Freie Universität Berlin

Bachelorarbeit

Entwicklung eines RSS-Aggregators mit kaskadiertem Empfehlungssystem unter Verwendung von implizitem und explizitem Feedback

Johannes Mihan

Betreuer / Gutachter
Dipl. Marko Harasic

Zeitgutachter
Prof. Dr. Claudia Müller-Birn

11. November 2016

Abstract

Das heutige Internet bietet einem Nutzer eine Vielzahl verschiedener Informationsquellen. Eine Möglichkeit, für Nutzer zu bestimmten Webquellen aktuelle Artikel zu beziehen, ist das Abonnieren sogenannter RSS-Feeds (Rich Site Summary). Das Ziel dieser Bachelorarbeit umfasst die Umsetzung eines Empfehlungssystems mit Benutzermodell für einen RSS-Aggregators. Der RSS-Aggregator ermöglicht dem Nutzer RSS-Quellen zu abonnieren, die enthaltenen Feeds zu lesen und diese nach dem Datum oder von einem Empfehlungssystem geordneten Liste anzuzeigen. Dazu wurden Einführend Bedingungen an den Aggregators gestellt. Anhand der Bedingungen wurde ein kaskadiertes Empfehlungssystem entwickelt, das in der ersten Kaskade ein kollaboratives Empfehlungssystem zum entdecken neuer Quellen und in der zweiten Kaskade ein inhaltlichen zur Ordnung der abgerufenen Feeds verwendet.

Eidesstattliche Erklärung

Ich versichere hiermit an Eides statt, dass diese Arbeit von niemand anderem als meiner Person verfasst worden ist. Alle verwendeten Hilfsmittel wie Berichte, Bücher, Internetseiten oder ähnliches sind im Literaturverzeichnis angegeben, Zitate aus fremden Arbeiten sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungskommission vorgelegt und auch nicht veröfentlicht.

Berlin, den 11. November 2016

Johannes Mihan

Inhaltsverzeichnis

Kapitel 1

Einleitung

Das heutige Internet bietet einem Nutzer eine Vielzahl verschiedener Informationsquellen. Von Onlinenachrichtenzeitungen, kommerziellen Firmenauftritten, privaten Weblogs, Wissensdatenbanken wie Wikipedia und Sozialen Netzwerken ist alles vertreten. Das Internet umfasst mehr als 673 Millionen Webseiten (Stand November 2013). Die Masse an Informationen macht es dem Nutzer schwer, nach für ihn relevanten bzw. interessanten Themen zu filtern. Abhilfe schaffen sollen Newsaggregatoren, die eine komprimierte Version diverser Inhalte von verschiedenen Quellen liefern. Dem Nutzer soll so eine Übersicht über den angebotenen Inhalt der Quellen gegeben werden. Von dieser Art Nutzerprogramm existieren schon unzählige Versionen. Diese sind im Internet (meist) frei zugänglich. Dazu gehören unter anderem Googlenews[1], Newstral[2] oder 10000flies[3]. Diese bereiten Artikel nach Themen sortiert für den Nutzer auf und stellen ihm die Verlinkungen für genauere Informationen zur Verfügung.

1.1 Motivation

Eine Möglichkeit, für Nutzer zu bestimmten Webquellen aktuelle Artikel zu beziehen, ist das Abonnieren sogenannter RSS-Feeds (Rich Site Summary). Mit dem vermehrten Aufkommen von Weblogs im Internet, boten die Betreiber zunehmend einen Service für ihre Artikel an, die RSS-Feeds. Zu verstehen sind darunter kleine Zusammenfassungen von Artikeln, die dem Nutzer zur Verfügung gestellt werden. Dieser hat dann die Möglichkeit, mit einem einfachen Klicks den gesamten Artikel aufzurufen und/oder zu lesen. Es handelt sich hierbei um ein standardisiertes XML-Schema, das 1999 von „My Netscape Network“herausgegeben und von unterschiedlichen Firmen bzw. Entwicklergruppen weiterentwickelt wurde. Das RSS-Format findet schon mehr als 15 Jahre in Webseiten Verwendung und ist als Standardfunktion in Content-Managmentsystemen vorhanden. Da dieser Service als Funktion mitgeliefert wird gibt es viele potenzielle Quellen die vom kleinen Blogger bis großen Onlinezeitungen Verwendung finden.
Ein RSS-Feed ist ein XML-Datenformat, welches grundlegend immer aus einem Titel, Beschreibung und einem Link besteht. Diese drei Parameter sind die Grundvoraussetzung für ein RSS-Feed. Der Title beschreibt eine Überschrift für den Feed. Description gibt einen kurzen Text zu Inhalt des Feed wieder und der Link ist ein HTML-Link, der zu dem Inhalt des Anbieters führt. Der Inhalt des Anbieters ist in meisten Fällen ein HTML-Dokument einer Webseite, kann aber auch für andere Formate wie z.B. Bittorrents, Podcasts, Internetvideos oder Musik-

[1] https://news.google.de/ - Stand Juni 2016
[2] https://newstral.com/de - Stand Juni 2016
[3] http://www.10000flies.de - Stand November 2016

dateien erweitert werden. Bekannte RSS-Reader zum lesen, organisieren und abonnieren von RSS-Quellen sind Firefox-Thunderbird[4], Feedreader[5] oder Newzcrawler[6].
Nach dem Abonnieren mehrerer RSS-Quellen kann eine große für den Nutzer unübersichtliche Menge an Artikeln entstehen. Abhilfe können dabei Empfehlungssysteme schaffen. Diese bieten die Möglichkeiten Texte aufzubereiten und anhand von Profilen diese nach den Interessen des Nutzers sortieren. Empfehlungssysteme zählen zum Bereich des Maschinellen Lernens und verwenden Methoden des Information Retrievals. Sie erstellen, basierend auf einem erzeugtem Benutzerprofil, aus einer Menge von unübersichtlichen Objekten eine Teilmenge an Empfehlungen, die für den Nutzer von Interesse sind. Objekte können hier verschiedene Formen annehmen. So können es Produkte eines Webshops, Bücher, Filme oder Artikel sein. Ein bekanntes Beispiel für diese Art von Systemen ist der Online Musikstreamingdienst Last.fm [6]. Hier werden Empfehlungen durch die Metadaten wie Künstler, Musikgenre oder Ähnlichkeiten von Audiospuren erstellt.

