Benchmark für ein personalisiertes Empfehlungssystem mit zeitlicher Segmentierung basierend auf Assoziationsregeln

Johanna Buchert
Kompetenzzentrum
für Informationstechnologie
Technische Hochschule Mittelhessen
Wiesenstr. 14, 35390 Gießen
johanna.buchert@web.de

Michael Guckert
Kompetenzzentrum
für Informationstechnologie
Technische Hochschule Mittelhessen
Wiesenstr. 14, 35390 Gießen
michael.guckert@mnd.thm.de

Christian Schulze
Kompetenzzentrum
für Informationstechnologie
Technische Hochschule Mittelhessen
Wiesenstr. 14, 35390 Gießen
christian.schulze@mnd.thm.de

***Zusammenfassung*—In der heutigen Zeit ist es üblich geworden, im Online-Marketing Empfehlungssysteme einzusetzen. Im Folgenden wird ein Empfehlungsalgorithmus vorgestellt, der auf Basis von Kaufinformationen mit Hilfe von Assoziationsregeln und unter Berücksichtigung des Zeitpunkts des Kaufs personalisierte Produktempfehlungen generiert. Des Weiteren wird ein Benchmark mit dem vorgestellten, einem auf Assoziationsregeln basierenden und einem ItemKNN-Algorithmus unter Verwendung realer Kaufdaten erstellt. Die beiden zuletzt genannten Algorithmen wurden der Open-Source-Bibliothek LibRec entnommen.**

SCHLÜSSELWÖRTER

Empfehlungssysteme, Assoziationsanalyse, Benchmark, LibRec

I. EINLEITUNG

Um den Kunden personalisierte Werbung anzeigen zu können, verwenden Betreiber von Online-Shops verschiedenste Empfehlungsalgorithmen. Diese können in die Kategorien Content Based-, Collaborative- (CF) und Hybrid Filtering unterteilt werden. Während Content Based Filtering ausschließlich die Kauf-/Ratinghistorie eines einzelnen Kunden für seine Kaufprognosen berücksichtigt, werden beim CF verschiedene Kunden und Produkte zur Erstellung von Kauf-/Ratingprognosen mit einbezogen. Als Hybrid Filtering wird eine Mischung der beiden zuvor genannten Varianten bezeichnet (vgl. [Kla09]).

Der im Folgenden als PAROT-Algorithmus bezeichnete Algorithmus basiert auf Assoziationsregeln, die den Kaufzeitpunkt des Produkts berücksichtigen. Somit handelt es sich um Hybrid Filtering. Um zu verifizieren, wie gut der PAROT-Algorithmus im Vergleich zu anderen Algorithmen abschneidet, werden zwei Open-Source-Algorithmen der LibRecBibliothek (www.librec.net) verwendet. Der erstellte Benchmark basiert auf Kaufdaten, welche binären Ratings entsprechen.

II. EMPFEHLUNGSSYSTEME IN DER PRAXIS

Mittlerweile gibt es eine Vielzahl an Empfehlungssystemen, die in der Praxis eingesetzt oder zur wissenschaftlichen Forschung verwendet werden. Sie basieren u.a. auf den oben genannten Ansätzen, die jedoch auf sehr vielfältige Weise umgesetzt werden. Empfehlungssysteme lassen sich bezüglich ihres Vorkommens in der Praxis grob in vier Gruppen einteilen:

- Empfehlungssysteme, die von Großkonzernen (z.B. Amazon und Netflix) selbst entwickelt und nur dort eingesetzt werden.
- Empfehlungssysteme, die als Software as a Service (SaaS) von IT-Unternehmen bereitgestellt und bei verschiedenen Unternehmen eingesetzt werden, die unter anderem aus Kosten- oder Wissensgründen keine eigenen Systeme entwickeln.
- Empfehlungssysteme, die von kleinen und mittelständischen Unternehmen selbst entwickelt und ausschließlich unternehmensintern eingesetzt werden.
- Empfehlungssysteme, die als Open-Source-Lösungen frei verfügbar sind, jedoch von den Unternehmen, die sie nutzen wollen, selbst in ihre IT-Systeme eingebunden werden müssen. Einige in der Forschung verwendete Empfehlungsalgorithmen fallen in diesen Bereich.

Zu Empfehlungssystemen aus der dritten Kategorie lassen sich im Allgemeinen keine Hinweise auf die verwendeten Algorithmen finden, da diese Empfehlungssysteme ausschließlich unternehmensintern verwendet werden.

Zwar lassen sich sehr viele Empfehlungssysteme aus den ersten beiden Kategorien finden, jedoch sind über sie nur eher oberflächliche Informationen dokumentiert. Großkonzerne und IT-Unternehmen, die Empfehlungssysteme kostenpflichtig entwickeln und anbieten, investieren oftmals viel Human- und Finanzkapital in ihre Systeme, damit ihre Empfehlungen treffsicherer sind als die ihre Konkurrenten. Wären zu viele Details über sie bekannt, könnten die Algorithmen nachgeahmt werden, was die Wettbewerbsfähigkeit beeinträchtigen und den Umsatz schmälern würde. Exemplarisch sind hier die Empfehlungssysteme von *Amazon*, *Netflix* sowie *Zalando* und als SaaS-Algorithmen Episerver Personalization (vgl. [EPI20] www.episerver.com) sowie *epoq internet services GmbH* (vgl.

[EPOa] www.epog.de) zu nennen.

