. This technique is employed in the Social Sent Project [Ha16] and by [Ve10].

Another approach to enriching sentiment lexicons is based on a Distributional Thesaurus, which is a lexical resource that contains information about semantically similar words. For highly positive or negative words or phrases, closely related terms can be found. By adding these terms to the sentiment lexicon, a higher coverage and therefore a higher sentiment classification performance can be achieved. [Ku16]

3 Lexicon overlap score

With the increase of automated lexicon creation methods for different languages and an upsurge in lexicon size a formalized metric for lexicon comparison is needed. This score should be language independent and solely taking the lexicon content into account. The proposed lexicon overlap score (LOS) takes into account lexicon size, presence of words, semantic orientation and the strength of the semantic orientation. Three different versions of the score exist. Since they are based on each other, they will be explained in order with rising complexity. Especially the first score is closely related to the widely known Jaccard index, sometimes also called Tanimoto similarity. All scores have in common that identical lexicons have a LOS of 1 and lexicons without mutual words have a score of 0. Furthermore, it is assumed that semantic orientation of a word larger than zero is interpreted as positive and semantic orientation smaller than zero interpreted as negative, as it is used by most algorithms.

3.1 Simple LOS

The simple version of the lexicons overlap score takes into account the size of both lexicons and the words they have in common. A denotes the size of the first lexicon, B the size of the second lexicon and n the count of common words, then the simple LOS is expressed by the following.

$$\frac{n}{A + B - n}$$

The simple LOS states to what extent the lexicons share a common vocabulary. It does not quantify the relation between the polarity strength of each word in the lexicons, not even about the general direction of the polarity. Depending on the lexicon creation algorithm that was used, it can be applied to find shared words of different domains. Given a fixed domain it can be used to quantify to what extent different algorithms extract a mutual vocabulary from a dataset.

3.2 Binary LOS

Building upon the simple LOS, the binary LOS additionally draws a distinction between positive and negative polarity of words. Words are only counted for the numerator if they are both positive or both negative and therefore share the same general semantic orientation. A denotes the size of the first lexicon, B the size of the second lexicon and n the count of common words. Furthermore, each word that both lexicons include, denoted by $w_1, \dots, w_n$, is modified by the function X which evaluates to one, if the polarities share the same sign, otherwise to zero. Given a word w the function S_a evaluates to the polarity score of the first lexicon and S_b evaluates to the polarity score of the second lexicon. The binary LOS is then expressed by the following.

$$\frac{\sum_{i=1}^{n} X(w_i)}{A + B - n}$$

and

$$X(w) = \begin{cases} 1, & 0 < S_a(w) * S_b(w) \\ 0, & else \end{cases}$$

Since positivity and negativity of single words can strongly vary between domains (*crazy* in sports and *crazy* in relationships [Ha16]) it should be assured that only words are counted that are similarly quantified by both lexicons. This is especially helpful for comparison of lexicons that have different underlying algorithms that in general assign strongly diverging scores. Alternatively, polarity scores can be normalized or ranked and then compared by using the general LOS, which is explained in the following part. Finally, depending on the context it may be beneficial to put a stronger punishment on words with opposite polarity. In that case, X could be modified to return -1 instead of zero for these words.

3.3 General LOS

Building upon the binary LOS, the general LOS not only takes the direction of each word polarity into account, but instead weighs them by their relative similarity. A denotes the size of the first lexicon, B the size of the second lexicon and n the count of common words. For each word present in both lexicons $w_1, \dots, w_n$ the relative similarity is calculated by the function Y. The case where both scores are equal is individually treated to correctly process zero values. Y can evaluate to negative values, which occurs whenever in one lexicon the word in question is assigned a positive value and in the other it is assigned a negative value. S_a and S_b return the polarity for the given word from their respective lexicon. The general LOS is then expressed by the following.

$$\frac{\sum_{i=1}^{n} Y(w_i)}{A + B - n}$$

and

$$Y(w) = \begin{cases} 1 \quad , & S_a(w) = S_b(w) \\ \dfrac{S_a(w)}{S_b(w)}, & |S_a(w)| < |S_b(w)| \\ \dfrac{S_b(w)}{S_a(w)}, & else \end{cases}$$

By utilizing the relative similarity of the polarities of each word, the general LOS is the most precise score of the ones proposed in this work. By default, it should be used for lexicon comparison. Regarding the general LOS, lexicons can also receive a negative score. This will practically occur if two lexicons contain several words that have opposite polarities assigned. However, it is more challenging to interpret compared to the simple and binary LOS. Without reference scores, the result is of low value. The following section will therefore elaborate on usage examples.

4 Example Applications

This section shows applications of the lexicon overlap score. LOS distributions are shown, and interpretation is demonstrated. For the computations numpy [vCV11], pandas [Mc10], matplotlib [Hu07] and scikit-learn [Pe11] were employed. This will additionally evaluate the utility and validity of the score for selected example usages. Generally, in this context sentiment lexicons should be perceived as entities. The lexicon overlap score represents the similarity between two entities and therefore acts as a distance measure between lexicons.

For the demonstrated applications, sentiment lexicons for different domains from the internet platform reddit are used, which are available via the SocialSent project. Reddit is a website with a focus on content generated by individuals and user interaction. Topics and domains have separated spaces where interaction takes place. Based on these so-called subreddits 250 sentiment lexicons have been created and made available by the authors of the SocialSent project. Most of these lexicons contain between 4000 and 5000 words. [Ha16]

4.1 LOS Distributions

First, distributions of the lexicon overlap scores are shown. Pairwise distances between all lexicons were computed, employing all three LOS variations. Density was estimated by Kernel Density Estimation. [Si86] In Fig. 1 it is evident that the simple LOS yields higher values than the binary LOS and the binary LOS yields higher numbers than the general LOS overall. Furthermore, negative general LOS values can be observed. As

explained, this occurs if two lexicons contain several words that have opposite polarities assigned.

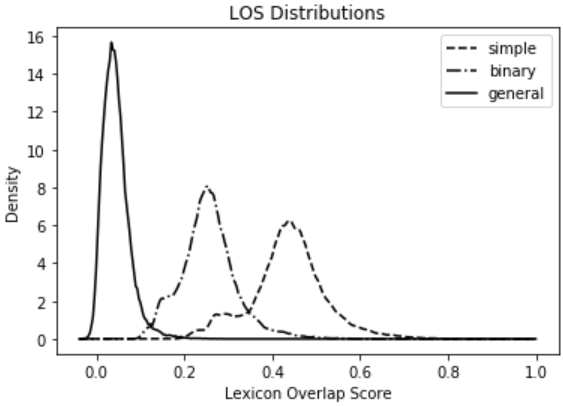


Fig. 1: Distributions of lexicon overlap scores

Selected comparisons between lexicons with all three lexicon overlap scores are displayed in Tab. 1. An extraordinary high similarity can be observed between *relationship_advice* and *relationships*, which is expectable due to strongly related topics. The second row shows a rather large common vocabulary (see simple LOS) for the subreddits *fantasyfootball* and *nba*, however the general LOS is close to zero, indicating that several common words have an opposite semantic orientation. The difference in semantic orientation between the lexicons *tipofmytongue* and *nba* is of such distinction that a negative general LOS is assigned. The last row shows two lexicons that have remarkably few words in common.

subreddit-specific lexicons		simple	binary	general
relationship_advice	relationships	0.901	0.708	0.342
fantasyfootball	nba	0.501	0.212	0.030
tipofmytongue	nba	0.343	0.145	-0.031
CasualPokemonTrades	askscience	0.190	0.102	0.008

Tab. 1: All lexicon overlap scores for selected comparisons.

4.2 Impact of lexicon enrichment via Distributional Thesaurus

Generally, the process of lexicon construction does not need to consist of only one step. Having multiple iterations for the creation of well performing lexicons is not uncommon. Kumar et al. [Ku16] used a Distributional Thesaurus (DT) to expand existing lexicons with semantically similar words and polarity scores. The lexicon overlap score can be

used to analyze the similarity of the treated lexicons before and after the process. This can answer the question, if the expansion using a Distributional Thesaurus leads to lexicons that become even more domain specific or if the enrichment process results in more similar lexicons.

To answer this question, the DT approach was applied to selected lexicons from the SocialSent project. For this purpose, the API of the JoBimText project [BR13], [Gl13] was accessed via the Java-based LexiExp Tool[2]. Due to computational limitations, only a small subset of seven randomly selected lexicons was processed. This application example uses the binary LOS, since scores vary considerably between the SocialSent lexicons and the results produced by the DT approach. For each pair of lexicons, the binary LOS was calculated before and after DT expansion. In Fig. 2 both distributions are shown. The expanded lexicons are generally more similar. Analysis of the change in similarity per lexicon pair shows that the mean increase is 0.04, while the minimum is 0.02 and the maximum 0.05, expressing that the lexicon expansion increases lexicon similarity in all cases. This is supposedly since the DT expansion adds words that are semantically similar but not specifically from the same domain, therefore creating a larger common vocabulary.

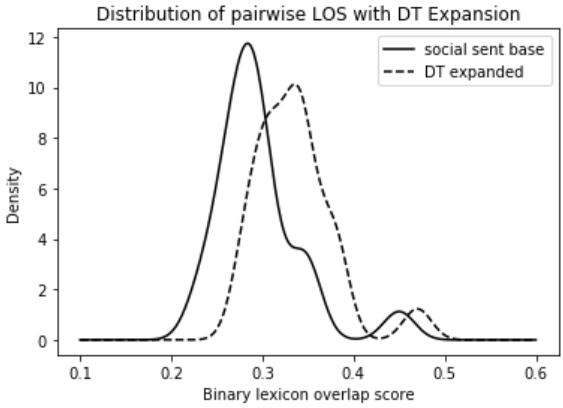


Fig. 2: Distribution of pairwise lexicon overlap scores before and after DT expansion

4.3 Lexicon clustering

Another possible application for the lexicon overlap score is the clustering of different sentiment lexicons based on their similarity. An affinity matrix can be constructed from the pairwise lexicon overlap scores which can be used as input for a clustering algorithm. The selection of the algorithm is restricted by the fact, that the lexicons have no values that indicate a location in an n-dimensional space (e.g. coordinates), which is

[2] https://github.com/uhh-lt/LexiExp

required by a fair amount of clustering approaches. In this case, Spectral Clustering [Vo07] was used.

The main parameter for Spectral Clustering is the number of clusters k. In the context of sentiment lexicons, it could be argued that the number of clusters should be set as the expected number of broader domains covered by the sentiment lexicons. This however is difficult to estimate. Therefore, the clustering algorithm was executed several times. It was found that with a high number of clusters in relation to the count of lexicons, groups of lexicons from similar domains were clustered.

Exemplarily results of the clustering (k=100) are shown in Tab. 2. Each cluster was manually annotated with a label which describes the commonalities of the contained lexicons. Only larger clusters are shown in the table. Lexicons in smaller clusters can be considered outliers in a sense that they do not belong in one category with other lexicons. Examples of these lexical outliers are *britishproblems, legaladvice, gameofthrones, nosleep, thewalkingdead, MakeupAddiction* and *BigBrother*. However, overall a negative silhouette coefficients [Ro87] suggests that clusters are overlapping and no clear distinction between lexicon clusters is possible.

Cluster Description	**Contained lexicons**
Trading in games	ACTrade, SVExchange, friendsafari
Technology	Android, buildapc, jailbreak, windowsphone
Family	AskWomen, Parenting, AskMen, BabyBumps
Competitive multiplayer games	CoDCompetitive, DotA2, GlobalOffensive, csgobetting, leagueoflegends, starcraft
IT jobs	programming, sysadmin, talesfromtechsupport
Fantasy multiplayer games	DestinyTheGame, Guildwars2, WildStar, archeage, elderscrollsonline, wow
Pokemon	CasualPokemonTrades, pokemon, pokemontrades, twitchplayspokemon
Religion	Christianity, DebateReligion, atheism, exmormon

Tab. 2: Clustered lexicons with human-annotated descriptions

Aside from the demonstrated examples, a range of further applications is possible. While this work focused on lexicons from different domains and constructed with the same algorithm, the comparison of lexicons from the same domain or dataset but constructed with different algorithms could yield interesting insights as well. The following chapter will discuss the possibilities of this approach in addition to the utility and expressiveness of the lexicon overlap score.

5 Discussion

The goal of the lexicon overlap score is to make sentiment lexicons comparable independent of context. The shown examples demonstrate different possibilities of applications. While the shown cases analyzed several lexicons constructed with the same approach, the lexicon overlap score can be used to compare lexicons from different approaches as well. As an example, it could be investigated if graph-based approaches yield lexicons that are more similar to each other compared to approaches based on lexical resources. Furthermore, it could be relevant to adjust the lexicon overlap score to take inflection into account when analyzing different lexicon construction methods.