1.2 Zielsetzung

Das Ziel dieser Bachelorarbeit umfasst die Umsetzung eines Empfehlungssystems mit Benutzermodell für einen RSS-Aggregators, der die Funktionen bietet, unterschiedliche RSS/ATOM-Quellen zu akquirieren, darzustellen und dem Nutzer zur Verfügung stellt. Der RSS-Aggregator ermöglicht dem Nutzer RSS-Quellen zu abonnieren, die enthaltenen Feeds zu lesen und diese nach dem Datum oder von einem Empfehlungssystem geordneten Liste anzuzeigen. Kernpunkt ist das Empfehlungssystem, das dem Nutzer neue RSS-Quellen empfiehlt und Feeds von abonnieren Quellen auflistet. Anhand der aufgestellten Anforderungen unter 1.2 werden die Techniken für das Empfehlungssystems und Benutzermodells erstellt.

1. **Kurze textuelle Daten**
   Bei RSS-Feeds handelt es sich um kurze Zusammenfassungen von Newsartikel einer Informationsquelle. Der Leser soll einen kurzen Anreiz zum Inhalt des Artikel bekommen und diesen gegebenenfalls auf der original Webseite komplett lesen können. Die Zusammenfassung beträgt im Durchschnitt rund 40 W'órter (ermittelt durch den mitgelieferten Datensatz).

2. **Der Nutzer möchte neues Entdecken**
   Der Nutzer möchte nicht nur weiterführende Artikel zu gelesen Themen informieren, sondern auch unterhalten werden. Er möchte neue, spannende Themen entdecken die nicht zwingend mit seine grundlegenden Interessen überstimmen müssen. [27].

3. **Personalisierte Empfehlungen ohne Aufwand**
   Der Nutzer möchte wie im vorherigen Punkt beschrieben neue Quellen entdecken, dieses darf aber nicht mit einem Mehraufwand verbunden sein. So sollte er seine Empfehlungen bekommen ohne viel Aufwand Daten an das System zu geben.

4. **Individuelle Quellen eintragen**
   Dem Nutzer kann selbst wählen welche Quellen er abonniert und diese auch erweitern

[4] https://support.mozilla.org/de/kb/nachrichten-feeds-und-blogs-abonnieren - Stand 2.11.2016
[5] http://www.rss-readers.org/windows/feedreader-de/ - Stand 2.11.2016
[6] http://www.newzcrawler.com/ - Stand 2.11.2016

können. Dies soll dem Nutzer Vertrauen in das System geben und für Kontrolle und Transparenz sorgen.

5. **Verschiedene Themenbereiche**
   Durch die Mögichkeit beliebige Quellen einzutragen, wird dem Nutzer ein breites Feld an verschiedene Themengebiete zur Auswahl gestellt. RSS-Quellen können thematisch von kleinen privaten Blogs mit speziellen vom Autor relevanten Themen über große etablierte Onlinemedien mit einer hohen Themenauswahl reichen.

6. **Änderung der Interessen**
   Die Interessen eines Nutzer im Bereich von Newsartikel kann sich von einen auf den anderen Tag ändern. Über einen gewissen Zeitraum kann bei einem Nutzer eine Übersättigung des Themas einsetzen. Fand er zu einem bestimmten Zeitpunkt die „Grüne Revolution“ noch interessant, kann sich dies auch wieder zu anderen Thema wie Rentenpolitik ändern. Ausgehend davon sollte dem Nutzer bei Interessenänderung nicht zu lange ältere Themen empfohlen werden. Somit ist eine weitere Anforderung an das Empfehlungssystem eine schnelle Reaktion auf ändernde Themenwechsel [22]

7. **zeitabhängig (Aktualität)**
   RSS-Feeds sind ein zeitabhängiges Medium. Private Blogs können abgeschaltet sein oder einzelne Artikel von Onlinenews-Zeitungen sind nicht mehr abrufbar oder aktuell. Außerdem folgen News einem zeitlichen Kontext. Es werden Artikel mit weiterführenden Themen veröffentlicht oder es liegen neue Informationen zu einem Sachverhalt vor über die der Nutzer informiert werden möchte. Daher muss innerhalb der Empfehlung von News auch dieses Problem beachtet werden.

1.3 Aufbau der Arbeit

Der Aufbau der Arbeit orientiert sich an den aufgestellten Anforderungen des RSS-Aggregators. Nach der Einleitung werden die grundsätzlichen Methodiken aus dem Information Retrieval (IR) vorgestellt die es ermöglichen textuelle Daten aufzubereiten. Durch IR-Systeme können aber nicht alle Anforderungspunkte wie “Empfehlungen ohne Aufwand“ gelöst werden. Ein weiteren Ansatz bieten die unter Kapitel 3 eingeführten Empfehlungssysteme. Dazu werden zuerst die traditionellen - kollaborative und inhaltsbasierte - Systeme mit ihren Vor- und Nachteilen vorgestellt. Anhand der Nachteile der klassischen Systeme wird auch auf die verschiedenen hybriden Empfehlungssysteme eingegangen. Durch Kombinieren der traditionellen Techniken wird versucht die Nachteile der traditionellen Systemen zu verbessern und den Empfehlungsqualität zu steigern. Ein weitere Grundlage damit das System Empfehlungen geben kann muss der Nutzer modelliert werden. Dazu wird Kapitel 4 auf die Modellierung des Benutzer eingegangen. Das Benutzermodell bestimmt welche Daten des Nutzer gesammelt werden und wie die für den Empfehlungsprozess verwendet werden. Die Art lässt sich dazu in zwei Arten einteilen. Zum einen das explizite Feedback, das dem Nutzer direkt nach Informationen fragt und das implizite Feedback das durch indirekt geschlossene Informationen durch die Benutzung des Programms sammelt. Es werden Vor- und Nachteile der einzelnen Aspekte zum impliziten und expliziten Feedback erläutert und anhand der oben genannten Punkte 1.2 beleuchtet. Nach dem die theoretischen Grundelemente mit ihren Vor- und Nachteilen behandelt wurden, wird in Kapitel 5 das Empfehlungssystem, sowie das Benutzermodell für den RSS-Aggregators theoretisch

entwickelt. Dazu wird auf jeden Anforderungspunkt unter 1.2 eingegangen und gelöst. Im ersten Teil des Letzten Kapitel wird eine Implementierung des RSS-Aggregators vorgestellt. Im zweiten Teil wird das Empfehlungssystem und Benutzermodell anhand einer Simulation offline evaluiert. Dazu wurden drei weitere Vergleichssysteme implementiert und mit dem neuen System in Bezug gesetzt. Abschließend wird das Kapitel mit einer Diskussion und einem Ausblick abgeschlossen.