- Amazon startete mit einem auf Item Based Collaborative Filtering basierenden Empfehlungssystem bereits 1998. Bis heute wurde dieser Ansatz beibehalten. Letztlich geht es darum, für Kunden individuell anhand ihrer Kauf- und Klickhistorie und ihrem aktuellen Verhalten auf der Homepage von Amazon ahnliche Produkte herauszusuchen. Dabei werden Produkte, die der Kunde bereits betrachtet oder gekauft hat, herausgefiltert und von den übrig gebliebenen die ausgespielt, für die sich der Kunde am wahrscheinlichsten interessieren wird (vgl. [SL17]).
- Das von Zalando verwendete personalisierte Empfehlungssystem basiert auf Künstlichen Neuronalen Netzen (KNN). Dazu werden für jeden Kunden seine Kauf-, Klick- und sonstigen Verhaltensdaten als Zeitreihe gespeichert. Trainiert wird dieses KNN ausschließlich mittels Kaufdaten, Ratingdaten werden nicht hinzugezogen. Die Kombination, welcher Kunde welches Produkt gekauft hat, wird in einer spärlich befüllten Kunden-Produkt-Matrix erfasst. Das KNN wird so trainiert, dass der durchschnittliche Cross-Entropie-Verlust für jeden Artikel über die Kunden minimiert wird. Mit einer logistischen Regression kann für jedes Feld der Kunden-Produkt-Matrix ein Wert geschätzt werden, der angibt, mit welcher Wahrscheinlichkeit ein bestimmter Kunde ein bestimmtes Produkt kaufen wird (vgl. [HBV17]).
  Um die zeitliche Abfolge mit einzubeziehen, wird im nächsten Schritt ein zweites KNN mit einem LSTM-Layer (Long short-term memory) hinzugefugt, in das die Ergebnisse des ersten Netzes und die Kaufhistorie des Kunden als Input einfließen. Als Resultat prognostiziert dieses Netz, welchen Kleidungsstil der Kunde bevorzugt und wie dieser sich mit der Zeit verandern wird (z.B. saisonale Unterschiede) (vgl. [HBV17]).
- Netflix verwendet eine Kombination aus verschiedenen Empfehlungsalgorithmen. Zu diesen zahlen u.a. der Personalized Video Ranker, der Top-N Video Ranker und die Video-Video Similarity. Mithilfe dieser Algorithmen entstehen für einen typischen Nutzer von Netflix ca. 1.000 verschiedene Reihen mit verschiedenen Schwerpunkten. Abschließend wird ein weiterer Algorithmus verwendet, um zu selektieren, welche dieser Reihen letztendlich für den Kunden auf der personalisierten Homepage angezeigt werden. Insgesamt basieren die von Netflix verwendeten Empfehlungsalgorithmen sowohl auf Ansätzen des überwachten Lernens wie Klassifikationen und Regressionen als auch auf Ansatzen des unüberwachten Lernens wie Dimensionsreduktion (z.B. Matrixfaktorisierung) (vgl. [GH15]).
- Der von Episerver verwendete Empfehlungsalgorithmus basiert nicht auf Regeln, sondern auf dem Verhaltensmuster von Besuchern und Besuchergruppen bzgl. ihrer Customer Journey. Für die Prognosen werden insbesondere die zu einem Nutzer ähnlichen Kunden berucksichtigt. Als Ziel generiert das Empfehlungssystem personalisierte Rankings von Produkten auf Basis von ahnlichen Kunden und inhaltsbasierten Empfehlungen (vgl. [EPI]).
  Diese Informationen über den Episerver-Empfehlungsalgorithmus lassen die Schlussfolgerung zu, dass es sich um einen hybriden Ansatz handelt, der nachbarschaftsbasiertes Collaborative Filtering mit einem Content Based Filtering Ansatz kombiniert.
- Das epoq-Empfehlungssystem analysiert das Kauf- und Klickverhalten der Nutzer. Mithilfe eines wissensbasierten Ansatzes werden damit personalisierte Empfehlungen ermittelt und ausgespielt. Falls über einen Nutzer noch keine Informationen bekannt sind, wird ihm keine personalisierte Empfehlung, sondern eine Bestsellerliste angezeigt. Die Empfehlungen basieren auf den Informationen, was andere Kunden gekauft haben (Kundenähnlichkeit) und inhaltlich passenden Produkten (vgl. [EPOb]).
  Es handelt sich beim epoq-Empfehlungsalgorithmus vermutlich um eine User Based Collaborative Filtering Variante, die mit einem Content Based Filtering kombiniert wird.

Viele in der Forschung verwendete Empfehlungssysteme fallen in den Bereich der Open-Source-Lösungen (vierte Kategorie). Einige dieser Algorithmen wurden sogar explizit von akademischen Einrichtungen entworfen, was dazu fuhrt, dass sie entweder bereits gut dokumentiert sind oder aufgrund des frei verfügbaren Quellcodes bis ins kleinste Detail analysiert werden können. Zu diesen Empfehlungssystemen zählen unter anderem die LibRec-Bibliothek und MyMediaLite.

- LibRec ist eine Java-Bibliothek für Empfehlungssysteme. Sie enthalt eine Auswahl an verschiedenen Empfehlungsansatzen, die sowohl für das Erstellen von Rating- als auch Produktprognosen verwendet werden können. Darüber hinaus bietet sie vorimplementiert verschiedene Ähnlichkeitsmaße an, die in den Algorithmen zum Einsatz kommen können und Methoden zur Datenaufbereitung und Berechnung von Metriken wie Precision und RMSE. Die Empfehlungsalgorithmen von LibRec können vielfältig eingesetzt werden, da sie je nach Ausrichtung Beziehungen zwischen Produkten, Kunden, Produkten und Kunden aber auch Beziehungen innerhalb von sozialen Informationen analysieren und darauf aufbauend Empfehlungen erstellen können (vgl. [Guo+15], [LIB17] www.librec.net).
- Die MyMediaLite Recommender System Library wurde von einem Team der Universität Hildesheim erstellt. Sie enthalt sowohl Algorithmen, die Ratings prognostizieren, als auch welche, die Rankings erstellen. Neben Basisalgorithmen wie dem Most Popular Recommender sind klassische Empfehlungsansätze wie ItemKNN und UserKNN implementiert. Zusätzlich sind Algorithmen, die eine Matrixfaktorisierung enthalten, wie z.B. BPRMF (Bayesian Personalized Ranking Matrix Factorization) und WRMF (Weighted Regularized Matrix Factorization), in der Bibliothek vorhanden. Die Algorithmen können un-

ter anderem durch verschiedene Parametereinstellungen und Ähnlichkeitsmaße modifiziert werden. MyMediaLite stellt dazu Klassen zur Berechnung der Ahnlichkeit (z.B. Jaccard Koeffizient, Pearsonscher Korrelationskoeffizient, Kosinusähnlichkeit) und Metriken zur Bewertung der Algorithmen wie den MAE, RMSE und prec@N bereit (vgl. [Gan+] und [Gan+15]).