As an incidental remark, a different approach to lexicon comparison could be based on the following. Lexicons can be converted to vectors, where each dimension corresponds to a word. Optionally dimensions could be reduced by application of a principal component analysis. The lexicons can then be compared by the cosine distance of the vectors. For the clustering approach as demonstrated in 4.3 this would offer coordinates instead of similarity scores, therefore enabling the usage of a wider range of clustering algorithms. However, a downside of this approach is the need to specify a mapping between words and dimensions, which reduces the ease by which new lexicons can be compared. The lexicon overlap score on the other hand side is not restricted in that way.

The sentiment lexicons used in this work were constructed in a way, that positive words have a value with a positive sign and negative words have a value with a negative sign. This does not need to be the case for every lexicon construction approach. The lexicon overlap score must be adjusted for lexicons where negative words carry a zero and positive words a one, and other cases alike. Furthermore, occasions of idioms, negations and multi word phrases have not been covered. Generally, the lexicon overlap score can be applied to sentiment lexicons which include said items as well. However, possibly it needs to be adjusted to correctly reflect cases where a word in the first lexicon occurs in a multi word phrase in the second lexicon and similar cases.

On an abstract level the lexicon overlap score can be adapted to any data structure which contains categorical items with assigned scores. Therefore, the application in different scenarios and research fields is possible as well. Generally, it should be noted that LOS values are best evaluated in comparison, since a single lexicon overlap score is challenging to interpret.

6 Conclusion

This work proposed a score for similarity measurement of sentiment lexicons for the task of sentiment classification. The three lexicon overlap scores have been implemented in python and made available as a pip package[3] for easy installation and usage in python

[3] https://pypi.org/project/lexicon-overlap-score/

projects. Several examples of applications have been shown to demonstrate the utility in different scenarios and underline the validity and significance.

The lexicon overlap score is independent of any domain, context and language and therefore can be easily applied. Aside from that, its independence from a specific set of classification items makes it faster to compute than performance benchmarks. In the end, it will help with objective quantification and more accurate descriptions of sentiment lexicons and therefore enables a better understanding of connections between lexicon construction methods as well as word polarity in different domains.

References

[AA16] Adamov, A. Z.; Adali, E.: Opinion mining and Sentiment Analysis for contextual online-advertisement: Application of Information and Communication Technologies - AICT2016. 12-14 October 2016, Baku, Azerbaijan conference proceedings. IEEE, Piscataway, NJ, 2016; S. 1–3.

[ACS08] Abbasi, A.; Chen, H.; Salem, A.: Sentiment analysis in multiple languages: Feature selection for opinion classification in web forums. In ACM Transactions on Information Systems (TOIS), 2008, 26; S. 1–34.

[BR13] Biemann, C.; Riedl, M.: Text: now in 2D! A framework for lexical expansion with contextual similarity. In Journal of Language Modelling, 2013, 1; S. 55.

[Gl13] Gliozzo, A. et al.: JoBimText Visualizer: A Graph-based Approach to Contextualizing Distributional Similarity: Proceedings of the 8th Workshop on TextGraphs in conjunction with EMNLP 2013, 2013; S. 6–10.

[GST16] Gentzkow, M.; Shapiro, J.; Taddy, M.: Measuring Polarization in High-Dimensional Data: Method and Application to Congressional Speech. Working Papers, 2016.

[Ha16] Hamilton, W. L. et al.: Inducing Domain-Specific Sentiment Lexicons from Unlabeled Corpora, 2016.

[HM97] Hatzivassiloglou, V.; McKeown, K. R.: Predicting the semantic orientation of adjectives. In (Cohen, P. R.; Wahlster, W. Hrsg.): Proceedings of the 35th annual meeting on Association for Computational Linguistics -. Association for Computational Linguistics, Morristown, NJ, USA, 1997; S. 174–181.

[Hu07] Hunter, J. D.: Matplotlib: A 2D Graphics Environment. In Computing in Science & Engineering, 2007, 9; S. 90–95.

[Jo18] Jochim, C. et al.: SLIDE – a Sentiment Lexicon of Common Idioms, 2018.

[Ku16] Kumar, A. et al.: IIT-TUDA at SemEval-2016 Task 5: Beyond Sentiment Lexicon: Combining Domain Dependency and Distributional Semantics Features for Aspect Based Sentiment Analysis. In (Bethard, S. et al. Hrsg.): Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016). Association for Computational Linguistics, Stroudsburg, PA, USA, 2016; S. 1129–1135.

[Mc10] McKinney, W.: Data Structures for Statistical Computing in Python: Proceedings of the 9th Python in Science Conference. SciPy, 2010; S. 56–61.

[MT10] Mohammad, S.; Turney, P.: Emotions Evoked by Common Words and Phrases: Using Mechanical Turk to Create an Emotion Lexicon. In (Association for Computational Linguistics Hrsg.): Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text, 2010; S. 26–34.

[Mu13] Muhammad, A. et al.: Contextual Sentiment Analysis in Social Media Using High-Coverage Lexicon. In (Bramer, M.; Petridis, M. Hrsg.): Research and Development in Intelligent Systems XXX. Incorporating Applications and Innovations in Intelligent Systems XXI Proceedings of AI-2013, The Thirty-third SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence. Springer International Publishing, Cham, s.l., 2013; S. 79–93.

[NSW94] Nunberg, G.; Sag, I. A.; Wasow, T.: Idioms. In Language, 1994, 70; S. 491–538.

[Pe11] Pedregosa, F. et al.: Scikit-learn: Machine Learning in Python. In Journal of Machine Learning Research, 2011, 12; S. 2825–2830.

[PL08] Pang, B.; Lee, L.: Opinion Mining and Sentiment Analysis. In Foundations and Trends® in Information Retrieval, 2008, 2; S. 1–135.

[PLV02] Pang, B.; Lee, L.; Vaithyanathan, S.: Thumbs up? Sentiment Classification using Machine Learning Techniques. In (Unknown Hrsg.): Proceedings of the ACL-02 conference on Empirical methods in natural language processing - EMNLP '02. Association for Computational Linguistics, Morristown, NJ, USA, 2002; S. 79–86.

[Po10] Potts, C.: On the negativity of negation. In Semantics and Linguistic Theory, 2010, 20; S. 636.

[Pr10] Princeton University: About WordNet. https://wordnet.princeton.edu/, 09.04.2020.

[Qi10] Qiu, G. et al.: DASA: Dissatisfaction-oriented Advertising based on Sentiment Analysis. In Expert Systems with Applications, 2010, 37; S. 6182–6191.

[Re05] Read, J.: Using Emoticons to Reduce Dependency in Machine Learning Techniques for Sentiment Classification: Proceedings of the ACL Student Research Workshop. Association for Computational Linguistics, USA, 2005; S. 43–48.

[RFN17] Rosenthal, S.; Farra, N.; Nakov, P.: SemEval-2017 Task 4: Sentiment Analysis in Twitter. In (Bethard, S. et al. Hrsg.): Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017). Association for Computational Linguistics, Stroudsburg, PA, USA, 2017; S. 502–518.

[Ro87] Rousseeuw, P. J.: Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. In Journal of Computational and Applied Mathematics, 1987, 20; S. 53–65.

[Sa02] Sag, I. A. et al.: Multiword Expressions: A Pain in the Neck for NLP. In (Gelbukh, A. Hrsg.): Computational Linguistics and Intelligent Text Processing. Third International Conference, CICLing 2002 Mexico City, Mexico, February 17-23, 2002 Proceedings. Springer, Berlin, Heidelberg, 2002; S. 1–15.

[Si86] Silverman, B. W.: Density estimation for statistics and data analysis. CRC press, 1986.

[Ta14] Tang, D. et al.: Building large-scale twitter-specific sentiment lexicon: A representation learning approach: Proceedings of coling 2014, the 25th international conference on computational linguistics: Technical papers, 2014; S. 172–182.

[TL02] Turney, P. D.; Littman, M. L.: Unsupervised Learning of Semantic Orientation from a Hundred-Billion-Word Corpus, 2002.

[To01] Tong, R. M.: An Operational System for Detecting and Tracking Opinions in On-line Discussion: Proceedings of the Workshop on Operational Text Classification (OTC), 2001.

[Tu02] Turney, P. D.: Thumbs up or thumbs down? In (Isabelle, P. Hrsg.): Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02. Association for Computational Linguistics, Morristown, NJ, USA, 2002; S. 417.

[vCV11] van der Walt, S.; Colbert, S. C.; Varoquaux, G.: The NumPy Array: A Structure for Efficient Numerical Computation. In Computing in Science & Engineering, 2011, 13; S. 22–30.

[Ve10] Velikovich, L. et al.: The Viability of Web-Derived Polarity Lexicons: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, USA, 2010; S. 777–785.

[Vo07] Von Luxburg, U. Von Luxburg, U.: A Tutorial on Spectral Clustering, 2007.

[VS16] Vishal, A.; Sonawane, S.: Sentiment Analysis of Twitter Data: A Survey of Techniques. In International Journal of Computer Applications, 2016, 139; S. 5–15.

[WWH05] Wilson, T.; Wiebe, J.; Hoffmann, P.: Recognizing contextual polarity in phrase-level sentiment analysis: Proceedings of human language technology conference and conference on empirical methods in natural language processing, 2005; S. 347–354.

[YS14] Yanqing Chen; Steven Skiena: Building Sentiment Lexicons for All Major Languages: ACL, 2014.

[Zh11] Zhang, L. et al.: Combining Lexicon-based and Learning-based Methods for Twitter Sentiment Analysis, 2011.

Short Papers

Klassische Migräne vs. gewöhnliche Migräne: Untersuchung basierend auf Altersstruktur und Geschlecht

Christa Strauß[1] und Tina Amann[2]

Abstract: Diese Studie zielt darauf ab, klassische Migräne und gewöhnlicher Migräne zu vergleichen. Dabei wird auf Basis von online gesammelten Daten des Projektes "Migräne–Radar", die Altersstruktur und das biologische Geschlecht näher betrachtet, um Rückschlüsse auf Hormoneinflüsse treffen zu können. Mithilfe des Projekts "Migräne–Radar" des Instituts für Informationssysteme der Hochschule Hof wurden 66886 Migräneattacken von 2904 Betroffenen zu Verfügung gestellt. Die Beteiligten wurden in vier Altersgruppen eingeteilt. Anhand dieser Aufteilung können sowohl geschlechtsspezifische als auch altersbedingte Unterschiede zwischen den beiden Krankheitsbildern untersucht werden. Die Ergebnisse lassen vermuten, dass Migräneanfalle ohne Aura sowohl bei Männern als auch bei Frauen durch Hormone beeinflussbar sein können. Migräneanfälle mit Aura hingegen weniger.

Keywords: Migräne, gewöhnliche Migräne, klassische Migräne, Östrogene, Testosteron, Altersstruktur, Geschlecht

1 Motivation

In der Vergangenheit wurde das Thema „klassische Migräne und gewöhnliche Migräne“ bereits des Öfteren in verschiedenen wissenschaftlichen Papern beleuchtet. Unter der gewöhnlichen Migräne versteht sich dabei die etablierte Kopfschmerzerkrankung mit einer Attackendauer von 4 – 72 Stunden (Migräne ohne Aura). Klassische Migräne hingegen bezeichnet eine Erkrankung, bei der vor dem typischen Migränekopfschmerz reversible, wiederkehrende und anfallsartige neurologische Ausfälle auftreten (Migräne mit Aura). Diese können sich sowohl sensorisch als auch fokal auswirken. [Ga16a]

Vor allem im 20. Jahrhundert wurde diese Thematik stark diskutiert, woraus unterschiedliche Meinungen hervorgingen. Eine große Partei vertritt die Meinung, dass beide Krankheitsbilder einer Krankheit zugehörig sind. Die Opposition hingegen manifestiert die Aufteilung in zwei differenzierte Krankheiten. Beispielsweise schließt der Arzt Dr. Blau aufgrund derselben Schmerzdauer, Reaktion auf die Medikation und sowohl Prodrom als auch Postdrom der beiden Entitäten, dass zwischen diesen ein Zusammenhang besteht und sie somit eine gemeinsame Krankheit darstellen [Bl95].