Kapitel 2

Information Retrieval

Die voranschreitende Entwicklung in der Elektronik und der Computertechnologie in den letzten 30 Jahren hat zu einer massiven Zunahme an Informationen geführt. Die Möglichkeiten durch eine starke Vernetzung der Welt Informationen immer schneller zu verteilen und zu erhalten stellte die Technik vor neue Herrausforderungen. Mit der wachsenden Größe dieses Feldes wurden neue Methoden nötig, die es ermöglichten, neue Informationen effizient zu organisieren. Das Problem liegt nicht nur in der steigenden Anzahl der Informationen, sondern auch deren unterschiedliche Arten diese zu verarbeiten. Abbildung 2.1 vermittelt einen kurzen Überblick über die verschiedenen Arten von Objekten durch die Informationen gesammelt werden können.

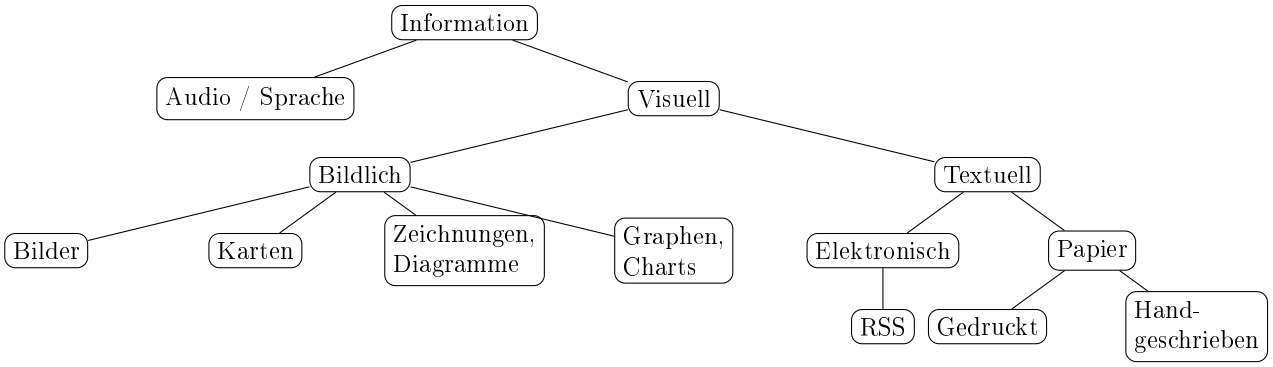

Abbildung 2.1: Verschiedene Formen von Informationen

Das Information Retrieval (folgend durch IR abgekürzt) umfasst das Forschungsgebiet, diese verschiedenen Informationsträger aufzubereiten und für den Menschen durchsuchbar zu machen. Ein bekanntes Anwendungsbeispiel für ein IR-System sind Suchmaschinen. Die Aufgabe besteht für die Suchmaschine darin für den Nutzer anhand seiner Suchanfrage bestimmte Dokumente zu liefern die ein bestimmtes Informationsbedürfniss erfüllten. Im Kontext des Internet handelt es sich um Webseiten und somit um textuelle Daten. Mittlerweile beschränkt sich die Suche im Internet nicht nur auf textuelle Daten sondern beinhaltet auch die Bilder-, Video oder Musiksuche [1]. Folgend wird die Arbeitsweise eines IR-Systems anhand von textuellen Daten erklärt. Abbildung 2.2 zeigt den grundlegenden Aufbau eines solchen IR-Systems.

2.1 Textuelle IR-Systeme

Grundlegend besteht die Aufgabe eines IR-Systems mit textueller Datenbasis darin zu einer Nutzeranfrage aus einer Sammlung von Dokumenten ähnliche Dokumente für den Nutzer zu

liefern. Anhand der textuellen Daten ergeben sich aber drei grundlegende Probleme für das System [2]:

1. Informationen innerhalb der Dokumente sind (meist) unstrukturiert
2. Texte sind (meist) in natürlicher Sprache geschrieben
3. Texte beinhalten einen weiten Themenbereich