III. DER PAROT-ALGORITHMUS

A. Datenaufbereitung

Für den PAROT-Algorithmus werden Datensätze benötigt, die eine eindeutige Kundennummer (Menge aller Kundennummern: ID), eine Produktnummer (Menge aller Produktnummern: Item) und den Kaufzeitpunkt (Menge aller Kauftage: Date) auf Tagesbasis enthalten. Zunächst müssen die Daten so aufbereitet werden, dass jede Transaktion, bestehend aus einer ID, Item und Date, nach ihren Kaufdaten einem der vier Zeiträume *Input*, *Target*, *Apply* und *Validate* zugeordnet werden kann (siehe Abb. 1). Die Zeiträume *Input* und *Apply* sowie *Target* und *Validate* sind jeweils gleich lang.

Abbildung 1: Einteilung in Zeiträume

Auf Basis der Daten in *Input* und *Target* wird ein Prognosemodell entwickelt. Dieses wird im nächsten Schritt auf den Zeitraum *Apply* angewandt. Mithilfe der Daten des *Validate*-Zeitraums kann anschließend verifiziert werden, wie zutreffend die erstellten Prognosen waren. Zusätzlich zur Einteilung in die vier Zeiträume werden die zu *Input* und *Apply* gehörenden Daten in Segmente unterteilt (Menge aller Segmente: Segment). Die Segmente sind von ihrer Länge her so zu wählen, dass sie länger werden, je weiter die Daten in der Vergangenheit liegen. Die Anzahl an Tagen in den einzelnen Segmenten ist für *Input* und *Apply* gleich.

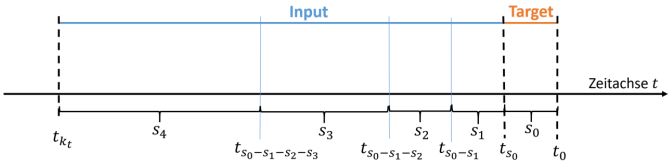

Abbildung 2: Einteilung in Segmente

Da im weiteren Verlauf lediglich die Segmentzuweisung bzw. die Einteilung in *Target* und *Validate* von Bedeutung ist, muss das genaue Kaufdatum nicht weiter berücksichtigt werden. Transaktionen, die in den Zeiträumen *Input* und *Apply* liegen können eindeutig mit

$$T_i := (ID_j, It_I, Seg_l) \in I, mit$$
$$ID_j \in ID, It_I \in Item, Seg_l \in Segment$$

beschrieben werden. Die Menge aller Transaktionen im *Input*-Zeitraum ist $I := \{T_1, \cdots, T_n\}$. Die Beschreibung der Transaktionen in *Apply* erfolgt analog. Da für die Zeiträume *Target* und *Validate* keine Segmenteinteilung erfolgt, können die Transaktionen dort folgendermaßen definiert werden:

$$T_i := (ID_j, It_T) \in T, mit$$
$$ID_j \in ID, It_T \in Item.$$

Die Menge aller Transaktionen im Target-Zeitraum ist $T := \{T_{n+1}, \cdots, T_m\}$.

B. Modellbildung

Nach der Datenaufbereitung erfolgt die Modellentwicklung: Zunächst wird überprüft, ob jedes der Produkte $It_I \in T_i \in I$ (*Input*-Zeitraum) und $It_T \in T_i \in T$ (*Target*-Zeitraum) häufig genug vorkommt, um als signifikant betrachtet zu werden. Dies entspricht dem Ermitteln des Item-Supports. Mithilfe der als signifikant eingestuften Transaktionen des *Input*- und *Target* Zeitraums werden Assoziationsregeln der Form

$$(It_I \rightarrow It_T) \text{ in Zeitraum } Seg_l$$

erstellt, was bedeutet: „Kunden, die Produkt It_I in Zeitraum Seg_l gekauft haben, kauften im Target-Zeitraum Produkt It_T“. Auf diese Weise werden herkömmliche Assoziationsregeln um die Information des Kaufsegments ergänzt. Im Gegensatz zu dem in [ES00] beschriebenen A priori-Algorithmus, der mengenorientiert häufig vorkommende Produktkombinationen in Warenkörben bestimmt, werden hier nur Paare von Produkten betrachtet, deren Item-Support im Input- bzw. Target-Zeitraum ausreichend hoch ist. Für jede der gefundenen Assoziationsregeln wird anschließend ein Score-Wert ermittelt. Dies erfolgt über die Berechnung des Added-Value:

$$Score_{It_I It_T Seg_l} := P(It_T|It_I) - P(It_T).$$

Damit kann für jeden Kunden für jede seiner Produktkombinationen der zugehörige Score-Wert angegeben werden. Da das Ziel des Algorithmus ist, zu prognostizieren, welches Produkt ein Kunde auf Basis seiner vorherigen Käufe kaufen wird, werden die einzelnen Score-Werte pro Kunde

und $It_I \in T_i \in I$ aufsummiert. Daraus ergibt sich für jeden Kunden und die Produkte $It_T \in T_i \in T$ des Target-Zeitraums eine zugehörige Score-Summe $Sum_{It_T}ID_iSeg_l$ (vereinfachte Schreibweise im Folgenden als Sum_l). Für jeden Kunden sind die Produkte $It_T \in T_i \in T$ mit den höchsten Score-Summen die Produkte, die er am wahrscheinlichsten als nächstes kaufen wird. Die Kombination aus Kundennummer, Produkt im *Target*-Zeitraum und Score-Summe kann bereits als Grundmodell zur Produktprognose verwendet werden. Aufbauend auf das Grundmodell können Kaufwahrscheinlichkeiten von Kunden für Produkte im *Target*-Zeitraum berechnet werden. Für die verschiedenen Produkte $It_T \in T_i \in T$ wird die Häufigkeit (*qty*) der verschiedenen zugehörigen Score-Summen ($It_T qty_{Sum_l}$) bestimmt, was der Anzahl potenzieller Käufer entspricht. Außerdem wird gezählt, wie viele der Kunden, denen ein bestimmtes Produkt It_T mit einer Score-Summe Sum_l zugewiesen wurde, dieses tatsächlich im *Target*-Zeitraum gekauft haben. Dies entspricht der Kaufhäufigkeit ${}_{It_T}bought_{Sum_l}$.