[1] Hochschule für Angewandte Wissenschaften Hof, Alfons–Goppel–Platz 1, 95028 Hof, strausschrista98@gmx.de
[2] Institut für Informationssysteme der Hochschule für Angewandte Wissenschaften Hof, Forschungsgruppe Analytische Informationssysteme, Alfons–Goppel–Platz 1, 95028 Hof, tina8katharina8@web.de

Auf der anderen Seite behauptet zum Beispiel die Medizinerin Dr. Rasmussen, dass klassische Migräne und gewöhnliche Migräne zwei unterschiedliche Krankheiten darstellen, da eine Person nur fälschlicherweise davon ausgehen kann beide Arten gleichzeitig zu haben. Dies führt sie darauf zurück, dass die Aura auch unterbewusst stattfinden kann, beispielsweise im Schlaf, und dadurch nur der Kopfschmerz zu spüren ist, welcher fälschlicherweise als Migräne ohne Aura gedeutet werden kann. Zu dieser Annahme kam sie unter anderem durch die oftmals unterschiedliche Pathophysiologie der beiden Krankheitsbilder. [Ra95]

Des Weiteren äußert Rasmussen einen höheren Zusammenhang zwischen Hormonen und gewöhnlicher Migräne, weniger mit klassischer Migräne [Ra92a]. Aufgrund der Bandbreite an differenzierten Meinungen und der noch unklaren Ursachen von Migräne, soll im Rahmen dieser Arbeit Migräne mit (MA) und ohne Aura (MO) verglichen werden. Das Ziel ist es dabei, eine mögliche Ursache der beiden Migränearten zu finden, von welcher Spezialisten später therapeutische oder womöglich auch präventive Maßnahmen ableiten zu können. Dies geschieht anhand der Altersstruktur und des biologischen Geschlechts, um mögliche Zusammenhänge sowie Unterschiede analysieren zu können. Hierbei wird auch eine mögliche Rolle der (Geschlechts–) Hormone im Zusammenhang mit MA und MO abgewogen. Diese Analyse wird mit Datensätzen durchgeführt, welche durch das Projekt „Migräne Radar" des Instituts für Informationssysteme der Hochschule Hof zur Verfügung gestellt wurden.

2 Methoden

2.1 Das Migräne Radar Projekt

Mithilfe des im Jahr 2011 gestarteten bürgerwissenschaftlichen Projekts "Migräne Radar" des Instituts für Informationssysteme der Hochschule Hof werden Migränedaten von Projektteilnehmern gesammelt, welche zur Auswertung verschiedener wissenschaftlicher Arbeiten genutzt werden können. Dabei kommen besondere Datenerhebungsmethoden über projekteigene Smartphone– und Webapps zum Einsatz. Hierdurch können beispielsweise regelmäßig ausgefüllte Migräne–Tagebücher und Fragebögen für die Auswertung gewonnen werden.

2.2 Studiendesign

Anhand dieser Studie wird der Zusammenhang zwischen Migräne mit und ohne Aura im Hinblick auf Altersstruktur und biologischem Geschlecht der Patienten näher untersucht. Es soll anhand der Altersverteilung ermittelt werden, ob in Zeiten besonderer Hormonausschüttung Migräneattacken bei beiden Krankheitsbildern ähnliche mögliche Veränderungen aufweisen oder nicht. Dabei werden die Geschlechter differenziert, um

besser auf den Hormonhaushalt und auf das Aufkommen der beiden Migränekrankheiten in den entsprechenden Altersgruppen schließen zu können.

2.3 Teilnehmer

Die Daten, die für diese Untersuchung genutzt werden, wurden in dem Zeitraum vom 01. Januar 2015 bis zum 13. Juni 2019 gesammelt. Dabei haben über 4214 Betroffene aus Deutschland, Österreich und der Schweiz teilgenommen und somit 68511 Migräneanfälle mitgeteilt. Aufgeteilt bestehen die Daten insgesamt aus 31388 klassischen und 37123 gewöhnlichen Migräneattacken.

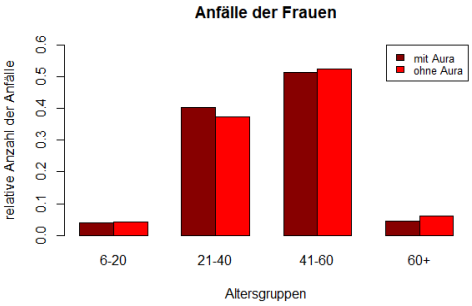


Abb. 1: Verteilung der Anfälle von Frauen in den verschiedenen Altersgruppen

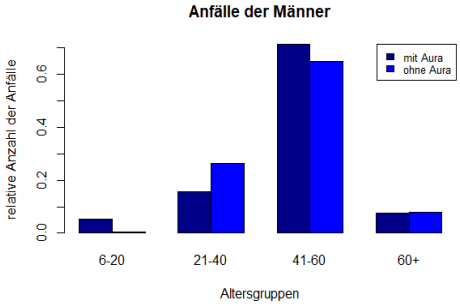


Abb. 2: Verteilung der Anfälle von Männern in den verschiedenen Altersgruppen

2.4 Datenvorverarbeitung

Um unzuverlässige Daten zu entfernen, wurden folgende Datensätze aussortiert. Zuerst sind alle Migräneanfälle von Personen, die ihr Alter auf unter sechs Jahre angegeben haben, ausgeschlossen worden. Diese Altersgruppe stellt keine hohe Relevanz in Bezug auf diese Studie dar, da der Anteil dieser Gruppe sehr gering ist und somit eine Verfälschung der Studie ausgeschlossen werden kann. Zusätzlich wurden alle Anfälle von Teilnehmern aussortiert, welche weniger als drei Tage am ganzen Projekt teilgenommen und weniger als zwei Anfälle gemeldet haben. Dadurch lässt sich gewährleisten, dass alle Anfälle von tatsächlichen Migränepatienten, die aktiv an dem Migräneprojekt teilnehmen, stammen.

Um Unterschiede im Alter bezüglich MA und MO deutlicher feststellen zu können, werden die Daten in vier verschiedene Altersgruppen eingeteilt. Diese wurden so gewählt, dass zwischen den Gruppen jeweils die markanten Hormonveränderungen auftreten und somit eine bessere Gegenüberstellung der Migräneanfälle und der hormonellen

Veränderungen ermöglicht wird. Die erste Gruppe besteht aus allen Teilnehmern im Alter von 6 bis 20 Jahren, die zweite aus Beteiligten zwischen 21 und 40 Jahren, die dritte aus Betroffenen zwischen 41 und 60 Jahren und die vierte betrachtet alle Patienten über 60 Jahren.

2.5 Statistik

Zur Datenanalyse wurde mit der Programmiersprache R in der Entwicklungsumgebung R Studio gearbeitet. Dabei wurden Chi–Quadrat–Tests durchgeführt, um die Gleichverteilung der Anfälle der Patienten mit und ohne Aura vergleichen zu können. Ebenfalls wurden U–Tests verwendet, um die Anfallshäufigkeit pro Monat der unterschiedlichen Altersgruppen gegenüber zu stellen.

3 Ergebnisse

3.1 Chi–Quadrat–Test

Zunächst wurde der Chi–Quadrat–Test mit einem Signifikanzniveau von 5% verwendet, um die Anfälle von MO und MA auf Gleichverteilung zu testen. Hierbei betrachtete man die Altersstufen einmal geschlechterunabhängig und einmal differenziert in weiblich und männlich. Dazu wurden alle Anfälle für jeden einzelnen Tag in MA– und MO–Anfälle geteilt und in eine Vierfeldertafel geschrieben. Zum Schluss wurden alle entstandenen Vierfeldertafeln aufsummiert und als Daten für die X^2–Tests verwendet.

	Teilnehmer MA	**Teilnehmer MO**	**Summe**
Anfall	Teilnehmer MA, die einen Anfall hatten	Teilnehmer MO, die einen Anfall hatten	Teilnehmer, die einen Anfall hatten
kein Anfall	Teilnehmer MA, die keinen Anfall hatten	Teilnehmer MO, die keinen Anfall hatten	Teilnehmer, die keinen Anfall hatten
Summe	Alle Teilnehmer MA	Alle Teilnehmer MO	

Tab. 1: Vierfeldertafel der Anfälle von MO und MA

Auf den Ergebnissen dieser Tests basierend wurde jeweils das zugehörige Odds Ratio (OR) für den Vergleich der Anzahl der MA–Anfällen mit der Anzahl der MO–Anfälle ermittelt. Hierfür wurde ein Konfidenzintervall von 95% gewählt. Auf Abb. 3 und Abb. 4 sind die Resultate als Grafik bildlich dargestellt. Hierbei steht ein OR > 1 dafür, dass die MA–Anfälle überwiegen. Ein OR < 1 hingegen bedeutet, dass MA–Anfälle weniger häufig vorkommen.

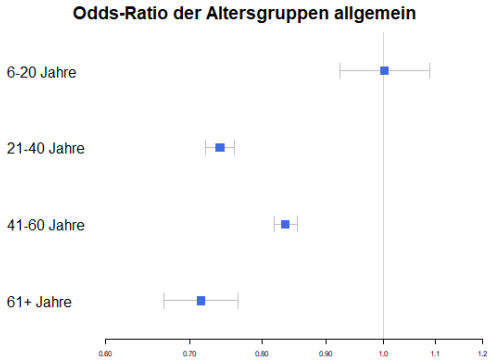


Abb. 3: OR des Vergleichs der Anfallshäufigkeit von MA mit MO

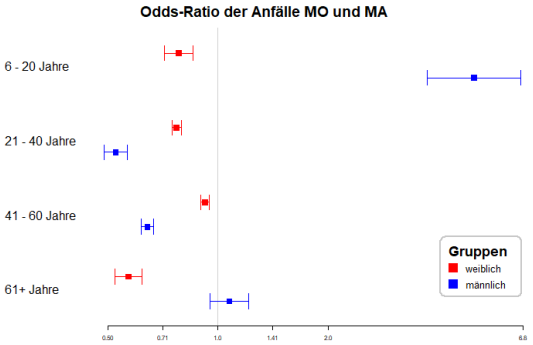


Abb. 4: OR des Vergleichs der Anfallshäufigkeit, Geschlechter getrennt

Nach der Vorverarbeitung der Daten bleiben noch 2904 Teilnehmer mit 66886 Migräneattacken, davon 1522 MA–Patienten mit 30540 Attacken und 1382 MO–Patienten mit 36346 Attacken (Verh. MA : OA 1,1 : 1), für die statistische Untersuchung bestehen. Dadurch ergeben sich folgende Teilnehmeranzahlen und Anfallszahlen pro Altersgruppe:

- 6–20 Jahre: 193 Personen und 2478 Attacken (1:13)
- 21–40 Jahre: 1408 Personen und 23233 Attacken (1:17)
- 41–60 Jahre: 1205 Personen und 37225 Attacken (1:31)
- über 60 Jahre: 98 Personen und 3950 Attacken (1:40)

Die Anzahl von Patienten mit klassischer Aura und gewöhnlicher Aura unterscheidet sich hierbei nicht deutlich.

Betrachtet man beide biologischen Geschlechter gemeinsam (Abb. 3), erkennt man, dass im Alter von 6–20 Jahren die Anfälle von MO und MA gleichverteilt sind (p = 0,969), in den älteren Altersgruppen wiederum ist ein deutlich signifikanter Unterschied zu erkennen (p < 0,001). In der Gruppe 21–40 fällt das Odds Ratio etwas ab ($OR_{21-40} = 0{,}740$) und die Anfallshäufigkeit von MO überwiegt. Bei den Teilnehmern zwischen 41 und 60 Jahren nimmt das Odds Ratio wieder ein wenig zu ($OR_{41-60} = 0{,}835$). Bei über 60–Jährigen sinkt das Odds Ratio noch einmal ab ($OR_{6-20} = 0{,}715$).

Bei den weiblichen Teilnehmern sind deutlich signifikante Unterschied erkennbar (Abb. 4). Diese treten zwischen der Anfallshäufigkeiten der Attacken mit und ohne Aura über den kompletten Studienzeitraum hinweg auf (p < 0,001 für alle Altersgruppen). Dabei ist zu erkennen, dass die Anfallshäufigkeit bei MO permanent überwiegt, wobei sich leichte Schwankungen zwischen den Gruppen ermitteln lassen. Im Alter von 6–40 Jahren ähnelt sich das Odds–Ratio sehr ($OR_{6-20} = 0{,}782$ und $OR_{21-40} = 0{,}771$). In der Altersgruppe 41– 60 wiederum verschiebt sich das Odds Ratio etwas auf die MA–Seite ($OR_{41-20} = 0{,}922$). Im Alter 61+ fällt das Odds Ratio ($OR_{60+} = 0{,}569$) wieder ab und verlagert sich ein wenig mehr auf die MO–Seite.