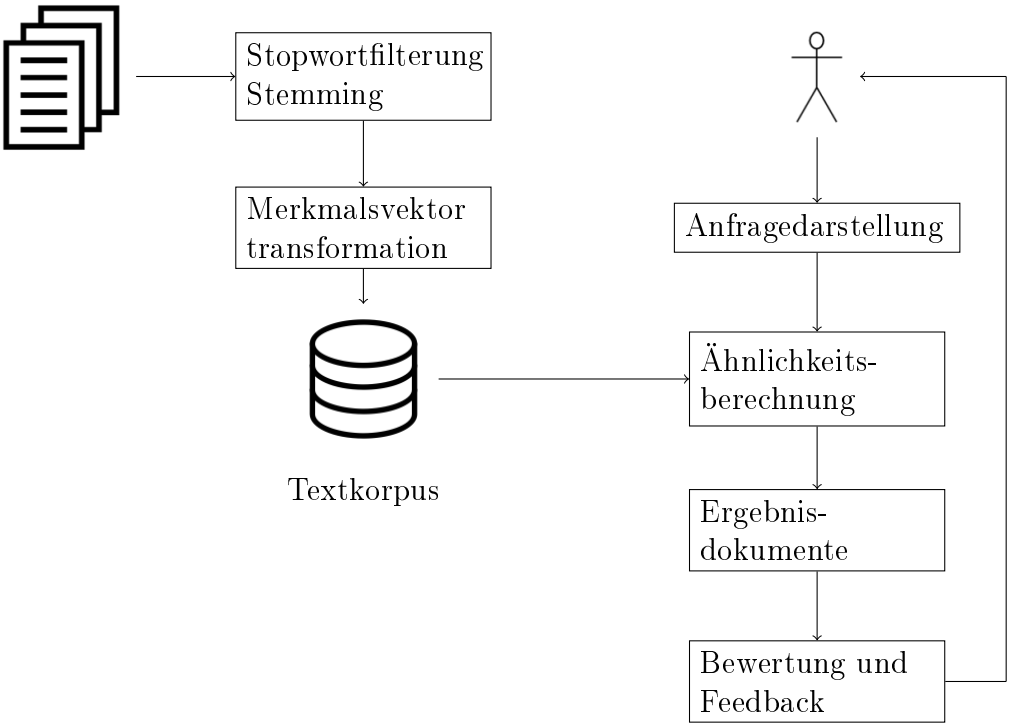

Abbildung 2.2: Ablauf eines textuellen IR-Systems

Ausgehend von diesen drei Problemen wandelt das IR-System die unstrukturierten Texte in eine strukturelle Form. Das IR-System indexiert die Dokumente, indem es die Dokumente in ihre *Schlüsselwörter* aufsplittet. Durch die Schlüsselwörter wird eine Dokument inhaltlich und thematisch charakterisiert Dazu werden zum einen die Texte von *Stopwörtern* bereinigt. Wörter gelten als *Stopwörtern* die bei der Indexierung nicht beachtet werden, da sie häufig vorkommen und keine Relevanz für den Inhalt des Textes besitzen. Dies können zum Beispiel für den deutschsprachigen Raum bestimmte und unbestimmte Artikel, sowie Konjunktionen, Präpositionen, Negationen und Satzzeichen sein. Ein weiterer Schritt zur Normalisierung der Schlüsselwörter ist das *Stemming*. Diese Methode reduziert verschiedene morphologische Varianten eines Wortes auf deren gemeinsamen Wortstamm. Zum Stemming gehört die Reduzierung von Deklination, Konjunktion und in einigen Sprachen die Wortzerlegung oder Zusammensetzung. Nachdem die Schlüsselwörter für die Dokumente erstellt wurde, werden diese als Merksmalsvektor formuliert 2.1.

$$f(d) = (f_1, f_2, ..., f_n) \tag{2.1}$$

wobei d ein Dokument darstellt und f_i ein Schlüsselwort aus dem Dokument beschreibt. Alle Merkmalsvektoren bilden dann zusammen den *Textkorpus*. Anhand dieses Vektors wird

der Inhalt des Dokuments charakterisiert. Alle Merkmale (spezielle Schlüsselwörter) aus allen Dokumenten bilden zusammen ein Vektorraummodell (VSM). Dieser besitzt als Basis die Dokumente und alle Vorkommenden Merkmalswörter. Stellt der Nutzer eine Anfrage an das System, wird diese ebenfalls in einen Merkmalsvektor transformiert und im VSM abgebildet. Durch Ähnlichkeitsmaß 2.3 bestimmt das System die ähnlichsten Dokumente und liefert diese dem Nutzer zurück.

2.2 Tf-Idf

Zur Verbesserung der Vektorraumpräsentation kann dem Merkmalsvektor auch eine Gewichtung zugeordnet werden. Ein Gewichtungsverfahren ist die Term Frequency - inverse document frequency (abgekürzt Tf-Idf). Term Frequency - inverse document frequency beschreibt ein statistisches Verfahren aus dem Textkorpus (IDF) und einem betrachtenden Text (TF) eine Gewichtung zu bestimmen. Das Verfahren beruht auf den zwei Annahmen zu Texten, das zum einem Terme die in Texten ein hohes Vorkommen (Term-Frequency) besitzen bedeutend sind. Die zweite Annahme ist, dass Wörter die oft in mehreren Texten vorkommen, wie Artikel oder Pronomen, für den Inhalt weniger bedeutsam (inverse Document Frequency) sind. Klassisch wird die Termfrequency über 2.2, nur anhand des Zählens der Vorkommen eines Termes t in einem Dokument d berechnet.

$$tf(t,d) = frequency(t,d) \tag{2.2}$$

Zum anderen können auch binäre Werte für das Vorhandensein eines Schlüsselwortes als Einträge oder eine logarithmische Normalisierung $1+log(freq(t,d))$ definiert werden. Dies hängt meist von den Designentscheidungen für das IR-System ab. Die inverse Document frequency wird durch 2.3 berechnet.

$$idf(t,D) = log(N|d \in D : t \in d|) \tag{2.3}$$

Unter diesen beiden Annahmen ordnet die Tf-Idf jedem Wort eine Gewichtung zu, wobei die wichtigsten Terme die mit der höhsten Gewichtung belegt werden. Grundlegend ist die tf-idf für ein Dokumentenkorpus D, ein Dokument $d \in D$ und einen Term t definiert durch:

$$tfidf(t,d,D) = tf(t,d) \cdot idf(t,D) \tag{2.4}$$

2.3 Ähnlichkeitsmaße

Ähnlichkeitsmaße definieren die Beziehung zwischen zwei Dokumenten. Das Maß wird durch eine Funktion *sim* beschrieben, die zwei Merkmalsvektoren eine reelle Zahl zuordnet. Anhand dieser Funktion kann über die Distanz eine Ähnlichkeit festgelegt werden. $sim : D \times D \rightarrow R$ Nachfolgend wird ein bekanntes Beispiel aus der Literatur vorgestellt [5].