Danach werden pro $It_T \in T_i \in T$ Gruppennummern startend bei 1 absteigend nach den Score-Summen Sum_l vergeben. Anschließend werden die Gruppen pro Produkt It_T so zusammengefasst, dass sie eine signifikante Aussagekraft haben. Dazu wird zunachstüberprüft, ob in einer Gruppe die Anzahl potenzieller Käufer (${}_{It_T}qty_{Sum_l}$) eine zuvor festgelegten Mindestanzahl überschreitet. Ist dies nicht der Fall, wird diese Gruppe mit der vorherigen vereinigt und die kleinere der beiden Score-Summe verwendet. Die Kaufquoten und die Häufigkeit der Score-Summen der beiden Gruppen werden jeweils addiert. Die Gruppen werden so lange auf diese Weise aggregiert, bis alle Gruppen die Mindestanzahl überschreiten bzw. nur noch eine Gruppe übrig ist. Anschließend wird die Kaufquote pro Produkt It_T und Gruppe berechnet:

$$Quote := \frac{{}_{It_T}bought_{Sum_l}}{{}_{It_T}qty_{Sum_l}}.$$

Die nun übrigen Gruppen werden auf Basis der Kaufquote so zusammengelegt, dass die Kaufquote fur jedes Produkt It_T monoton fallend für die Gruppen sind. Damit ist die Modellbildung abgeschlossen und kann im nächsten Schritt auf die neuen Daten des *Apply*-Zeitraums angewendet werden.

C. Anwendung auf neue Daten

Um personalisierte Empfehlungen zu erhalten, wird für jeden Kunden des *Apply*-Zeitraums zunächst überprüft, welche bei der Modellbildung erstellten Assoziationsregeln auf ihn zutreffen und mit welchem Score-Wert und welcher Kaufquote diese versehen sind. Mithilfe der Score-Werte kann erneut für jeden Kunden, jedes Produkt It_T pro Segment Seg_l die Score-Summe berechnet werden. Anschließend werden pro Kunde die Produkte It_T nach ihrer Kaufquote absteigend sortiert. Pro Kunde werden die $n \in \mathbb{N}$ Produkte mit der höchsten Kaufquote als Produktempfehlung verwendet.

IV. VERGLEICHSALGORITHMEN

Es gibt zwar einige Benchmarks, die Kennzahlen für Empfehlungsalgorithmen liefern, jedoch sind sie alle nicht für den PAROT-Algorithmus geeignet. Meist sind die Benchmarks nicht präzise genug dokumentiert (fehlende Angaben über verwendete Datensatze, Parameterwahl etc.) oder verwenden Datensätze, die nicht frei verfügbar bzw. auf den PAROT-Algorithmus aufgrund seiner zeitlichen Komponente (Kunden mussen mit bestimmtem Zeitabstand mindestens zwei Käufe getätigt / Wertungen abgegeben haben) nicht anwendbar sind (vgl. [Sar+02], [SB14]). Deshalb wird im Folgenden ein Benchmark mit zugehorigen Parametern für den PAROT-Algorithmus vorgestellt. Zur Erstellung des Benchmarks wurden zwei Open-Source-Algorithmen der LibRec-Bibliothek als Vergleichsempfehlungssysteme implementiert: der ItemKNN-Algorithmus, da dies eine Standardherangehensweise bei Empfehlungssystemen ist und der AssociationRule-Recommender, da er, ebenso wie der PAROT-Algorithmus, auf Assoziationsregeln basiert.

V. UMSETZUNG DES VERGLEICHS

A. Verwendeter Datensatz

Für den Benchmark wurde ein realer Kaufdatensatz eines Online-Versandhandels betrachtet:

Bezeichnung	**Wert**
Betrachtungszeitraum	1 Jahr
Anzahl Kunden	300.885
Anzahl Produkte	8.904
Anzahl Transaktionen	1.850.735

Tabelle I: Verwendeter Datensatz

B. Parameterwahl

Bei allen drei betrachteten Algorithmen sollen jeweils 10 Produkte pro Kunde prognostiziert werden. Für den ItemKNN-Algorithmus müssen zusätzlich lediglich die Parameter Nachbarschaftsgröße und das Ähnlichkeitsmaß vorab festgelegt werden. Als Optimierungsgröße wurde die F1-Metrik betrachtet. Nachdem die für Kaufdaten (binäre Daten) verwendbaren Ähnlichkeitsmaße mit verschiedenen Nachbarschaftsgrößen getestet wurden, stellte sich heraus, dass grundsatzlich beim hier verwendeten Datensatz das Jaccard-Maß hohere Werte für die $F1$-Metrik liefert. Nach einer Grid-Search wurde 25 als optimaler Wert für die Nachbarschaftsgröße gewählt. Der AssociationRule-Recommender bezieht weder den Support,

noch die Konfidenz oder sonstige Gütekriterien für Assoziationsregeln mit ein. Deshalb müssen für diesen Algorithmus keine Parameterwerte festgelegt werden. Beim PAROT-Algorithmus gibt es einige vorab zu wählende Parameter:

Parameter	Wert
Anzahl Segmente	5
Segmentlängen	7, 14, 28, 56, 365
Mindestsupport c_0	5
Mindest qty q_{min}	40

Tabelle II: Paramterwahl fur PAROT

C. Datensplit

Um eine Vergleichbarkeit der Algorithmen zu gewährleisten, wurde der Datensatz mittels eines chronologischen Splits geteilt. 95% der Daten stellen den Trainingsdatensatz dar, die übrigen 5% den Testdatensatz, was dem *Validate*-Zeitraum entspricht (siehe Abb. 3). Der Testdatensatz ist so klein gewählt, da Kaufprognosen erstellt werden, die für Online-Targeting genutzt werden sollen. Es ist deshalb sinnvoll, maximal das Kaufverhalten der Kunden in den nächsten zwei Wochen zu prognostizieren, was 5% des Datensatzes entspricht.

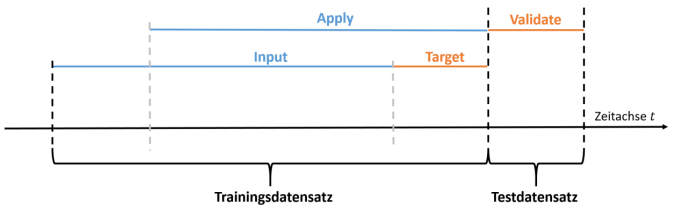

Abbildung 3: Datensplit

Ein Random-Split ist ungeeignet, da für den PAROT-Algorithmus die Daten nach Kundenzugehörigkeit aufgeteilt werden müssen (Kunden kommen entweder ausschließlich im Test- oder im Trainingsdatensatz vor). Für den ItemKNN- und AssociationRule-Recommender hingegen mussten die Kunden jeweils sowohl im Trainings- als auch im Testdatensatz vorhanden sein. Somit können für die drei Algorithmen nicht die gleichen Trainings- und Testdatensätze verwendet werden, womit die Ergebnisse nur eingeschränkt vergleichbar wären.