Bei den männlichen Patienten wird ein signifikanter Unterschied in den Altersgruppen 6–20, 21–40 und 41–60 Jahren ersichtlich (p < 0,001), in der Altersgruppe 61+ wiederum sind die Anfälle mit und ohne Aura gleichverteilt (p = 0,251), siehe Abb. 4. Vor allem auffällig ist, dass gerade im Alter zwischen 6 und 20 Jahren Anfälle von MA ca. fünfmal so oft vorkommen wie MO–Anfälle ($OR_{6-20} = 5{,}005$). In der Altersgruppe 21–40 Jahren sinkt das Odds Ratio ($OR_{21-40} = 0{,}525$), sodass nun die MO–Anfälle sichtlich häufiger vorkommen. In den zwei höchsten Altersgruppen hingegen steigt das Odds Ratio an ($OR_{41-60} = 0{,}641$), bis er schließlich im Alter von 60 + eine Gleichverteilung zwischen MO– und MA–Anfällen erreicht ($OR_{60+} = 1{,}073$).

Bei den männlichen Patienten wird ein signifikanter Unterschied in den Altersgruppen 6–20, 21–40 und 41–60 Jahren ersichtlich (p < 0,001), in der Altersgruppe 61+ wiederum sind die Anfälle mit und ohne Aura gleichverteilt (p = 0,251), siehe Abb. 4. Vor allem auffällig ist, dass gerade im Alter zwischen 6 und 20 Jahren Anfälle von MA ca. fünfmal so oft vorkommen wie MO–Anfälle ($OR_{6-20} = 5{,}005$). In der Altersgruppe 21–40 Jahren sinkt das Odds Ratio ($OR_{21-40} = 0{,}525$), sodass nun die MO–Anfälle sichtlich häufiger vorkommen. In den zwei höchsten Altersgruppen hingegen steigt das Odds Ratio an ($OR_{41-60} = 0{,}641$), bis er schließlich im Alter von 60 + eine Gleichverteilung zwischen MO– und MA–Anfällen erreicht ($OR_{60+} = 1{,}073$).

3.2 U–Test

Um die zentralen Tendenzen der Verteilungen der durchschnittlichen Anfälle pro Monat in den einzelnen Altersgruppen der beiden biologischen Geschlechter (aufgeteilt in MO und MA) vergleichen zu können, wurden diese zunächst mithilfe des Shapiro-Wilk- und des Jarque-Bera-Tests auf eine Normalverteilung geprüft (α = 0,05). Diese Tests ergaben,

dass lediglich ein sehr geringer Teil von diesen normalverteilt ist, deshalb wurden folglich nur Mann-Whitney-U-Tests ($\alpha = 0{,}05$) zur Überprüfung der Gleichheit der Lageparameter verwendet. Zwischen den Altersstufen wurde der U–Test zum einen durchgeführt, um Änderungen zwischen diesen ermitteln zu können. Die dabei entstandenen Ergebnisse von MA und MO in derselben Gruppierung wurden anschließend verglichen. Zum anderen wurden innerhalb der Altersstufen die durchschnittliche Anfallshäufigkeit von MO und MA pro Monat gegenübergestellt.

Altersgruppe 6 – 20		
	$MA_{6\text{-}20}$ vs. $MA_{21\text{-}40}$	kein signifikanter Unterschied (p=0,17)
	$MO_{6\text{-}20}$ vs. $MO_{21\text{-}40}$	signifikanter Unterschied (p=0,025; $\mu_1 > \mu_2$)
	$MO_{6\text{-}20}$ vs. $MA_{6\text{-}20}$	kein signifikanter Unterschied (p=0,257)
Altersgruppe 21 – 40		
	$MA_{21\text{-}40}$ vs. $MA_{41\text{-}60}$	signifikanter Unterschied (p=0,029; $\mu_1 < \mu_2$)
	$MO_{21\text{-}40}$ vs. $MO_{41\text{-}60}$	kein signifikanter Unterschied (p=0,178)
	$MO_{21\text{-}40}$ vs. $MA_{21\text{-}40}$	kein signifikanter Unterschied (p=0,086)
Altersgruppe 41 – 60		
	$MA_{41\text{-}60}$ vs. MA_{61+}	kein signifikanter Unterschied (p=0,753)
	$MO_{41\text{-}60}$ vs. MO_{61+}	signifikanter Unterschied (p=0,002; $\mu_1 < \mu_2$)
	$MO_{41\text{-}60}$ vs. $MA_{41\text{-}60}$	kein signifikanter Unterschied (p=0,584)
Altersgruppe 61+		
	MO_{61+} vs. MA_{61+}	signifikanter Unterschied (p=0,021; $\mu_1 > \mu_2$)

Tab. 2: Ergebnisse U-Test Frauen

Altersgruppe 6 – 20		
	$MA_{6\text{-}20}$ vs. $MA_{21\text{-}40}$	signifikanter Unterschied (p=0,007; $\mu_1 > \mu_2$)
	$MO_{6\text{-}20}$ vs. $MO_{21\text{-}40}$	kein signifikanter Unterschied (p=0,522)
	$MO_{6\text{-}20}$ vs. $MA_{6\text{-}20}$	signifikanter Unterschied (p=0,016; $\mu_1 < \mu_2$)
Altersgruppe 21 – 40		
	$MA_{21\text{-}40}$ vs. $MA_{41\text{-}60}$	kein signifikanter Unterschied (p=0,348)
	$MO_{21\text{-}40}$ vs. $MO_{41\text{-}60}$	signifikanter Unterschied (p=0,043; $\mu_1 < \mu_2$)
	$MO_{21\text{-}40}$ vs. $MA_{21\text{-}40}$	kein signifikanter Unterschied (p=0,367)
Altersgruppe 41 – 60		
	$MA_{41\text{-}60}$ vs. MA_{61+}	kein signifikanter Unterschied (p=0,207)
	$MO_{41\text{-}60}$ vs. MO_{61+}	kein signifikanter Unterschied (p=0,324)
	$MO_{41\text{-}60}$ vs. $MA_{41\text{-}60}$	signifikanter Unterschied (p=0,012; $\mu_1 > \mu_2$)
Altersgruppe 61+		
	MO_{61+} vs. MA_{61+}	kein signifikanter Unterschied (p=0,56)

Tab. 3: Ergebnisse U-Test Männer

Vergleicht man die zentralen Tendenzen der durchschnittlichen monatlichen Anfälle von Frauen zwischen den verschiedenen Altersgruppen, fällt auf, dass sich MA und MO immer unterschiedlich verhalten (Tab. 2). In der Gegenüberstellung von 6–21 und 21–40 gibt es einen signifikanten Unterschied zwischen der durchschnittlichen MO–Anfallshäufigkeit pro Monat, aber nicht bei MA. Bei 21–40 vs. 41–60 existiert ein signifikanter Unterschied bei MA, jedoch bei MO nicht. Hingegen lässt sich bei 41–60 gegenüber 61+ nur ein signifikanter Unterschied bei MO feststellen. Außerdem gibt es eine nachweisliche Differenz zwischen MA und MO in der Altersgruppe 61.

Nun wird die männliche Gruppierung der durchschnittlichen Anfälle im Monat näher betrachtet (Tab. 3). In dieser Einteilung ist ein signifikanter Unterschied zwischen 6–20 und 21–40 bei MA erkennbar, aber nicht bei MO. Im Vergleich zwischen 21–40 und 41–60 ist es invertiert: MO ist signifikant, MA jedoch nicht. Die Vergleiche von MA und MO innerhalb der einzelnen Altersgruppen zeigen einen signifikanten Unterschied bei den Gruppen 6–20 sowie 40–60.

4 Diskussion

Im Rahmen dieser Arbeit wurden 66886 Migräneattacken von 2904 Patienten untersucht. Zunächst fällt auf, dass zwar mehr aktive Teilnehmer mit Aura an der Studie teilnehmen (1522 Patienten, 30540 Attacken), wiederum aber mehr Attacken von MO-Patienten gemeldet wurden (1382 Patienten, 36346 Attacken). Dies deutet darauf hin, dass MO–Patienten öfter Anfälle haben als MA–Patienten, was eine Gleichverteilung der Anfälle von MO und MA bereits eher als unwahrscheinlich erscheinen lässt. Um das näher zu

betrachten, wurden die Anfälle bei MO und MA mithilfe von Chi–Quadrat–Tests auf Gleichverteilung getestet. Anschließend wurde das Odds Ratio bestimmt, um Trends ausfindig zu machen. Da das Odds Ratio der Altersgruppen allgemein (Abb. 3) keine große Aussage darüber treffen lässt, bei welchem Geschlecht welche Krankheit in den einzelnen Altersgruppen überwiegt, wird im Folgenden nur das Odds Ratio, welches die Geschlechter getrennt behandelt (Abb. 4), betrachtet.

Nach Ermittlung der Werte fällt vor allem das Odds Ratio der Männer im Alter von 6–20 auf, welcher als einziger stark in die MA–Richtung tendiert. MA–Anfälle sind hier ca. fünfmal häufiger als MO–Anfälle. Beobachtet man den Verlauf des Odds Ratio in den weiteren Altersstufen, sieht man eine starke Schwankung in Richtung der Altersstufe 21–40, nun kommen MO–Anfälle nahezu doppelt so häufig vor wie MA–Anfälle (OR = 0,525). In den Altersgruppen 40–60 flacht der große Unterschied zwischen den zwei Migränearten wieder etwas ab, sodass MO nur noch leicht überwiegt. Bis sich in der Altersgruppe 61+ sogar eine Gleichverteilung zwischen den Anfällen bei MO und MA einpendelt.

Betrachtet man den Verlauf des Odds Ratio der männlichen Teilnehmer (Abb. 4), erinnert das Verhalten von MO an den Testosteronspiegel eines Mannes (Abb. 5). Dieser steigt erst ab der Pubertät an, in welcher ebenso MO–Anfälle ein auffällig geringeres Vorkommen an den Tag legen als MA–Anfälle. Zwischen dem 21. und 40. Lebensjahr steigt er weiter, hier überwiegen ebenfalls die MO–Anfälle und nehmen deutlich zu. Um die 40 Jahre erreicht der Hormonspiegel seinen Höhepunkt bis er schließlich wieder schleichend etwas zurück geht, auch die Differenzen zwischen MA und MO nehmen ab. Zudem erreicht die Testosteronkonzentration im Alter nie einen so niedrigen Wert wie in der Kindheit. Auch die MO–Anfälle scheinen ab 60+ wieder abzunehmen, jedoch überwiegen die MA–Attacken nicht, wie es bei 6–20 der Fall ist.

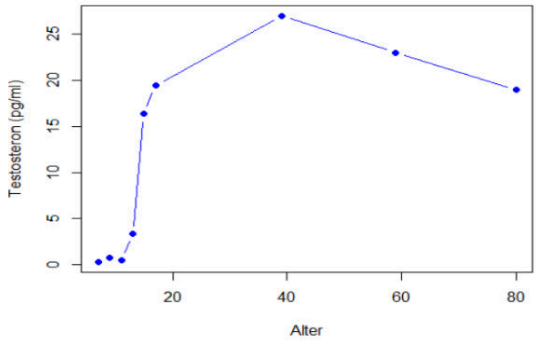


Abb. 5: Grober Verlauf des Testosteronspiegels von Männern [Te19]

Um genauere Informationen über die Unterschiede zwischen den Altersgruppen und zwischen MO und MA zu erhalten, wurden die durchschnittlichen Anfälle im Monat der Teilnehmer mit U–Tests verglichen (Tab. 3). Dabei wurde bei MA–Anfällen ein signifikanter Unterschied zwischen der Altersgruppe 6–20 und 21–40 festgestellt, die Anfälle im Monat nahmen, anders als die Testosteronwerte deutlich ab. Bei MO gab es keinen signifikanten Unterschied, was auch eher gegen den Testosteronvergleich spricht. Jedoch kommen in der Altersgruppe 21–40 MO–Anfälle deutlich häufiger vor als die MA–Anfälle. Dies lässt die Ähnlichkeit mit dem Testosteronspiegelverlauf doch nicht komplett ausschließen, vielleicht verursacht das Hormon hier nur den Ausbruch der Krankheit, denn es kommen sehr viele Patienten hinzu. Das kann jedoch auch daran liegen, dass in dieser Altersstufe mehr Menschen auf die App aufmerksam werden.
Beim Vergleich zwischen den durchschnittlichen Anfällen von MO und MA in der Altersgruppe 6–21 ist ebenfalls ein signifikanter Unterschied feststellbar. Das spricht zusätzlich dafür, dass MA–Anfälle im Alter, in welchem die Testosteronkonzentration recht gering ist, besonders häufig und somit eher von dem Steroidhormon unabhängig sind. Wobei dies mit kleineren Altersgruppen näher untersucht werden müsste.
Im Altersvergleich von 21–40 und 41–60 ist bei MO sowie bei MA ein signifikanter Unterschied in der Anfallshäufigkeit zu erkennen. Die Anfälle nehmen in der Zeit, in welcher Testosteron sehr stark ist, zu. Die MO–Anfälle verdoppeln sich, wobei sich die MA–Anfälle fast verfünffachen. Hier spricht jedoch eher der leichte Anstieg für den Testosteron–Spiegel, nicht der starke, da zu diesem Zeitpunkt Testosteron nur bedingt ansteigt. Im Vergleich zwischen MO und MA in der Altersgruppe 41–60 wiederum erkennt man, dass MO signifikant mehr Anfälle pro Monat aufweist, was aber vermutlich daran liegt, das MO generell häufiger vorkommt und zusätzlich noch mehr gestiegen ist. Im weiteren Verlauf gibt es keine signifikanten Unterschiede in der Anfallshäufigkeit im Monat. Das spricht nicht zwanghaft gegen den Testosteronspiegel, da dieser langsam absinkt. Um das genauer herauszufinden müsste die Gruppe verkleinert und nochmal näher untersucht werden.