Kosinus-Ähnlichkeitsmaß

Das Kosinus-Ähnlichkeitsmaß bestimmt anhand des Kosinus-Winkels zwischen zwei Merkmalsvektoren, ob diese in die selbe Richtung zeigen.

$$sim(\vec{v},\vec{w}) = \frac{\sum_{i=1}^{n} v_i \cdot w_i}{\sqrt{\sum_{i=1}^{n} v_i^2} \cdot \sqrt{\sum_{i=1}^{n} \cdot w_i^2}} \tag{2.5}$$

Werden die Vektoren als Mengen von Wörtern betrachtet, so gilt:

$$sim(v,w) = \frac{|v \cap w|}{|v| \cdot |w|} \tag{2.6}$$

Das Maß ordnet einen Wert zwischen 0 und 1 zu, wobei 0 unabhängig (Orthogonal) und 1 genau gleichgerichtet bedeutet. Im VSM wird dies mit den termgewichteten Vektoren bestimmt. Die Ähnlichkeit ist somit im positiven Bereichen und liegt zwischen 0 und 1. Die Berechnung des Winkels zwischen den beiden Merkmalsvektoren findet anhand des Standardskalarproduktes statt.

2.4 Zusammenfassung

IR-Systeme sind Systeme, die eine Datenmenge aufbereiten und einem Nutzer durch Anfragen ermöglicht Informationen zurück zuliefern. Sie bedienen sich verschiedener Verfahren wie dem Vektorraummodell bei textuellen Daten und eine Gewichtungm'oglichkeit durch die Tf-Idf um die Elemente aufzubereiten. Anhand dieser Aufbereitung können Elemente verglichen werden und Ähnlichkeiten zu Anfragen bestimmt werden. Durch diese Stärke liefern IR-Systeme gute Methoden zum Aufbereiten und Verarbeiten von textuellen Daten wie es unter den Punkten 1.2 als Anforderung formuliert ist. Somit bilden IR-Systeme eine gute Grundlage für die Verarbeitung von textuellen Daten. Ein reines IR-System würde sich für eine Suchfunktion der Feeds und RSS-Quellen eigenen. Durch die schnelllebige Welt der Nachrichten ist es für den Nutzer aber nicht praktikabel, unablässig neue Anfragen nach für ihn interessanten Themen zu stellen. Somit würde der Nutzer ein erheblichen Mehraufwand erfahren 1.2, wenn er für personalisierte Quellen und Feeds immer wieder ein Suchanfrage an das System stellen müsste. Als Erweiterung gelten die Empfehlungssysteme, die anhand einer Benutzermodells Informationen des Nutzers sammelt und daraus Anfragen an das IR-System generiert. Diese werden im nächsten Kapitel genauer betrachtet.

Kapitel 3

Empfehlungssysteme

Empfehlungssysteme entwickelten sich aus verschieden Forschungsbereichen wie Expertensystemen, Information Retrieval und Kognitionswissenschaften [6]. Empfehlungssysteme bündeln Informationen über die Vorlieben eines Nutzers und erstellen anhand dieser Informationen aus einer ungeordneten Menge an Entitäten eine personalisierte Empfehlung. Bei den Entitäten kann es sich um diverse Elemente wie Webseiten, Filme, Musiksongs, Zeitungsartikel, Reiseziele, Restaurants, E-Mails etc. handeln. Die Informationen zur Bestimmung der Benutzerinteressen können über explizites (normalerweise Informationen durch direktes bewerten eines Nutzers) oder implizites (durch Beobachtung von Verhalten des Nutzers wie Anklicken, Kaufen, Hören, Sehen) Feedback gewonnen werden. In einem Empfehlungssystem können auch weitere Einflussfaktoren, wie demographische Parameter (Geschlecht, Herkunftsland oder Alter) oder mit höherer Nutzerzahl in Sozialen Netzwerken weitere Parameter wie Social Tags, Follower oder Nutzerkommentare zur Steigerung der Empfehlungsqualität genutzt werden.

Formal lässt sich ein Empfehlungssystem nach Andre Klahold [8] definieren als die Maximierung des Nutzwertes einer Empfehlungsmenge T einer Entitätsmenge M für einen Benutzer B in einem Kontext K.

$$max(Nutzwert(B,K,T)) \tag{3.1}$$

Dieser Kontext K aus dem Benutzerprofil P, der Entitätsmenge M und weiteren kontextuellen Rahmenparametern der realen Welt wie Datum, Uhrzeit oder Geoinformationen zusammen (S). Durch die komplexen Einsatzgebiete wie Musik, Filme oder Zeitungsartikel besteht die Herausforderung in der Festlegung des Nutzwertes eines Entitätsobjektes für einen Nutzer. Andre Klahold definiert die Ähnlichkeit eines Benutzerprofils zu einem Kontext als Maß stab für die Nützlichkeit [8]. Welche Informationen gesammelt und zum Beschreiben eines Benutzerprofil verwendet werden, hängt vom Einsatzgebiet des Empfehlungssystems ab. So können Informationen beispielsweise über explizite Bewertungen wie das Einstellen eines Schiebereglers auf einer Skala von 1 bis 200 für einen Witz von Jester.com[1] [7] sein. Das Empfehlungssystem besitzt dann eine Trainingsmenge, mit der es ähnliche Entitäten vorhersagen kann. Im Fall von Jester.com bedeutet das, wenn der Nutzer mehrere Witze mit dem Thema Ärzte eine besonders hoch bewertet hat, dann ist anzunehmen, dass der Nutzer an neuen Ärztewitzen interessiert ist. Ob, wie in dem Fall eher nicht, vorher gesehene Entitäten wieder empfohlen werden hängt von der Gestaltung und den Rahmenbedingungen des Empfehlungssystems ab. So verfolgen die beiden klassischen Ansätze zwei verschiedene Arte von Informationsnutzung zur Erstellung von

[1] http://eigentaste.berkeley.edu/ Stand 20.04.2016

Empfehlungen. In diesem Kapitel werden die wichtigsten traditionellen Empfehlungssysteme mit ihren Funktionsweisen vorgestellt und ihre Vor- und Nachteile beleuchtet.