D. Vergleichsmetriken

Da es sich um Produktprognosen handelt, werden als Metriken zum Vergleich zwischen den verschiedenen Algorithmen die Precision, der Recall und die $F1$-Metrik berechnet. Als Vergleich mit den prognostizierten Produkten werden die von den Kunden tatsächlich im Testdatensatz gekauften Produkte verwendet. Außerdem werden sowohl Item- als auch User-Coverage berechnet, um eine Aussage über die Reichweite der Kaufempfehlungen treffen zu können. Abschließend wird mit dem Exakten Fisher Test überprüft, ob die Hypothese der stochastischen Unabhängigkeit von Empfehlungen und tatsächlichen Käufen abgelehnt werden kann. Der Chi-Quadrat-Test ist für diese Verifikation nicht anwendbar, da das Auftreten der Beobachtungen für ein Element der Kontingenztafel (insbesondere die Anzahl der Produkte, die prognostiziert und gekauft wurden) im Allgemeinen beim betrachteten Datensatz zu klein ist (weniger als fünf Beobachtungen).

E. Theoretischer Vergleich

Betrachtet man die verschiedenen Algorithmen zunächst nur unter Berücksichtigung ihrer Herangehensweise, fällt auf, dass für den ItemKNN-Algorithmus eine Berechnung der Produktähnlichkeiten erforderlich ist. Die Güte der generierten Empfehlungen beim ItemKNN-Algorithmus hängt in hohem Maße von der verwendeten Ahnlichkeitsberechnung ab. Aufgrund der großen Anzahl an Ähnlichkeitsmaßen kann es vorkommen, dass das am besten geeignete schlichtweg nicht betrachtet wird. Dies ist ein Nachteil gegenüber dem AR- bzw. PAROT-Algorithmus, da diese Algorithmen keine Produktähnlichkeiten verwenden. Da beim AR-Algorithmus von LibRec keine Gütekriterien wie Support oder Konfidenz bestimmt werden, werden alle im Betrachtungszeitraum vorhandenen Produkte für das Bilden der Assoziationsregeln verwendet. Im PAROT-Algorithmus hingegen wird zunächst für jedes Produkt überprüft, ob die Häufigkeit des Produkts den vorgegebenen Mindestsupport übertrifft. Das Zählen der Häufigkeiten der Produkte lässt sich im Programm effizient umsetzen. Dem Apriori-Algorithmus folgend, werden ausschließlich Assoziationsregeln aus Produkten gebildet, deren Auftreten als häufig genug eingestuft wurde. Somit ist die Anzahl der im PAROT-Algorithmus gebildeten Assoziationsregeln deutlich geringer als im AR-Algorithmus, wodurch der PAROT-Algorithmus schneller Produktempfehlungen generieren kann und somit zeiteffizienter ist.

F. Vergleich anhand realer Kaufdaten

	Prec@10	Rec@10	F1	Item-Coverage	User-Coverage	exakter Fisher Test
ItemKNN	0,1176	0,4018	0,1820	93,56%	100,00%	99,97%
AR	0,1175	0,3665	0,1779	57,29%	100,00%	99,97%
PAROT	0,2143	0,3821	0,2746	3,74%	3,46%	100,00%

Abbildung 4: Benchmark

Wie sich Abb. 4 entnehmen lasst, liegt die Stärke des ItemKNN- und des AR-Algorithmus in ihrer Reichweite. Der ItemKNN-Algorithmus weist eine Item-Coverage mit über 90% auf, also einen deutlich höheren Wert, als die anderen beiden Algorithmen. Somit decken die Empfehlungen bei diesem Algorithmus nahezu die gesamte Produktpalette ab. Daraus folgt, dass der ItemKNN-Algorithmus in der Lage ist, auch ungewöhnliche und selten gekaufte Produkte zu empfehlen. Eine hohe Item-Coverage kann zwar für einen guten Algorithmus sprechen, beim betrachteten Datensatz handelt es sich jedoch um die Kaufdaten eines Online-Versandhandels,

dessen Produktpalette überwiegend aus Saisonartikeln wie Schuhen und Kleidung besteht. Infolgedessen ist eine Item-Coverage von 90% sogar eher nachteilig, da dies bedeutet, dass einigen Kunden nicht zur Saison passende Produkte empfohlen werden, die für den Kunden daher nicht interessant und ggf. momentan auch nicht verfügbar sind (z.B. die Empfehlung von Winterstiefeln zum Sommerbeginn). Eine Item-Coverage von 60% oder weniger ist beim betrachteten Datensatz aus diesem Grund besser. Der ItemKNN- und AR-Algorithmus schneiden dafür vergleichsweise schlecht bezüglich ihrer Vorhersagegenauigkeit ab - insbesondere in puncto Prec@10-Metrik und damit in der $F1$-Metrik. Der PAROT-Algorithmus zeigt hingegen gerade hinsichtlich der Präzision der Empfehlungen seine Stärke, was zu höheren $F1$- Werten fuhrt. Mit dem Exakten Fisher Test wird überprüft, ob die Vermutung, dass die erstellten Kaufprognosen und die tatsächlichen Käufe stochastisch unabhängig voneinander sind, verworfen werden kann. Der Hypothesentest wird dazu für jeden Kunden einzeln durchgeführt. Der Prozentwert in Abbildung 4 gibt an, wie groß der Anteil der Kunden, bei denen die Vermutung der Unabhängigkeit bei einer Wahl von $\alpha = 5\%$ abgelehnt wurde, an allen Kunden des Testzeitraums ist. Da dieser Wert bei allen betrachteten Algorithmen nahezu 100% ist, kann angenommen werden, dass die von den Algorithmen generierten Kaufprognosen signifikant sind. Ob der PAROT- oder der AR- bzw. ItemKNN-Algorithmus besser fur einen Shop geeignet ist, lässt sich aufgrund der entgegengesetzten Stärken und Schwächen nicht beurteilen und hängt letztlich von der Zielsetzung des Shops ab. Da ItemKNN- und AR-Algorithmus die gleichen Stärken und Schwächen haben, können sie, im Gegensatz zum PAROT-Algorithmus, exakter miteinander verglichen werden: Es wirkt so, als ob der ItemKNN- den AR-Algorithmus schlägt, da alle Werte der verwendeten Vergleichsmetriken beim ItemKNN-Algorithmus besser sind. Jedoch ist zu berücksichtigen, dass die sehr hohe Item-Coverage bei diesem Datensatz gegen den ItemKNN-Algorithmus spricht (Begründung siehe oben). Deshalb ist eine klare Aussage, welcher der beiden Algorithmen tatsächlich geeigneter ist, nicht möglich.