Betrachtet man hingegen die weibliche Seite anhand von X^2–Tests wird sichtbar, dass die Anfälle von MO und MA in allen Gruppen nicht gleichverteilt sind. Des Weiteren wird durch die Odds Ratio deutlich, dass anders als bei den Männern, keine großen Schwankungen zwischen den Altersgruppen herrschen (Abb. 4). Es hält sich in allen Gruppen auf der MO–Seite. Das bedeutet, bei Frauen überwiegen durchgehend die MO–Anfälle. Das liegt möglicherweise daran, dass MO–Anfälle generell häufiger sind. Betrachtet man die Entwicklung des Odds Ratio Graphen im Alter genauer, sieht man, dass dieser in der Gruppe 2–40 relativ gleich bleibt, mit 41–60 etwas mehr in die Richtung der Gleichverteilung schwankt und im Alter von 60+ eine starke Tendenz zur MO–Seite hat. Bei der Verteilung der Anfälle von Frauen (Abb. 1) lassen sich ebenfalls keine großen Unterschiede zwischen MA und MO feststellen, bis auf minimale Schwankungen. Allerdings ergeben sich zwischen den Anfalls–Verteilungen von MA und MO leichte Parallelitäten zu der Östrogenverteilung von Frauen im unterschiedlichen Alter. Um diese Erkenntnis zu überprüfen, wurden hier ebenfalls U–Tests zum Vergleich von

durchschnittlichen Anfällen im Monat bei MA und MO durchgeführt. Vergleicht man in der Gruppe MO, die Altersgruppen 6–20 und 21–40 fällt ein signifikanter Unterschied auf, die Anfallshäufigkeit nimmt ab. In der Altersgruppe, in welchen der Östrogenspiegel der Frau ansteigt, sinken die Anfälle. Bei MA ist kein Unterschied auffindbar. Beim Vergleich der Altersgruppe 21–40 und 41–60 konnte ein signifikanter Unterschied hinsichtlich MA ermittelt werden. Die Anfälle im Monat steigen leicht an. Bei MO war dies nicht der Fall, was erneut dem Östrogenverlauf (Abb. 6) gleicht, da dieser in der Altersspanne von 21–60 zwar etwas schwanken kann, allerdings grundsätzlich stets einen ähnlichen Wert beibehält. Zwischen der Altersgruppe 41–60 und 61+ gibt es einen sehr signifikanten Unterschied der durchschnittlichen Anfälle pro Monat von MO. Hier sinkt die Anfallshäufigkeit im Monat deutlich. Bei MA gibt es keinen signifikanten Unterschied. In dieser Zeitspanne beginnt bei den meisten Frauen die Menopause, bei welcher die Östrogenproduktion rapide zurückgeht und die Konzentration des Hormons sehr niedrig ist.

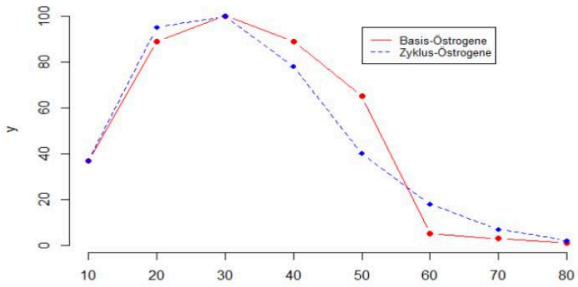


Abb. 6: Grober Verlauf des Östrogenspiegels von Frauen [Ro14]

Der rapide Anstieg von MO–Anfällen im Monat spiegelt sich auch im Vergleich von MO– und MA–Anfällen im Alter von 61+ wider. Hier besteht zum ersten Mal ein signifikanter Unterschied zwischen MO und MA in einer Altersstufe, MO–Patientinnen haben im Durchschnitt deutlich mehr Anfälle als MA–Patientinnen.

Zusammengefasst lassen sich bei Männern Parallelitäten zu dem Konzentrationsverlauf von Testosteron erkennen. Das gilt sowohl für die MA– als auch für die MO–Anfälle. Dabei sprechen aber mehr Aspekte für einen Zusammenhang zwischen MO und Testosteron, was mithilfe von medizinischen Studien näher beleuchtet werden könnte.
Die Erkenntnisse der Untersuchung von MA und MO bei Frauen ergeben mögliche Gemeinsamkeiten bzw. Zusammenhänge mit Östrogenen in Bezug auf die durchschnittliche Anfallshäufigkeit im Monat. Dieses Phänomen ist besonders bei MO zu beobachten. Bei MA ist es eher auszuschließen, da sich die durchschnittlichen Anfälle pro Monat eher gegensätzlich zu der Hormonverteilung von Östrogen entwickeln.
Grundsätzlich ist zu beachten, dass zwar die Anfälle im Monat in vorwiegend hormonbelasteten Zeiten zurückgehen, aber prozentual dennoch in dieser Gruppe die meisten Anfälle aufgezeichnet wurden. Allgemein deuten die Ergebnisse dieser

Untersuchung darauf hin, dass MO durch Hormone sowohl bei Männern als auch bei Frauen beeinflusst werden könnte.

Es existieren bereits einige Studien, welche das Thema Migräne in Verbindung mit Östrogen behandeln. Die meisten davon beziehen sich dabei lediglich auf die menstruelle Migräne. Diese ist eine Form der Migräne, welche im Bezug mit der Periode auftritt [Ga16b]. In Papern wurde ebenfalls davon berichtet, dass MO bei menstrueller Migräne häufiger vorkommt und mit Zeiten, in welchen ein Hormonwechsel vorherrscht, wie z. B. bei einem starken Rückgang von Östradiol kurz vor der Menstruation, verbunden sind. [Bo04][Ma06]
Diese Studien gehen aber nicht konkret auf den Unterschied zwischen MA und MO in diesem Bezug ein. Außerdem wird in unserem Paper nicht die Periode und somit menstruelle Migräne als Einzelnes behandelt, sondern MA und MO relativ unabhängig von der Menstruation in den verschiedenen Altersgruppen. Zusätzlich gibt es sehr alte Studien, welche sich mit der Unterscheidung von MA und MO auseinandersetzen. Diese fanden einen höheren Zusammenhang zwischen hormongeprägten Zeiten und MO–Anfällen der Frau, jedoch nicht mit MA–Anfällen. [Ra92b] Des Weiteren wurden nur die Frauen in Verbindung mit Hormonen gesetzt, die Männer wurden nicht wirklich beleuchtet, welches in unserem Paper ebenfalls durchgeführt wird.

Da in unserer Untersuchung in einigen Altersgruppen (6–20 Jahre und 61+) im Vergleich zu den übrigen Gruppen weniger Datensätze von aktiven Teilnehmern und deren Anfälle zu Verfügung standen, könnten diese Gruppen als nicht repräsentativ angesehen werden. Dies kann zum einen daran liegen, dass tatsächlich die Anfalls– und Erkrankungsrate sehr gering ist und deshalb weniger Personen teilnehmen. Des anderen kann es auch dadurch verschuldet sein, dass Menschen in höheren Altersgruppen die Medien, über welche die Anfälle gemeldet werden, sehr selten nutzen. Ebenfalls ist es möglich, dass die Altersstufen in dieser Studie zu groß gewählt wurden und kleinere Gruppen eventuell signifikantere Ergebnisse erbracht hätten. Dies kann in weiteren Studien berücksichtigt werden. Die Hormonverteilungswahl spielt bei der Interpretation der Resultate in dieser Arbeit eine sehr bedeutsame Rolle. Dies ist ein großer Kritikpunkt unserer Arbeit, weil wir keine besonders aussagekräftigen und zertifizierten Hormonverteilungen gefunden bzw. zur Verfügung gestellt bekommen haben (sowohl bei Testosteron als auch bei Östrogen). Für weitere Untersuchungen dieses Themenbereichs wäre es sinnvoll auf authentifizierte Quellen zurückzugreifen. Interessant wäre es, neben den Hormonauswirkungen, ebenfalls den Grund für das deutlich vermehrte Auftreten von MA Anfällen bei Männern im jungen Alter zu untersuchen.

Literaturverzeichnis

[Ga16a] C. Gaul, H.C. Diener. Kopfschmerzen, Pathophysiologie - Klinik - Diagnostik - Therapie. Georg Thieme Verlag, Königstein Essen 2016, S.58.

[Bl95] J.N. Blau. Migraine with aura and migraine without aura are not different entities. *Cephalalgia*. 1995; 15(3):186–90.

[Ra95] B.K. Rasmussen. Migraine with and without aura are two different entities. *Cephalalgia*. 1995 Jun;15(3):183–6.

[Ra92a] B.K. Rasmussen. Migraine with aura and migraine without aura: an epidemiological study. *Cephalalgia*. 1992; 12:221–8.

[Te19] Testosteron, In: Laborlexikon. URL: http://www.laborlexikon.de/Lexikon/Infoframe/t/Testosteron.htm (Abgerufen: 06.07.2019)

[Ro14] A. Römmler. Hormone, Leitfaden für die Anti–Aging–Sprechstunde. Georg Thieme Verlag, Stuttgart 2014, S.6.

[Ga16b] C. Gaul, H.C. Diener. Kopfschmerzen, Pathophysiologie – Klinik – Diagnostik – Therapie. Georg Thieme Verlag, Königstein Essen 2016, S.62.

[Bo04] M.G. Bousser. Estrogens, Migraine, and Stroke. *Stroke*. 2004; 35:2652–2656.

[Ma06] E.A. MacGregor, A. Frith, J. Ellis, L. Aspinall, A. Hackshaw. Incidence of migraine relative to menstrual cycle phases of rising and falling estrogen. *Neurology*. 2006; 67(12).

[Ra92b] B.K. Rasmussen. Migraine with aura and migraine without aura: an epidemiological study. *Cephalalgia*. 1992; 12:221–8.

Mensch-Maschinen-Schnittstelle mit Vibrationsaktoren zum „Fühlen“ von Texten

Jan Krämer[1]

Abstract: In diesem Paper wird ein Vibrationsgürtel zum „Fühlen“ von Texten vorgestellt. Der Fokus liegt dennoch auf dem Erkennen einzelner Buchstaben. Dieser Vibrationsgürtel stellt allgemeiner eine Mensch-Maschinen-Schnittstelle über den Tastsinn dar, mit der prinzipiell beliebige Daten übertragen werden können.

Keywords: Mensch-Maschinen-Schnittstelle; Tastsinn; Vibrationsgürtel; Text fühlen

1 Einleitung

In unserer Interaktion mit Computer spielt der Tastsinn, mit Ausnahme der Hände und deren motorischen Fähigkeiten eine eher untergeordnete Rolle. So ist nur das Anzeigen neuer Nachrichten (mittels Vibration) eine verbreitete Smartphonefunktion. Der Tastsinn könnte aber gerade durch die fast vollständige Verdeckung mittels Kleidung ein interessanter Weg der Kommunikation mit Maschinen darstellen. Des Weiteren wird die für den Alltag wichtige visuelle und auditive Wahrnehmung nicht blockiert, wie es z.B: durch Kopfhörer oder Bildschirme der Fall ist. Motivation war die Entwicklung eines Gürtels mit dem eingehende Kurznachrichten „gefühlt“ werden können.

Schon Rittmannsberger hat 1968 zu der Vibrationsempfindung geforscht [Rit68] und Bäuerle spricht 1973 bereits von dem Tastsinn als Sinnesprothese beispielsweise für Gehörlose [Bäu73]. Arbeiten in einer ähnlichen Richtung wurden von Chu und Peng unternommen. Sie haben das Erkennen von Buchstaben mit einem 2x2 Layout von Vibratoren auf der Rückseite eines Smartphones untersucht [CP18]. Darüber hinaus haben Kaspar u. a. untersucht wie sich ein Vibrationsgürtel als eine Art Kompass nutzen lässt und so einen Magnetsinn schafft [Kas+14]. Novich und Eagleman haben mit einer Vibrationsweste und dem Ziel gesprochene Sprache zu erkennen, den Datendurchsatz des Tastsinns und Wahrnehmung von Vibrationsmustern untersucht [NE15].