3.1 Traditionelle Empfehlungssysteme

Die Autoren von Recommender systems survey [9] beschreiben einen guten Überblick von traditionellen Empfehlungssystemen. Sie unterteilen diese in

- kollaborative Systeme : Systeme die mehrere Nutzer als Grundlage zur Empfehlungsbestimmung benutzen.
- inhaltsbasierte Systeme : Systeme die auf Grundlage der Eigenschaften der Empfehlungselemente Empfehlungen treffen
- hybride Systeme : System die die Vorteile aus kollaborativen und inhaltsbasierten Systemen verwenden.

Diese Konzepte verfolgen verschiedene bzw. gemischte Ansätze zur Bestimmung der Ähnlichkeiten zwischen den Entitätselementen.

3.1.1 Kollaborative Filtern

Kollaborative Systeme verfolgen den Ansatz, die Bestimmung der Empfehlungsmenge auf der Grundlage ähnlichen Nutzerinteressen zu bestimmen. Menschen diskutieren zum Beispiel an verschiedenen sozialen Orten über Bücher, die sie gelesen haben, Restaurants, die sie besucht haben oder Filme die sie gesehen haben. Diese Diskussionen dienen wiederum als Empfehlung für andere. So stellt eine Person die am liebsten Filme aus dem Horrorgenre mag, für einen Freund, der auch Filme aus diesem Genre favorisiert, einen Garant für potenziell gute Empfehlungen eines Horrofilmes dar.

Nach der Formel 3.1 wird der Nutzwert über die Ähnlichkeit von zwei Nutzern mit ähnlichen Benutzerprofilen zu einer Entitätsmenge bestimmt. Dazu werden die Bewertungen implizit oder explizit gesammelt, von einem Nutzer zu mehreren Entitäten mit einem Bewertungsvektors gespeichert. In Matrixdarstellung werden die Vektoren als *User-Item-Matrix* bezeichnet.

	The Matrix	Speed	Vergiss mein Nicht	The Ring	Der Exozist
Isabell	1	3	5		
Paul	3	3		5	5
Emmal	5	4		2	3
Claudia	2	1	4	1	1
Benjamin		1	5	1	1
Marko	4		1	4	5
...	...	...	...	...	...

Tabelle 3.1: Beispiel einer fiktiven Filmempfehlung User-Item Matrix

Zum Quantisieren einer subjektiven Meinung zu einem Entitätsobjekt können die Werte eines User-Item-Vektor aus skalaren oder binären Werten bestehen. Tabelle 3.1 zeigt als Beispiel eine User-Item-Matrix mit einem Skalar-Rating mit numerischen Werten von 1-5 eines

fiktiven Filmempfehlungssystems. Bei Empfehlungen werden für einen Nutzer Entitäten empfohlen mit ähnlichen User-Item-Vektoren. Beispiele für kollaborative System sind Tapestry [12], GroupLens[2] [13] oder Jester.com[3] [7]. Kollaboratives Filtern lässt sich nach [9] in zwei Typen, Memory-based und Model-based, kategorisieren. Beide Arten werden folgend kurz vorgestellt.

Memory-based
Diese Algorithmen sind heuristisch und generieren Bewertungsvorhersagen basierend auf allen bisher von Nutzer bewerteten Objekten. Dieser Ansatz erstellt die Empfehlungen direkt anhand der User-Item-Matrix und nutzt alle vorgehenden Bewertungen vor dem Empfehlungsprozess. Dadurch sind die Empfehlungen immer Aktuell, da die letzten Bewertungen zur Berechnung in den Prozess mit einbezogen werden. Zur Bestimmung der Relevanz eines Empfehlungselement i für einen Nutzer u wird eine Aggregationsfunktion über die Menge U aller n ähnlichsten Nutzer ausgeführt.

$$r(u,i) = aggr_{u' \in U}(r(u',i)) \tag{3.2}$$

Die Berechnung durch die Aggregationsfunktion kann, abhängig von der Designentscheidung von System zu System, unterschiedlich bestimmt werden. Als Beispiele für die Aggregationsfunktion dienen [6]:

$$r(u,i) = \frac{1}{N} \sum_{u' \in U} r(u',i) \tag{3.3}$$

$$r(u,i) = \frac{\sum_{j=1,n} S(i,i_j) \cdot R(i,u)}{\sum_{j=1,n} S(i,i_j)} \tag{3.4}$$

$$r(u,i) = r_u + k \cdot \sum_{u' \in U} sim(u,u') \cdot (r_{u'_i} - \bar{r}_{u'}) \tag{3.5}$$

wobei k ein normalisierter Koeffizient ist und üblich als $k = \frac{1}{\sum_{u' \in U} |sim(u,u')|}$ festgelegt wird. Die durchschnittliche Bewertung des Benutzers $ur(u)$ wird definiert als

$$r_u = \frac{1}{I_u} \sum_{i \in I_u} r_{u,i} \tag{3.6}$$

wobei I_u die Menge aller vom Nutzer u bewerteten Items darstellt. Im einfachsten Fall lässt sich die Aggregation durch eine *Durchschittsfunktion* 3.3 ermitteln. Der häufigste verwendete Ansatz ist aber durch die Berechnung einer *gewichteten Summe* 3.4 Das Ähnlichkeitsmaß $sim(u,u')$ zwischen den Nutzers u und u' ist im Grunde die Messung des Abstandes zwischen den Benutzern und dient hier als Gewicht. Je ähnlicher sich die Nutzer sind, desto größer wird das Gewicht von $r_{u',i}$ für die Vorhersage der Bewertung $r_{u,i}$. Ein Problem der gewichteten

[2] http://grouplens.org/ - Stand 10.2016
[3] http://eigentaste.berkeley.edu - Stand 20.8.2016