G. Weitere Analyse des PAROT-Algorithmus

Obwohl der PAROT-Algorithmus auf Assoziationsregeln basiert und diese im Grunde lediglich weiter verarbeitet, schneidet er bezüglich der $F1$-Metrik deutlich besser und bei Betrachtung der User- und Item-Coverage schlechter ab, als der AR-Algorithmus. Welche Unterschiede zwischen den beiden Algorithmen dazu fuhren, dass ihre Stärken und Schwächen genau entgegengesetzt sind, wird in diesem Abschnitt ermittelt. Die Abweichungen zwischen dem PAROT- und dem AR-Algorithmus sind folgende:

- Nach dem Bilden und Bewerten der Assoziationsregeln werden diese beim PAROT-Algorithmus, zu Gruppen zusammengefasst. Auf Basis der Kaufquote für die Gruppen lasst sich für einen Kunden eine Kaufwahrscheinlichkeit für ein Produkt ermitteln.
- Beim AR-Algorithmus wird der gesamte Betrachtungszeitraum als ein einziger Warenkorb pro Kunde betrachtet. Die Assoziationsregeln werden jeweils mittels zweier Produkte aus einem gemeinsamen Warenkorb gebildet. Im Gegensatz dazu fließt beim PAROT-Algorithmus eine zeitliche Komponente mit ein: Der Betrachtungszeitraum wird in die beiden Zeitabschnitte *Input* und *Target* aufgeteilt und die Regeln so gebildet, dass jeweils ein Produkt des *Inputs* auf ein Produkt des *Targets* verweist.
- Neben der Aufteilung in *Input* und *Target* berücksichtigt der PAROT-Algorithmus einen weiteren zeitlichen Aspekt: Durch die Einteilung des Inputs in Segmente und dem späteren segmentierten Assoziationsregelbilden soll erfasst werden, ob Produkte in regelmäßigen Abstanden zueinander gekauft werden.
- Ein weiterer Unterschied ist die Verwendung einer support-ähnliche Größe im PAROT-Algorithmus: Tritt eine erstellte Regel zu selten auf, wird sie als nicht signifikant eingestuft und nicht zum Generieren der Empfehlungen verwendet.

Wird der PAROT-Algorithmus dahingehend angepasst, dass jeweils eins der zuvor genannten Merkmale abgeändert wird (Betrachtung verschiedener Werte fur den Mindestsupport, keine Aufteilung des Datensatzes in Segmente etc.), lassen sich folgende Ergebnisse feststellen:

		Prec@10	Rec@10	F1	Item-Coverage	User-Coverage	exakter Fisher Test
ItemKNN		0,1176	0,4018	0,1820	93,56%	100,00%	99,97%
AR		0,1175	0,3665	0,1779	57,29%	100,00%	99,97%
ursprünglicher PAROT		0,2143	0,3821	0,2746	3,74%	3,46%	100,00%
PAROT	c_o = 1	0,1162	0,4773	0,1869	46,98%	4,71%	100,00%
	c_o = 2	0,1456	0,4505	0,2201	24,22%	4,45%	100,00%
	c_o = 3	0,1709	0,4164	0,2423	10,96%	4,09%	100,00%
	c_o = 5	0,2143	0,3821	0,2746	3,74%	3,46%	100,00%
	c_o = 7	0,2342	0,3475	0,2798	1,90%	2,93%	100,00%
	c_o = 10	0,3162	0,3313	0,3236	1,10%	2,40%	100,00%
	c_o = 13	0,3848	0,3231	0,3513	0,68%	2,08%	100,00%
	c_o = 15	0,4129	0,3181	0,3593	0,57%	1,95%	100,00%
	c_o = 17	0,4092	0,3106	0,3532	0,46%	1,72%	100,00%
	c_o = 20	0,4255	0,3032	0,3541	0,38%	1,48%	100,00%
	c_o = 23	0,4652	0,3024	0,3665	0,32%	1,39%	100,00%
	c_o = 25	0,4668	0,3059	0,3696	0,29%	1,25%	100,00%

Abbildung 5: Mindestsupport

		Prec@10	Rec@10	F1	Item-Coverage	User-Coverage	exakter Fisher Test
ItemKNN		0,1176	0,4018	0,1820	93,56%	100,00%	99,97%
AR		0,1175	0,3665	0,1779	57,29%	100,00%	99,97%
ursprünglicher PAROT		0,2143	0,3821	0,2746	3,74%	3,46%	100,00%
PAROT	c_o = 1	0,1067	0,5197	0,1771	10,32%	4,70%	100,00%
	c_o = 3	0,1411	0,4983	0,2199	12,79%	4,48%	100,00%
	c_o = 5	0,1957	0,4797	0,2780	5,00%	4,07%	100,00%
	c_o = 10	0,3427	0,4527	0,3900	1,70%	3,24%	100,00%
	c_o = 15	0,3925	0,3594	0,3752	0,79%	2,59%	100,00%
	c_o = 20	0,4384	0,3431	0,3849	0,57%	2,21%	100,00%

Abbildung 6: ohne Segmente

Aus der Analyse des PAROT-Algorithmus lässt sich schließen, dass der Grouping-Abschnitt (siehe Abb. 7) des Programms keine nennenswerte Veränderung der Vergleichsmetriken verursacht. Er ist lediglich zur Ermittlung von Kaufwahrscheinlichkeiten sinnvoll. Eine Einteilung in Segmente (siehe Abb. 6) (zumindest bei dem hier verwendeten Datensatz) verschlechtert die Ergebnisse tendenziell. Außerdem haben sowohl die Einteilung in *Input* und *Target* (siehe Abb. 8) als auch