[1] DHBW Karlsruhe, Informatik, Erzbergerstraße 121, 76133 Karlsruhe BW mail@jan-kraemer.de

2 Methoden

2.1 Apparatur

Abb. 1: Rückseite des Vibrationsgürtels mit Elektronik

Abb. 2: Vibrationsgürtel am Körper getragen

Die Schnittstelle zum Tastsinn wurde mit einem selbst entwickelten Vibrationsgürtel geschaffen. Als Basis für diesen wurde ein Taillengürtel mit gummierter Innenseite und einem Klettverschluss gewählt. Der Vibrationsgürtel ist mit 16 Vibratoren ausgestattet und wird mit einer tragbaren Elektronik angesteuert. Der Vibrationsgürtel kann Abbildungen 1 und 2 entnommen werden. Alle 16 Vibratoren können einzeln angesteuert werden. Bei den Vibratoren handelt es sich um Linear-Resonanz-Aktoren, die auch in Smartphones verbaut werden. Diese wurden auf den Gürtel aufgenäht und auf der Rückseite mit der Elektronik verbunden (vgl. Abbildung 1). Die Vibratoren haben jeweils einen Abstand von 10cm zueinander und der Gürtel ist auf einen Taillenumfang von 90cm ausgelegt, bei diesem Umfang haben auch die äußeren Vibratoren eine Distanz von 10cm. Diese Distanz liegt somit über der, von Rittmannsberger bestimmten minimalen Distanz von 4-8cm [Rit68]. Das diese Unterscheidbarkeit auch in diesem konkreten Setup gegeben ist, konnte mit einem Versuch nochmals bestätigt werden. Die Ansteuerung erfolgt durch eine eigens entwickelte Smartphone-App über Bluetooth. Für das Training wurde eine weitere Anwendung entwickelt, welche das Training auf Basis der Ergebnisse anpassen kann und die Daten live visualisiert.

2.2 Codierung

Die Codierung von Text zu Vibrationen lässt viele Möglichkeiten zu, so können alle 16 Vibratoren zeitlich völlig frei angesteuert werden, um entsprechende Muster zu generieren. Damit ein Großteil der Schriftsprache abgebildet werden kann, wird eine Codierung auf Basis von Buchstaben gewählt. So können beliebige Wörter zu Vibrationen codiert werden. Alternativen wären z. B. Codierungen auf Basis von Wortsilben oder Phonemen der gesprochenen Sprache, diese wurden aber nicht weiter untersucht.

2.3 Zufällige Mustergenerierung

In einem ersten Versuch wurden Muster auf Basis der Hamming-Distanz einer binär Repräsentation generiert. Dazu wurden systematisch alle Muster mit zwei, drei und vier Einsen in dieser Repräsentation generiert und es wurde eine Liste mit ca. gleich vielen Mustern dieser drei Gruppen zusammengestellt, sodass jedes Muster zu jedem anderen ausgewählten Muster eine Hamming-Distanz von 2 Bit hat. Die Muster wurden alle für eine Dauer von 200ms vibriert und hatten keine zeitlichen Variation. Ein Training mit diesen Mustern, die auf 56 Zeichen abgebildet wurden, zeigte jedoch sehr schlechte Ergebnisse, sodass ohne weitere Untersuchung ein neuer Ansatz zur Mustergenerierung gewählt wurde.

2.4 Individualisierte Muster

Durch die Erkenntnis im ersten Versuch wurde eine neue Codierung erstellt und die Anzahl der Zeichen auf 34 reduziert. Als Basis wurden zunächst systematisch Muster mit 3 aktiven Vibratoren generiert und diese wurden anschließend interaktiv angepasst. Dazu wurden manuell die Muster getestet, wurden zwei sehr ähnliche Muster identifiziert, wurden diese verändert. Nach einigen Veränderungen entstanden die Muster, welche in Auflistung 3 abgebildet sind. Es wurde sowohl die Anzahl der Vibrationen pro Zeichen als auch die Dauer der jeweiligen Vibrationen variiert. Dies macht jedes individuelle Muster deutlich komplexer dadurch aber auch eindeutiger. Abbildung 4 zeigt beispielhaft eine Visualisierung für die Codierung der Zeichen „a“ und „b“. Es ist für jeden der 16 Vibratoren der Zustand über eine Dauer von 500ms visualisiert, die blauen Balken stellen den aktiven Zustand bzw. eine Vibration dar.

```
'a' -> (0000000010000000|150ms),(0110000000000000|150ms)
'b' -> (0000000100011001|200ms)
'c' -> (0010000100000001|200ms)
'd' -> (0000000101000001|200ms)
'e' -> (0000000000000011|150ms),(0000000000001100|150ms)
'f' -> (0001001000000010|200ms)
'g' -> (0000001000100010|300ms)
'h' -> (0100001000000010|200ms)
'i' -> (0000000000001100|150ms),(0000010000000000|150ms)
'j' -> (0000000000000101|250ms)
'k' -> (0000110000000000|200ms),(0000000000110000|200ms)
'l' -> (0010010000000100|200ms)
'm' -> (0000010001000100|250ms)
'n' -> (1000010000000100|200ms)
'o' -> (0000000000000011|150ms),(0000000100000001|150ms)
'p' -> (0000110000000010|200ms)
'q' -> (0000110000100000|200ms)
'r' -> (0100110000000000|200ms)
's' -> (0000110010000000|150ms)
't' -> (0000000000110001|200ms)
'u' -> (0000000000110000|150ms),(0001000000010000|150ms)
'v' -> (0000000000110100|150ms)
'w' -> (0000000001110000|300ms)
'x' -> (1000000000110000|200ms)
'y' -> (0110000100000000|300ms)
'z' -> (0101000000000010|150ms)
'ä' -> (1000000000000000|150ms),(0000000011000000|200ms)
'ö' -> (1000000000000000|150ms),(0000000000000011|200ms)
'ü' -> (1000000000000000|150ms),(0000000000110000|200ms)
'ß' -> (0000000011000001|200ms)
'.' -> (0000000010011001|350ms)
',' -> (0000000011000000|200ms),(0000000000001100|200ms)
'?' -> (0000000000011000|150ms),(0000000001100000|150ms),(0000000000011000|150ms)
'!' -> (0000000000010000|150ms),(0000000000001000|150ms),(0000000000000100|150ms)
```

Abb. 3: Vibrationscodierung aller Zeichen

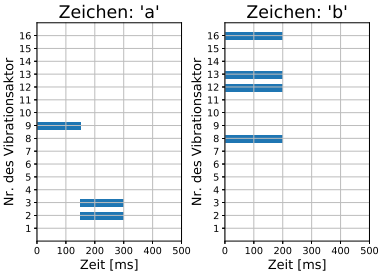


Abb. 4: Zeitliche Visualisierung der Vibrationscodierung der Zeichen „a“ und „b“

Für die Repräsentation der Muster wurde ein 16 Bit String genutzt. Jedes Bit repräsentiert den Zustand eines Vibrators für ein bestimmtes Zeitintervall. Die ersten acht Bit stehen für die obere Reihe Vibratoren auf dem Vibrationsgürtel und die letzten acht Bit für die untere Reihe.
Die tatsächliche Zusammenstellung erfolgte mit einer zufälligen Auswahl.

3 Versuche

3.1 Versuch 1: Buchstaben erkennen

In diesem Versuch wurde die entwickelte Codierung verwendet. Es wurde ein Zeichen vibriert und anschließend hatte der Proband alle 34 Zeichen zur Auswahl. Die Vibration durfte wiederholt werden und es gab keine Zeitbeschränkung für die Auswahl. Sobald die Auswahl getroffen war, wurde ein neues zufälliges Zeichen vibriert. Die hier vorgestellten Daten stammen aus einem Datensatz mit 826 vibrierten Buchstaben. Der Versuch wurde lediglich an einem männlichen Probanden, Alter 22, durchgeführt und kann somit nicht als repräsentativ gewertet werden. Zuvor fanden zwei Durchgänge mit 1000 und mit 539 vibrierten Zeichen statt. Es erfolgte also bereits ein Lernprozess.

3.2 Versuch 2: Wörter erkennen

Ein weiterer Versuch zum Erkennen ganzer Wörter, soll zeigen, ob sich das Erkennen einzelner Buchstaben auch auf Wörter übertragen lässt. Dazu wurde die gleiche Codierung wie für Buchstaben verwendet. Die Buchstaben wurden im Wort jeweils mit einer Pause von 200ms sequenziell vibriert. Dabei fand zunächst die Vibration des Wortes statt und anschließend wurde dem Probanden eine Auswahl von fünf Wörter präsentiert. Die Vibration konnte nicht wiederholt werden und es bestand keine zeitliche Beschränkung der Auswahl. Auch dieser Versuch wurde lediglich an einem Probanden durchgeführt und kann somit nicht als repräsentativ gewertet werden. Es handelte sich um den gleichen Probanden wie im ersten Versuch.

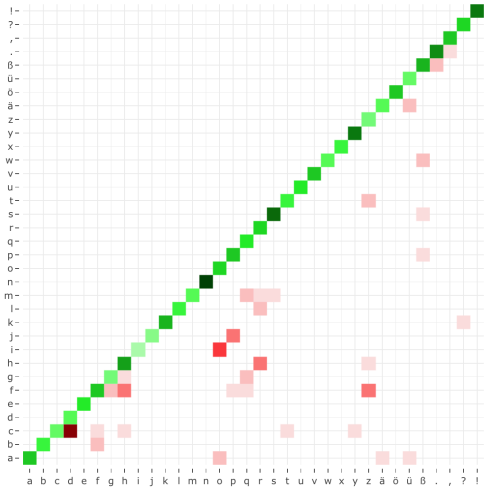
4 Ergebnisse

4.1 Buchstaben erkennen

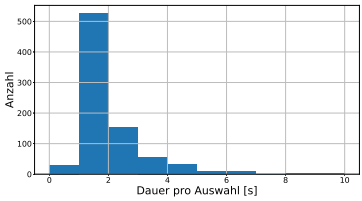
Das Ergebnis war eine Erkennungsrate von 93%. Abbildung 5a zeigt eine Erkennungsmatrix, wo die Ergebnisse *vibriertes Zeichen* und *ausgewählten Zeichen* gegeneinander aufgetragen sind. Die Farbintensität spiegelt die Häufigkeit wider. In dieser Erkennungsmatrix ist zu erkennen, dass bestimmte Buchstaben immer wieder verwechselt wurden. Teilweise können diese Verwechselungen auf Ähnlichkeiten der Muster zurückgeführt werden. Daraus lässt sich schließen, dass die individualisierten Muster noch nicht ideal sind.

Abbildungen 5b und 5c zeigen eine Analyse der Auswahl des Probanden. Diese zwei Histogramme zeigen, dass die Auswahl meist 1-2s dauerte und, dass die Vibration meist nicht wiederholt wurde, obwohl dies möglich war.

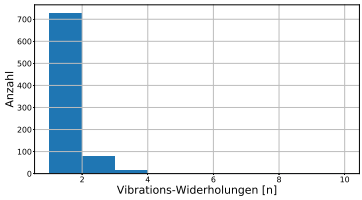
Der erste Durchgang wurde mit einer Erkennungsrate von 86% und der zweite Durchgang mit einer Erkennungsrate von 92% abgeschlossen.



(a) Erkennungsmatrix der Zeichen



(b) Histogramm der Auswahldauer



(c) Histogramm der Wiederholungen der Vibration

Abb. 5: Analyse Versuch 1

4.2 Wörter erkennen

Der Versuch zeigte sehr schnell, dass die gute Erkennung der Buchstaben sich nicht direkt auf Worte übertragen lässt. Allein aus der Vibration des Wortes war dessen Erkennung nicht möglich. Die Erkennungsrate fiel mit 80% dennoch recht gut aus, was sich durch die beschränkte Auswahl von fünf Wörtern erklären lässt. So konnte das Wort bereits durch das Erkennen ein paar weniger Buchstaben aus dieser Auswahl identifiziert werden. Aussagekräftiger als die Erkennungsrate ist eine Beschreibung des Probanden:

Während und nach der Vibration hat man keine Ahnung, um welches Wort es sich handelt. Eine Erkennung war nicht möglich, man nimmt lediglich eine Struktur war. Interessant ist aber, sobald die Auswahl der Wörter erscheint, hat man eine gewisse Intuition für das richtige Wort. Diese Intuition könnte sich aus wenigen erkannten Buchstaben, markanter Silben und der Wortlänge bilden. Die Vibrationen fühlen sich im ganzen Wort verschieden zu Vibrationen der einzelnen Buchstaben an und werden schnell abgespielt. Beim Erkennen von Worten scheint es auf Silben und Buchstabengruppen anzukommen.