Summe besteht darin, dass unterschiedliche Benutzer die Bewertungsskala unterschiedlich interpretieren. Diesem Problem wird mit der *angepasste gewichtete Summe* 3.5 entgegengewirkt. Die angepassten gewichteten Summe nutzt nicht die absoluten Werte der Bewertungen, sondern summiert die Differenz von den Durchschnittsbewertungen der entsprechenden Nutzer. Zur Implementierung der Ähnlichkeitsfunktionen $sim(u,u')$ existieren unterschiedliche Ansätze in kollaborativen Empfehlungssystemen [6]. Die populärsten Ansätze sind der korrelationsbasierte (correlation-based) und der kosinusbasierte (cosinus-based):

$$sim(u,u') = \frac{\sum_{i \in I_{u,u'}} (r_{u,i} - \bar{r}_u)(r_{u',i} - \bar{r}_{u'})}{\sqrt{\sum_{i \in I_{u,u'}} (r_{u,i} - \bar{r}_u)^2) \times \sum_{i \in I_{u,u'}} (r_{u',i} - \bar{r}_{u'})^2}} \tag{3.7}$$

$$sim(u,u') = \frac{\sum_{i \in I_{u,u'}} r_{u,i} \cdot r_{u',i}}{\sqrt{\sum_{i \in I_{u,u'}} r_{u,i}^2} \cdot \sqrt{\sum_{i \in I_{u,u'}} r_{u',i}^2}} \tag{3.8}$$

wobei $I(u,u')$ die Schnittmenge der beiden Nutzer u und u' bewerteten Items darstellt. Im korrelationsbasierten Ansatz wird dazu die *Pearson-correlation-coefficient* 3.7 zum Messen der Ähnlichkeit verwendet. Im kosinusbasierten Ansatz 3.8 werden die Benutzer u und u' als Vektoren in einem m-dimensionalen Raum interpretiert , wobei $m = |I_{u,u'}|$. Die Ähnlichkeit ergibt sich dann aus dem Wert des Kosinuswinkels dieser beiden Vektoren.

Model-based
Mit zunehmender Größe der User-Item-Matrix kann es zu Perfomenceproblemen mit dem Memory-based Ansatz kommen, da direktes Berechnen auf der Matrix nicht mehr praktikabel ist. Als Alternative werden Model-based Methoden verwendet. Model-based Methoden verwenden Algorithmen aus den Bereichen von Data-Mining und Maschinellen Lernen. Darunter fallen Algorithmen wie *Bayesian-Klassifikatoren*, *Neurale Netzwerke*, *Fuzzysysteme*, *Generische Algorithmen* oder *Matrixfaktorisierung* [6]. Der Hauptbestandteil der Methoden arbeitet mit der Klassifizierung oder Clustertechniken des Nutzers anhand eines Trainingsets. Der Vorteil liegt in der besseren Handhabung von sparsity als durch *memory-based* Verfahren. Als Nachteil benötigt das Erstellen des Modelles eine lange Designzeit. Dazu muss ein Kompromiss zwischen der Vorhersageleistung und der Skalierbarkeit gefunden werden.

Vorteile

- **Cross-Genre**
  Kollaborative Filtern hat den Vorteil, dass es flexibel auf das Nutzerverhalten reagieren kann. Durch eine hohe Anzahl an Nutzern ermöglicht es eine Cross-Genre-Empfehlung. Somit kann der Nutzer neue Empfehlungen erhalten, die nicht allein durch den Inhalt der Empfehlungselemente ermittelt werden kann [2],[11].

Nachteile

- **New Item**
  beschreibt die Schwierigkeit des kollaborativen System, neue Entitäten zu empfehlen. Neue Elemente müssen erst durch Nutzer bewertet werden, damit das Empfehlungssystem diese mit in den Empfehlungsprozess einbeziehen kann. Auf Grundlage dessen muss

das System erst einen Schwellwert an Nutzern erreichen, um sinnvolle Empfehlungen zu geben. In hybriden Systemen wird das durch die Ähnlichkeit der Eigenschaften der Entität abfangen.

- **New User**
  beschreibt das gleichartige Problem wie bei New-Item-Problem nur mit Nutzer und nicht mit Entitäten. Damit Empfehlungen für einen Nutzer getroffen werden können, muss erst eine hinreichende Menge an interessanten Elementen für den Nutzer bestimmt werden. So kann das System erst nach mehreren Bewertungen des Nutzer Elemente empfehlen. Zusammen mit dem New Item ergibt dies das Sold-Start Problem. Dies beschreibt die Schwierigkeit, erst Elemente empfehlen zu können, wenn eine Datenmenge mit ausreichend Informationen gesammelt wurde.

- **Sparsity**
  entsteht, wenn durch eine große Menge an Entitätselementen die Nutzer nicht ausreichend Elemente bewertet haben. Durch spärliches Bewerten der Nutzer entstehen in der User-Item-Matrix Leereinträge. Diese können die Qualität der Empfehlungen stark reduzieren, da Entitäten durch zu große Spärlichkeit nicht mehr aufgefunden werden. Nach [6] nimmt die Empfehlungsgüte des kollaboratives Filterns drastisch ab, wenn im Mittel die Nutzer 99,5% der Entitäten nicht bewertet haben.

3.1.2 Inhaltsbasiertes Filtern

Inhaltsbasierte Empfehlungssysteme verfolgen den Ansatz, dem Nutzer auf Grundlage von den Merkmalen der Objekte neue, ähnliche Objekte zu empfehlen. Dazu bedienen diese sich der Methoden aus den Forschungsgebieten des Information Retrievals und der Künstlichen Intelligenz [24]. Basierend auf der positiven bzw. negativen Bewertung des Benutzer von Objekten, bestimmt das System über ein Ähnlichkeitsmaßes die Empfehlungen. Durch eine Eigenschaftsanalyse (Feature Selection) können die charakteristischen Eigenschaften der Elemente ermittelt. Beispielhaft für Merkmale eines Objektes können bei Filmen die Beschreibung der Handlung, der Regisseur oder die darstellenden Schauspieler sein. Bei Musik die Liedertexte oder das Audiosignal herangezogen werden. Methoden aus dem Information Retrieval wie unter 2.1 beschrieben können für textuelle Daten Anwendung finden. Die Schwierigkeit besteht im Designprozess darin ausreichend beschreibbare Merkmal der Objekte zu formalisieren. Unter Umständen können solche Informationen auch nicht existieren oder die Beschaffung zu Kostenintensiv sein.