		Prec@10	Rec@10	F1	Item-Coverage	User-Coverage	exakter Fisher Test
ItemKNN		0,1176	0,4018	0,1820	93,56%	100,00%	99,97%
AR		0,1175	0,3665	0,1779	57,29%	100,00%	99,97%
ursprünglicher PAROT		0,2143	0,3821	0,2746	3,74%	3,46%	100,00%
PAROT	c_o = 1	0,1153	0,4702	0,1852	46,45%	4,71%	99,96%
	c_o = 2	0,1440	0,4421	0,2173	20,64%	4,45%	100,00%
	c_o = 3	0,1690	0,4090	0,2392	9,39%	4,09%	100,00%
	c_o = 5	0,2094	0,3793	0,2698	3,63%	3,47%	100,00%
	c_o = 7	0,2228	0,3419	0,2698	1,88%	2,97%	100,00%
	c_o = 10	0,2939	0,3263	0,3092	1,08%	2,48%	100,00%
	c_o = 13	0,3630	0,3181	0,3390	0,67%	2,12%	100,00%
	c_o = 15	0,3892	0,3116	0,3461	0,57%	2,01%	100,00%
	c_o = 17	0,3841	0,3031	0,3388	0,46%	1,81%	100,00%
	c_o = 20	0,4068	0,2978	0,3439	0,38%	1,62%	100,00%
	c_o = 23	0,4503	0,2964	0,3575	0,32%	1,53%	100,00%
	c_o = 25	0,4528	0,2984	0,3597	0,29%	1,43%	100,00%

Abbildung 7: ohne Gruppierung

		Prec@10	Rec@10	F1	Item-Coverage	User-Coverage	exakter Fisher Test
ItemKNN		0,1176	0,4018	0,1820	93,56%	100,00%	99,97%
AR		0,1175	0,3665	0,1779	57,29%	100,00%	99,97%
ursprünglicher PAROT		0,2143	0,3821	0,2746	3,74%	3,46%	100,00%
PAROT ohne Grouping	c_o = 1	0,1153	0,4702	0,1852	46,45%	4,71%	99,96%
	c_o = 5	0,2094	0,3793	0,2698	3,63%	3,47%	100,00%
	c_o = 10	0,2939	0,3263	0,3092	1,08%	2,48%	100,00%
	c_o = 15	0,3892	0,3116	0,3461	0,57%	2,01%	100,00%
	c_o = 20	0,4068	0,2978	0,3439	0,38%	1,62%	100,00%
PAROT ohne Input und Target	c_o = 1	0,1232	0,6106	0,2050	97,25%	97,41%	100,00%
	c_o = 5	0,1241	0,6096	0,2062	77,79%	97,41%	99,98%
	c_o = 10	0,1259	0,6101	0,2087	71,12%	97,41%	99,98%
	c_o = 15	0,1286	0,6096	0,2124	67,20%	97,41%	99,98%
	c_o = 20	0,1321	0,6103	0,2172	63,38%	97,41%	100,00%

Abbildung 8: ohne Input & Target

die Verwendung einer Mindesthäufigkeit (siehe Abb. 5) einen großen Einfluss auf die Vergleichsmetriken: Die Einteilung des Betrachtungszeitraums in *Input* und *Target* sorgt zwar für eine Verbesserung der $F1$-Metrik, dies geschieht jedoch zulasten der Item- und User-Coverage. Ähnlich verhält es sich mit den Auswirkungen des Mindestsupports. Bei steigenden Werten für den Mindestsupport nimmt die Item- und User-Coverage drastisch ab, dafür verbessert sich die $F1$-Metrik deutlich. Insbesondere diese beiden zuletzt genannten Besonderheiten des PAROT-Algorithmus sorgen für die unterschiedlichen Stärken und Schwächen des PAROT- und AR-Algorithmus.

VI. FAZIT

Da kein in der Literatur verfügbarer Benchmark für eine Beurteilung des PAROT-Algorithmus geeignet ist, wurde ein eigener Benchmark erstellt. Dazu wurde der PAROT-Algorithmus mit den Open-Source-Algorithmen ItemKNN und AssociationRule-Recommender (abgekürzt mir Folgenden als AR) der LibRec-Bibliothek sowohl theoretisch als auch mittels realer Kaufdaten verglichen. Der ItemKNNAlgorithmus ist ein klassisches Item Based Collaborative Filtering, das zum Memory-based Collaborative Filtering gezählt wird. Der AR-Algorithmus kann als Model-based Collaborative Filtering kategorisiert werden, während der PAROT-Algorithmus ein hybrider Ansatz ist, der in diesem Fall aus einer Assoziationsanalyse mit zeitlicher Komponente und einem Clusterverfahren besteht. Als Vergleichsmetriken wurden die Precision Prec@10, der Recall Rec@10, die $F1$-Metrik, der MRR, die User Coverage, die Item Coverage und der Exakte Fisher Test betrachtet.

Zusammenfassend lässt sich festhalten, dass die Stärke des PAROT-Algorithmus seine gute Vorhersagegenauigkeit ist, was insbesondere für die hohen Werte der $F1$-Metrik gilt. Als seine Schwache ließ sich seine mangelnde Reichweite, was sich in sehr niedrigen Werten der User- und Item-Coverage widerspiegelt, identifizieren. Die Vergleichsergebnisse des ItemKNNund AR-Algorithmus sind genau gegensätzlich: Die Schwäche dieser beiden Algorithmen ist ihre niedrige Vorhersagegenauigkeit, ihre Stärke die große Reichweite. Ein Vergleich von ItemKNN- und AR-Algorithmus zeigte zudem, dass sich keine Aussage drüber treffen lässt, welcher der beiden Algorithmen zu favorisieren ist.

Um zu analysieren, warum der PAROT- den ARAlgorithmus bezüglich der $F1$-Metrik deutlich übertrifft und hinsichtlich der Item- und User-Coverage massiv unterliegt, obwohl beide Algorithmen auf Assoziationsregeln basieren, wurde zunächst ermittelt, in welchen Aspekten sich die beiden Algorithmen unterscheiden: Im Gegensatz zum ARAlgorithmus erfolgt im PAROT-Algorithmus nach dem Bilden der Assoziationsregeln eine Gruppierung. Außerdem beinhaltet der PAROT-Algorithmus zwei zeitliche Komponenten: Zum einen wird der Betrachtungszeitraum in die beiden Zeiträume Input und Target unterteilt, zum anderen wird der InputZeitraum in Segmente gegliedert. Ein weiterer Unterschied zwischen den beiden Algorithmen ist die Verwendung der Mindesthäufigkeit (eine support ähnliche Größe) im PAROT-Algorithmus. Jeder dieser vier die Unterschiede ausmachenden Bestandteile wurde einzeln betrachtet und jeweils aus dem PAROT-Algorithmus entfernt, um zu überprüfen, welchen Einfluss er auf die verwendeten Vergleichsmetriken hat.