5 Diskussion

Kann Text mit Vibrationen „gefühlt“ werden?

Das Erkennen von ganzen Texten konnte so im Versuch nicht bestätigt werden, allerdings konnten Buchstaben mit einer Erkennungsrate von 93% gut erkannt werden. Die Ergebnisse

des zweiten Versuchs sollten dennoch nicht zu dem Schluss führen, dass das Erkennen ganzer Worte nicht funktioniert. Dieser Versuch zeigt nur, dass sich das gute Erkennen von Buchstaben nicht direkt auf Worte übertragen lässt. Chu und Peng konnten mit dem 2x2 Layout von Vibratoren auf der Rückseite eines Smartphones eine Erkennungsrate von 84.6% erreichen. Der hier vorgestellte Vibrationsgurt mit den individualisierten Vibrationsmustern kann diese also mit 93% übertreffen, obwohl es sich um 34 Zeichen und nicht nur 26 Buchstaben handelte. Es sei aber nochmal darauf hingewiesen, dass die vorgestellten Ergebnisse nicht repräsentativ sind und für den Vibrationsgurt 16 Vibratoren verwendet wurde.

6 Ausblick

Der zweite Versuch konnte bereits Anhaltspunkte für Modifikationen des Worttrainings geben. So könnte das Lernen von Wortsilben und kleineren Buchstabengruppen helfen ganze Worte besser zu erkennen. Auch ein generell größerer Trainingsumfang könnte Fortschritte bewirken, sodass gelernte Vibrationsmuster nicht mehr explizit bewusst identifiziert werden, sondern mit entsprechendem Training schneller unterbewusst erkannt werden. Des Weiteren sind die Vibrationsmuster, eine Verbesserung des Vibrationsgürtels (Erhöhung des Tragekomforts und Nutzung anderer Körperregionen) und andere Daten als Text spannende Themen zum Weiterarbeiten.

Literatur

[Bäu73] R. Bäuerle. „Informationsübertragung durch elementare taktile Signale“. In: *Kybernetik* 13.3 (Sep. 1973), S. 172–181. ISSN: 03401200. DOI: 10.1007/BF00270512.

[CP18] Shaowei Chu und Mei Peng. „Designing a 2 × 2 spatial vibrotactile interface for tactile letter reading on a smartphone“. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Bd. 10907 LNCS. Springer Verlag, 2018, S. 327–336. ISBN: 9783319920511. DOI: 10.1007/978-3-319-92049-8_24.

[Kas+14] Kai Kaspar u. a. „The experience of new sensorimotor contingencies by sensory augmentation“. In: *Consciousness and Cognition* 28.1 (2014), S. 47–63. ISSN: 10902376. DOI: 10.1016/j.concog.2014.06.006.

[NE15] Scott D. Novich und David M. Eagleman. „Using space and time to encode vibrotactile information: toward an estimate of the skin’s achievable throughput“. In: *Experimental Brain Research* 233.10 (Okt. 2015), S. 2777–2788. ISSN: 14321106. DOI: 10.1007/s00221-015-4346-1.

[Rit68] Norbert Rittmannsberger. „Über die Vibrationsempfindung bei örtlich verteilter Reizung der Haut“. In: *Kybernetik* 4.5 (Mai 1968), S. 190–194. ISSN: 03401200. DOI: 10.1007/BF00289041.

Der Einsatz maschinellen Lernens zur inertialen Bewegungsanalyse am Pferderumpf für das digitale Trainingsmonitoring im Leistungssport

Johannes Pavel[1], Christina Fercher[2] und Frank Herold[3]

Abstract: Das leistungssportliche Training von Pferd und Reiter ist geprägt durch die langjährige Erfahrung und die individuelle Einschätzung von Reiter und Trainer. Ergänzend dazu soll langfristig eine objektive Möglichkeit zur Beurteilung von Bewegung geschaffen werden. Auf Grund der notwendigen Feldbedingungen der Sportpraxis eignet sich die Verwendung von Inertialsensoren am Pferderumpf [PF05], [NA09], [MA14], [FE17], [WA19]. Ziel dieser Arbeit ist es durch den Einsatz von *Maschinellen Lernen* über neuronalen Netzen die pferdesportpraktischen Bewegungen anhand der inertialen Bewegung des Pferderumpfs zu detektieren. Als Basis dienen die zyklischen Gangarten Schritt, Trab, Linksgalopp und Rechtsgalopp, sowie die Bewegungsrichtung, das Halten und in einem nächsten Schritt die azyklische Springbewegung [DE97], [DE12] und dressurspezifischen Lektionen, um einerseits das tägliche Training im Heimatstall dokumentieren und andererseits eine kinematische Bewegungsanalyse, bspw. am Sprung, durchführen zu können. Dabei beweist die Analyse, dass die verschiedenen Gangarten sehr gut klassifiziert werden können.

Keywords: maschinelles Lernen, neuronales Netz, Long short Term Memory Network, Convolutional Network, Leistungssport, Pferdesport, inertiale Bewegungsanalyse

1 Einleitung

Die Zielsetzung aller beteiligten Personen ist es, talentierte Reiter und Pferde zu fördern, langfristig aufzubauen und über viele Jahre gesund im Spitzensport zu halten. Dabei sind der Ausbau einer systematischen Trainingsstrategie und die Periodisierung der Saisonhöhepunkte mit ausreichenden Regenerationsphasen wichtige Aspekte für die Leistungsfähigkeit sowie Gesunderhaltung der Athleten. Im Pferdesport spielt vor allem das subjektive Empfinden von Reiter und Trainer eine wichtige Rolle. Dabei werden eine allgemeine Saison- und Trainingsplaung, Stärken-Schwächenprofile und Trainingsziele noch nicht fokussiert genug umgesetzt. Bei vielen Pferden ist das Dokumentieren der täglichen Arbeit und Bewegung darüber hinaus mühsam und zeitaufwendig. Eine Trainingsdatenbank existiert bereits, allerdings ist den Athleten der langfristige Mehrwert

[1] Deutsche Reiterliche Vereinigung e.V. (FN), Freiherr-von-Langen-Straße 13, 48231 Warendorf | Universität Kassel, Fachgebiet Verteilte Systeme, Wilhelmshöher Allee 71-73, 34121 Kassel, johannes@uni-kassel.de
[2] Deutsche Reiterliche Vereinigung | Olympiastützpunkt Westfalen, Dr.-Rau-Allee 32, 48231 Warendorf, c.fercher@osp-waf.de
[3] Peiker Cee Consumer Electronics Evolution GmbH, Gartenstraße 25, 61352 Bad Homburg, frank.herold@peiker-cee.de
Betreuer: Harun Baraki, Universität Kassel, Fachgebiet Verteilte Systeme

dieser Arbeit noch nicht ausreichend klar. Aus diesem Grund soll eine einfache und zuverlässige Methode entwickelt werden, welche das Dokumentieren von Trainingsinhalten vereinfacht.

2 Methodik

Um im Grundstein eine große Bewegungsvielfalt in den Gangarten abbilden zu können, werden sowohl Daten von Spitzenpferden als auch von Freizeitpferden an vier verschiedenen Tagen erfasst. Insgesamt liegen inertiale Bewegungsdaten von 32 unterschiedlichen Pferden (Größe: 130 – 180 cm; Alter: 4 – 23 Jahre) zur Auswertung vor, wobei ein Teil der Pferde an mehreren Messtagen mit unterschiedlichen Bewegungsaufgaben berücksichtigt werden. Das Exterieur, der Ausbildungsstand und auch der disziplinspezifische Verwendungszeck der Pferde ist sehr vielfältig. Den Messungen liegen vier verschiedene Reitbahnen zu Grunde: drei unterschiedliche Reithallen mit den Maßen 20 x 60 m, 70 x 30 m und 20 x 40 m und eine Außenplatz mit den Maßen 65 x 25 m.
Die Anbringung des inertialen Bewegungssensors (*MTw Awinda Motion Tracker XSens*, Niederlande) erfolgt am kaudalen Teil des Brustbeins des Pferdes durch eine mittige Fixierung am Sattelgurt. Mit Hilfe der Software (MT Manager 2012.1.1 beta) wird zu Beginn für jedes Pferd in einer ruhendenden Position mit einer Ausrichtung entlang einer der langen Seite der Reitbahn ein Reset der Orientierung (Methode: „*Alignment Reset*“) durchgeführt, um das ursprüngliche sensororientierte Koordinatensystem (S), unabhängig der abdominalen Linie oder der Sattelgurtform, per Rotationsmatrizen auf das Objekt Pferd (O) abzustimmen und zusätzlich die Richtung des Weltkoordinatensystems (WKS) auf die Reitbahn (RKS) auszurichten. Damit wird die Ausgangslage und die Vergleichbarkeit der Daten auch für spätere Bewegungsanalysen hergestellt. Abbildung 1 (modifiziert nach FN 2012 & XSens 2016) veranschaulicht die Koordinatensysteme.

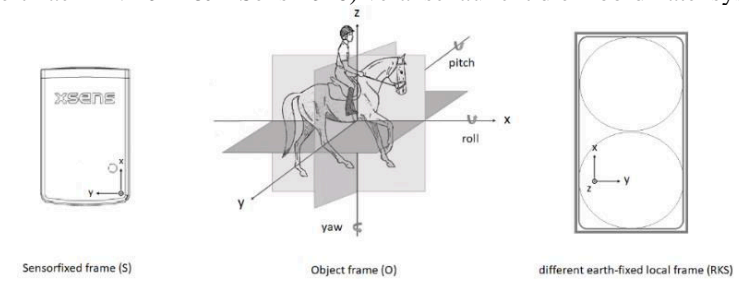


Abb. 1: Ausrichtung der Koordinatensysteme für weitere Bewegungsanalysen

Das Modell des neuronalen Netzes wurde mithilfe von Googles *Framework TensorFlow* erstellt. Dieses Framework stellt diverse Hilfsmittel für den vorgesehenen maschinellen Lernprozess bereit [AB15]. Für diese Arbeit wird die Schnittstelle *Keras* verwendet, um ein komplexes Modell des neuronalen Netzes zu entwickeln. Die Wahl für die vorgesehene Gangartendetektion fällt auf ein Modell, in dem *rekurrente* und

convolutionale, also faltbare, Schichten kombiniert werden. Der convolutionale Teil bereitet dabei die Daten auf und entdeckt neue Merkmale, während der rekurrente Teil für die Klassifizierung einzelner Sequenzen zuständig ist [BR17]. Das Modell des neuronalen Netzes besteht aus insgesamt acht Schichten verschiedener Neuronen. Die erste bildet eine eindimensionale Convolution-Schicht. Durch Reduktion auf eine Dimension ist es möglich die Eigenschaften der ursprünglichen Bilderkennung auf die Klassifizierung von Sensordaten zu übertragen [MI18]. Die zweite und sechste Neuronen Schicht besteht aus Dropout-Neuronen. Diese Neuronenart soll gewährleisten, dass das Neuronale Netz nicht nur für die bereits bekannten Pferde zuverlässige Klassifikationen liefert und setzt aus diesem Grund einen festen Prozentsatz an weitergegebenen Informationen gleich null. Die zweite Neuronen Schicht setzt 50 % der Informationen auf null, die sechste Schicht 40 %. Die dritte Schicht reduziert die Komplexität der vorhergehenden Convolution-Schicht, indem nur Maximalwerte der erlernten Eigenschaften durch eine eindimensionale Max-Pooling-Schicht weiterverwendet werden [CH15]. Die bisherigen Schichten geben jeweils eine zweidimensionale Matrix an die nachfolgenden Schichten des rekurrenten Teils weiter. Für die anschließende Klassifizierung wird allerdings ein eindimensionaler Vektor benötigt. Dieser Vektor wird in der vierten Schicht mithilfe einer *Flatten*-Schicht erzeugt. Die Schichten eins, drei und vier werden von einer *TimeDistributed*-Schicht umschlossen. Diese Schicht ermöglicht es die verwendeten Daten in mehrere Ebenen einzuteilen. Für dieses Neuronale Netz werden jeweils 50 Datensätze in einer Sequenz zusammengefasst und anschließend noch einmal in fünf gleich große Ebenen unterteilt [CH15]. Der Kern des Modells liegt in der rekurrenten Schicht aus *Long Short-Term-Memory-Neuronen* (LSTM). Dabei handelt es sich um die fünfte Schicht des neuronalen Netzes. Diese Schicht zeichnet sich besonders dadurch aus, dass sie in der Lage ist Informationen über lange Zeitabschnitte hinweg zu speichern [HO97]. Die letzten beiden Schichten bestehen aus zwei vollständig verbundenen *Dense*-Schichten. Die erste Dense-Schicht verringert die Komplexität der Ausgabe der vorhergehenden LSTM-Schicht. In der zweiten Dense-Schicht repräsentiert je ein Neuron die Wahrscheinlichkeit der klassifizierten Klasse [CH15]. Für die Aufgabe der Gangartenerkennung gibt es entsprechend der Klassen Halt, Schritt, Trab, Rechtsgalopp und Linksgalopp fünf Neuronen. Für das detektieren der Bewegungsrichtung werden nur drei Klassen benötigt, namentlich Halt, Rechts herum und Links herum. Das Modell des neuronalen Netzes wird zwei Mal unabhängig voneinander mit den unterschiedlichen Aufgaben trainiert und getestet. Dabei entscheidet die unterschiedliche Einteilung der Trainings- und Testdaten über angestrebte Klassifizierungsaufgabe. Für die Gangartenanalyse werden die Daten in die Klassen Halt, Schritt, Trab, Galopp Rechts herum und Galopp links herum eingeteilt, für die Bestimmung der Richtung sind es die Klassen Halt, rechts herum und links herum. Die Trainingsdauer variiert je nach Aufgabe, die Gangartenerkennung benötigt 12 Epochen, die Richtungsklassifizierung benötigt nur die Hälfte der Zeit.