Als maßgebliche Einflussfaktoren auf die gegensätzlichen Stärken und Schwächen des PAROT- und AR-Algorithmus sind die Einteilung des Betrachtungszeitraums in Input und Target sowie die Verwendung einer Mindesthäufigkeit auszumachen: Die Einteilung in Input und Target verringert zwar die Reichweite der Empfehlungen, was die User- und Item-Coverage widerspiegeln, jedoch ist nur auf Basis dieser Einteilung eine Verbesserung der $F1$-Metrik bei steigender Mindesthaufigkeit möglich.

VII. AUSBLICK

Aufbauend auf die hier gewonnenen Erkenntnisse konnte überprüft werden, wie sich das Ergänzen der Produktprognosen des PAROT-Algorithmus mit den Empfehlungen eines Short-Term-Score, der das aktuelle Klickverhalten der Kunden analysiert, auf die verwendeten Vergleichsmetriken auswirkt. Die Vermutung dabei ist, dass sich zumindest die Werte der User-Coverage erhöhen werden. Beim PAROT-Algorithmus wird die User-Coverage dadurch beeintrachtigt, dass nur Kunden betrachtet werden, die sowohl im *Input*- als auch im *Target*-Zeitraum Kaufe tätigten. Durch das Betrachten der momentan aktiven Kunden beim Short-Term-Score ist für diese Kunden ihre aktuelle Kaufhistorie bekannt, was eine Vorhersage für sie ermöglicht.

Außerdem sollten neben dem hier verwendeten Datensatz weitere Datensätze mit anderen Strukturen betrachtet werden, die beispielsweise aus ausschließlich nichtsaisonalen Produkten oder nur aus saisonalen Produkten bestehen. So ließe sich ermitteln, ob die zuvor beschriebenen Auswirkungen der vier Unterschiede zwischen AR- und PAROT-Algorithmus als allgemeingültig betrachtet werden können.

Zusätzlich wären weitere Vergleiche mit anderen Algorithmen interessant, die unter anderem zeitliche Komponenten oder eine Matrixfaktorisierung enthalten. Damit konnte verifiziert werden, ob es andere Algorithmen gibt, deren Vorhersagegenauigkeit ebenfalls so gut ist oder ob dies ein Alleinstellungsmerkmal des PAROT-Algorithmus ist. Des Weiteren wäre ein Vergleich mit Künstlichen Neuronalen Netzen möglich. Diese werden, wie in Abschnitt II angedeutet, bereits in der Praxis verwendet, wurden jedoch in der Forschung bisher im Rahmen von Empfehlungsalgorithmen kaum betrachtet und sind derzeit in den gangigen Open-Source-Bibliotheken für Empfehlungsalgorithmen nicht vorhanden.

Abschließend lässt sich sagen, dass der PAROT-Algorithmus im Vergleich zu anderen Algorithmen durch seine präzisen Produktprognosen hervorsticht, was jedoch die Reichweite verringert. Wurde der PAROT-Algorithmus mit einem Short-Term-Algorithmus kombiniert, ließe sich dieser Nachteil womöglich beheben, was in weiteren Arbeiten untersucht werden konnte.

LITERATUR

[EPI] GmbH EPISERVER. *Episerver by the numbers*. URL: https://www.episerver.com/company/.

[EPI20] GmbH EPISERVER. *Bereitstellung eines kanalubergreifenden, personalisierten Einkauferlebnisses.* 14.02.2020. URL: https://www.episerver.de/produkte/plattform/episerver-personalization/.

[EPOa] GmbH EPOQ. *Softwareentwicklung mit Empathie, Sorgfalt, Freundschaft und Herzblut.* URL: https://www.epoq.de/company/.

[EPOb] GmbH EPOQ. *Was ist eine Recommendation Engine.* URL: https://www.epoq.de/produkte/%20recommendation-engine/.

[ES00] Martin Ester und Jörg Sander. *Knowledge Discovery in Databases - Techniken und Anwendungen*. Berlin Heidelberg: Springer-Verlag Berlin Heidelberg, 2000. ISBN: 978-3-540-67328-6. DOI: 10.1007/978-3-642-58331-5.

[Gan+] Zeno Gantner u. a. *Recommender System Library*. URL: http://www.mymedialite.net/.

[Gan+15] Zeno Gantner u. a. *Class List*. 31.12.2015. URL: http://mymedialite.net/documentation/doxygen/annotated.html.

[GH15] Carlos Alberto Gomez-Uribe und Neil Hunt. "The Netflix Recommender System: Algorithms, Business Value, and Innovation". In: *ACM Trans. Management Inf. Syst.* 6 (2015), 13:1–13:19.

[Guo+15] Guibing Guo u. a. "LibRec: A Java Library for Recommender Systems". In: *UMAP Workshops*. 2015.

[HBV17] Sebastian Heinz, Christian Bracher und Roland Vollgraf. *An LSTM-Based Dynamic Customer Model for Fashion Recommendation*. 2017. URL: http://arxiv.org/pdf/1708.07347v1.

[Kla09] André Klahold. *Empfehlungssysteme: Recommender Systems - Grundlagen, Konzepte und Lösungen*. 1. Aufl. IT-Management und -Anwendungen. Wiesbaden: Vieweg+Teubner Verlag / GWV Fachverlage GmbH Wiesbaden, 2009. ISBN: 9783834805683. DOI: 10.1007/978-3-8348-9558-5.

[LIB17] LIBREC. *A Simple Tutorial on the LibRec - Overview.* 13.05.2017. URL: https://www.librec.net/example.html.

[Sar+02] Badrul M Sarwar u. a. "Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering". In: *Proceedings of the fifth international conference on computer and information technology*. Bd. 1. 2002, S. 291–324.

[SB14] Alan Said und Alejandro Bellogın. "Comparative recommender system evaluation: benchmarking recommendation frameworks". In: *Proceedings of the 8th ACM Conference on Recommender systems*. 2014, S. 129–136.

[SL17] Brent Smith und Greg Linden. "Two decades of recommender systems at amazon. com". In: *Ieee internet computing* 21.3 (2017), S. 12–18.