Den Reitern wird lediglich die Gangart und Richtung vorgegeben. Wege, Tempo oder Wendungsgrad (Lektionen) in den entsprechenden Gangarten wurden frei gewählt, um näher am alltäglichen Trainingsablauf zu sein. Die Reiter-Pferd-Paare sind sich vertraut und den einfach gehaltenen Bewegungsaufgaben gewachsen.

Für das neuronale Netz werden die Beschleunigungs-, Drehraten und Magnetfelddaten, sowie die berechneten freien Beschleunigungs- und Lagedaten (Quaternionen) aus den Messungen in 100 Hz berücksichtigt. Ein Drittel der Reiter-Pferd-Paare werden zufällig ausgewählt, von den anderen separiert und stellen fortan den Testdatensatz bereit. Trainings- und Testdatensatz sind dabei disjunkt. Die besondere Architektur des neuronalen Netztes erfordert eine Einteilung der gesammelten Daten in einzelne Sequenzen zu je 50 Datensätzen aufgeteilt, was 0,5 s der Echtzeit entspricht. Pro Datensatz werden 16 Merkmale für das neuronale Netz verwendet (800 Elemente). Durch dieses Verfahren stehen 37.050 Sequenzen für das Training und 10.219 Sequenzen zum Testen des neuronalen Netzes zur Verfügung.

3 Ergebnisse

Für möglichst praxisnahe Ergebnisse wird das neuronale Netz nur über, dem Netz unbekannte, Pferde getestet. Abbildung 2 und Abbildung 3 zeigen die Ergebnisse des Maschinellen Lernvorgangs in Bezug auf die Gangartenerkennung und die Richtungsklassifizierung. Dabei befindet sich auf der Y-Achse die tatsächliche Klasse und auf der X-Achse die vorhergesagte Klasse. Die Zahlen beziffern die Anzahl der klassifizierten Sequenzen Blau sind besonders wenig oder keine Sequenzen in der entsprechenden Klasse, Rot entspricht besonders vielen Sequenzen in den Klassen. Demnach wäre das ideale Ergebnis ein diagonaler, roter Strich von oben links nach unten rechts. Die Fehlerrate steigt mit der Komplexität der gerittenen Aufgabe an. 1200 Sequenzen Halt werden von dem neuronalen Netz korrekt als Halten klassifiziert. 2393 Sequenzen Schritt werden korrekt erkannt, während vier Sequenzen Schritt für Trab, sechs für Rechts- und eine für Linksgalopp erkannt werden. Insgesamt 2330 Sequenzen Trab werden korrekt klassifiziert, während 28 fälschlicherweise als Rechtsgalopp und vier als Linksgalopp detektiert werden. Insgesamt 2126 Sequenzen gehören der Klasse des Rechtsgalopps an, davon werden 15 für Trab und 25 für Linksgalopp gehalten. 2117 Sequenzen Linksgalopp werden korrekt klassifiziert, während vier Mal die Merkmale des Rechtsgalopps und sechs Mal des Trabs überwiegen. Die Genauigkeit der Gangartenanalyse über die verschiedenen Pferde liegt bei über 99 % mit einer allgemeinen Fehlerrate von unter 1 %. Die Fehlerrate variiert pro Pferd und Lektion, so ist im Laufe der Evaluation das Pferd Nummer 18 mit einer ungleich höheren Fehlerrate im Linksgalopp aufgefallen. Es handelt sich hierbei um ein Springpferd, welches gerade im Linksgalopp sichtbar schief galoppiert. Die Fehlerrate bei diesem Pferd und der Aufgabe liegt bei etwa 7 %.

Das Ergebnis der Klasse Halt zeigt sich ähnlich ideal wie bei der Gangartenunterscheidung, lediglich 2 Sequenzen werden als links herum erkannt. Allerdings ist die Unterscheidung von rechts und links deutlich fehleranfälliger. Von 4501 Sequenzen auf der linken „Hand“ wurden insgesamt 595 fälschlicherweise für rechts herum gehalten und eine wurde als Halt klassifiziert. 4034 Sequenzen werden korrekt als geritten der rechten „Hand“ detektiert, während 484 Sequenzen als links herum erkannt

werden. Insgesamt liegt die Genauigkeit der Unterscheidung von rechts und links liegt bei etwa 89 %.

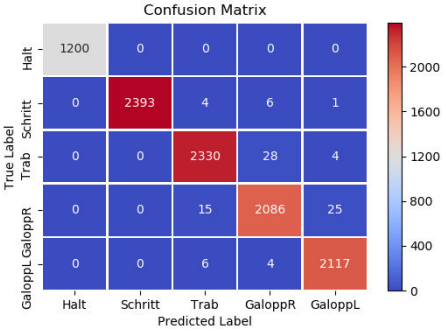


Abb. 3: Ergebnis der Gangartenerkennung

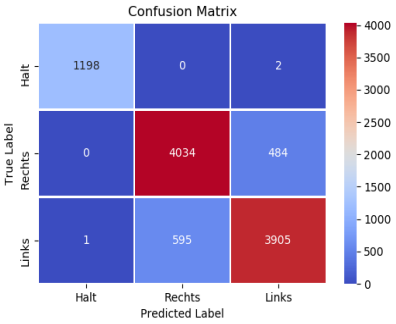


Abb.4: Ergebnis der Richtungsanalyse

4 Diskussion

Die Ergebnisse des maschinellen Lernverfahrens haben gezeigt, dass es möglich ist die Bewegungen von Pferden mit unterschiedlichen Bewegungsamplituden anhand bekannter IMU Daten sehr gut zu klassifizieren. Dennoch lassen sich zwei mögliche Fehlerquellen festhalten. Durch die Vielfalt der Pferde, deren Exterieur und Ausbildungsstand, zeigen sich auch sportpraktisch bekannte Probleme. Der Takt, als räumliches und zeitliches Gleichmaß der Bewegung, bildet den Grundstein der Ausbildung. Ungleichheiten im Rhythmus der Fußfolge, bspw. durch mangelnden Durchsprung im Galopp, Schiefe im Bewegungsablauf oder fehlender Losgelassenheit im Schritt, zeigen sich als solche auch in den Daten und bilden die Ausreißer in der Gangartenerkennung über 1 %. Mit einer wachsenden Datenbasis und weiterem Training des neuronalen Netzes werden diese weiter minimiert. Dennoch sind Störungen und „Grenzgänger“ in der Sportpraxis vorhanden und sollten bei starken Abweichungen von Normen möglicherweise objektiv als solche aufgezeigt werden.Die zuverlässige Detektion der gerittenen Reitrichtung stellt dabei eine weitaus größere Herausforderung dar. Die Zeitabschnitte, in welchen sich die Pferde auf einer geraden Linie, also entlang der langen oder kurzen Seite, bewegen, lassen sich nur sehr schwer in eine bestimmte Richtung einordnen. Hierzu wird ein weiteres Messprojekt durchgeführt, in welchem die Kategorie gerade ergänzt wird. Damit wird nicht nur die Fehlerrate minimiert, sondern zusätzlich in Summe die gerittenen Handwechsel erfasst. Da die Aufteilung einer Trainingseinheit in linke und rechte „Hand“ üblich ist, sollte für eine Auswertung die Zuordnung von gerade ohne Handwechsel dennoch nach links und rechts erfolgen.

5 Literatur

[AB15] Abadi, M. et al. (2015). Tensorflow. Zugegriffen am 15.03.2020 unter: https://www.tensorflow.org/

[BR17] Brownlee, J. (2017). CNN Long Short-Term Memory Networks. Zugegriffen am 12.03.2020 unter: https://machinelearningmastery.com/cnn-long-short-term-memory-networks/

[CH15] Chollet, F. et al. (2015). Keras Documentation. Zugegriffen am 13.03.2020 unter: https://www.keras.io/

[DE12] Deutsche Reiterliche Vereinigung e.V. (Hrsg.). (2012). *Grundausbildung für Reiter und Pferd.* (Richtlinien für Reiten und Fahren, Bd. 1). (29. Auflage). Warendorf: FN Verlag.

[DE97] Deutsche Reiterliche Vereinigung e.V. (Hrsg.). (1997). *Ausbildung für Fortgeschrittene.* (Richtlinien für Reiten und Fahren, Bd. 2). (13. Auflage, Nachdruck 2010). Warendorf: FN Verlag.

[FE17] Fercher, C. (2017). Erfassung leistungsrelevanter Parameter mit Inertialsensoren zur Darstellung der Springbewegung von Sportpferden. *Pferdeheilkunde – Equine Medicine 33(4),* 377-385.

[HO97] Hochreiter, S., Schmidhuber, J. (1997). Long Short-term Memory. *Neural Computation (journal), vol. 9, issue 8, S.1735-1780*

[MA14] Martin, P., Chateau, H., Pourcelot, P., Duray, L. & Cheze, L. (2014). Comparison Between Inertial Sensors and Motion Capture System to Quantify Flexion-Extension Motion in the Back of a Horse. *Equine Vet J* 46(46), 43.

[MI18] Missinglink (Hrsg.). (2018) Convolutional Neural Network: How to Build One in Keras & PyTorch. Zugegriffen am 14.03.2020 unter: https://missinglink.ai/guides/neural-network-concepts/convolutional-neural-network-build-one-keras-pytorch/

[NA08] Nankervis, K., Hodgins, D. & Marlin, D. (2008). Comparison between a sensor (3D accelerometer) and ProReflex motion capture systems to measure stride frequency of horses on a treadmill. *Comp. Exercise Physiol. 5(3-4),* 107-109.

[PF05] Pfau T., Witte T. H., Wilson A. M. (2005) A method for deriving displacement data during cyclical movement using an inertial sensor. J. Experim. Biol. 208, 2503-2514.

[WA19] Warschun, F., Fercher, C., Schleichardt, A. & Ueberschär, O. (2019). Reliabilität von XSens MTw Awinda Motion Trackern im Springreiten. In: Arampatzis, A., Braun, S., Schmitt, K. & Wolfahrt, B. (Hrsg.): *Sport im öffentlichen Raum. 24. dvs-Hochschultag Berlin 18.-20. September 2019 Abstracts*. Schriften der Deutschen Vereinigung für Sportwissenschaft. Vol 282. Hamburg: Feldhaus: 54.

[XS16] XSens (Hrsg.). (2016). MTW Awinda User Manual. Zugeriffen am 11.09.2019 unter:
https://www.xsens.com/hubfs/Downloads/Manuals/MTw_Awinda_User_Manual.pdf

Autorenverzeichnis

A
Amann, Tina, 131
Amarkhel, Omar, 103

D
Diez, Savina, 103
Dröse, Hannes, 77

F
Fercher, Christina, 149

H
Hanke, Karlson, 37
Hellenbrand, Andreas, 63
Herold, Frank, 149

K
Kadavanich, Annabella, 89
Krämer, Jan, 143

M
Miller, Mihail, 49
Möllers, Frederike, 23

O
Oberthür, Tim, 23

P
Pavel, Johannes, 149
Petrak, Dominic, 103

S
Savinov, Andrej, 89
Schulz, Franziska, 11
Sieper, Anna Abad, 103
Strauß, Christa, 131

W
Welter, Felix J. M., 117
Wilson, Philip, 89