Arbeitsweise

Grundlegend wird ein Inhaltsbasiertes Empfehlungssystem in drei Komponenten *Content Analyzer*, *Profil Learner* und *Filtering Component* unterteilt [24], die die Aufbereitung der Objekte vornimmt, das Benutzerprofil erzeugt und zum Schluss relevante Objekte empfehlen.

Content Analyzer
Diese Komponente übernimmt den ersten Vorverarbeitungsschritt für unstrukturierte Informationen, wie zum Beispiel Texte. Der Content Analyzer extrahiert Merkmale der Objekte und bereitet diese f'úr die nächsten Schritte auf. Bei textuellen Daten werden dazu Methoden unter anderem des Information Retrievals verwendet die im vorherigen Kapitel 2 genauer beschrieben wurden.

Profile Learner
In diesem Modul werden Informationen zum Benutzer gesammelt, analysiert und ein Benutzerprofil erstellt. Dazu werden Techniken des maschinellen Lernens [10] angewendet. Das Ziel ist es aus vorher bewerteten Objekten die Relevanz für weitere Objekte für einen Nutzer schätzen.

Filtering Component Relevante Objekte werden anhand des Benutzerprofils vorgeschlagen. Hierfür wird ein Ähnlichkeitsmaß 2.3 auf dem Objektraum definiert. Für Texte (nach dem Preprocessing) kann das Kosinus-Ähnlichkeitsmaß 2.3 verwendet um die Dokumenten zu vergleichen. Dadurch lässt sich eine Reihenfolge der ähnlichsten Dokumente erstellen, von denen dann die k ähnlichsten Dokumente empfohlen werden. Tabelle 3.2 zeigt eine Schematische Darstellung des Objektraumes nach dem Preprocessing.

	doc_1	doc_2	...	doc_{n-1}	doc_n
$term_1$	tf_{11}	tf_{12}	...	$tf_{1,n-1}$	$tf_{1,n}$
$term_2$	$tf_{2,1}$	$tf_{2,2}$	...	$tf_{2,n-1}$	$tf_{2,n}$
$\vdots$	$\vdots$	$\vdots$	...	$\vdots$	$\vdots$
$term_{n-1}$	$tf_{n1,1}$	$tf_{n-1,2}$	...	$tf_{n-1,n-1}$	$tf_{n-1,n}$
$term_n$	$tf_{n,1}$	$tf_{n,2}$	...	$tf_{n,n-1}$	$tf_{n,n}$

Tabelle 3.2: Schemadarstellung eines VSM nach dem Preprocessing

Vorteile

- **Nutzerunabhängigkeit**
  Im Gegensatz zu kollaborativen Systemen basieren Inhaltsbasierte Empfehlungssysteme direkt auf den Empfehlungselementen. Folglich sind die Empfehlungen Unabhängig von anderen Nutzern.

- **Transparenz**
  Der Vergleich durch das Ähnlichkeitsmaß liefert eine Erklĭrung für den Nutzer wie die Empfehlungen getroffen wurde. Klassische kollaborativ arbeiten Systeme sind eine Blackbox, die die Interessen der Nutzer nicht erklären kann. Auf Grundlage der Ähnlichkeitsmaße kann das inhaltsbasierte System dem Nutzer auflisten anhand, welcher Eigenschaften die Elemente empfohlen werden. Dies schafft ein Vertrauensbasis für den Nutzer in das System [24].

- **New Item**
  Da inhaltsbasiertes Filtern direkt auf den Eigenschaften der Elemente arbeitet werden, anders als bei kollaborativen Systemen, neue Elemente direkt in den Empfehlungsprozess mit einbezogen. Die Elemente müssen nicht erst bewertet werden, bevor diese mit einbezogen werden.

Nachteile

- **Limited-content-analysis**
  Inhaltsbasierte Systeme können nur eine begrenzte Inhaltsanalyse geben. Dieses System ist durch seine Anzahl an Merkmalen der Elemente beschränkt und auß dem abhängig

von der Bewertung durch den Designer welche Informationen genutzt werden sollen. Auch kann es sein, dass nicht genug Informationen für eine genaue Beschreibung des Benutzer vorliegen. Somit ist das Empfehlungssystem in der Beschreibung des Nutzers beschränkt.

- **Overspecialization**
  Ein weiterer Nachteil besteht in der Überspezialisierung. Wenn sich das Benutzerprofil stark eine Präferenz entwickelt, wird dem Nutzer nur noch eine Art an Elementen empfohlen. Dadurch haben Elemente aus anderen Kategorien keine Möglichkeit mehr betrachtet zu werden [11].

- **New User**
  Auch bei neuen Benutzern hat dieser Ansatz seine Schwachstelle. Das System muss erst Daten über den Nutzer sammeln, um gute Empfehlungen zu treffen.

3.1.3 Hybride Systeme

Bei hybriden Empfehlungssystem handelt es sich um eine Kombination der vorher genannten Verfahren. Das System kombiniert mehrere Techniken und versucht mit den Stärken des einen Empfehlungssystem die Schwächen des anderen Empfehlungssystems zu beheben. Beispielsweise leidet der kollaborative Ansatz unter dem New Item Problem und kann keine Objekte empfehlen, die noch nicht von anderen Benutzern bewertet wurden. Häufig liegen jedoch Beschreibungen für die zu empfehlenden Objekte vor. Das bedeutet, mit einem inhaltsbasierten Empfehlungssystem könnte dieses Problem behoben werden [11]. Die Tabelle 3.3 listet die Vor- und Nachteile der traditionellen Systeme noch einmal auf.

System	Vorteile	Nachteile
Kollaborativ	Cross-Genre	New Item Sparsity New User
Inhaltsbasiert	Nutzerunabhängigkeit New Item	Limited-content-analyse Overspecialization New User

Tabelle 3.3: Überblick der Vor- und Nachteile von kollaborativen und inhaltsbasierten Systemen

Es existieren verschiedene Ansätze, um die klassischen Verfahren zu kombinieren und daraus ein hybrides System zu erstellen. In der Literatur wird zwischen folgenden Methoden unterschieden [14].

Weighted
Die in dem hybriden System verwendeten Ansätzen arbeiten unabhängig von einander und erstellen jeweils eine Liste von Empfehlungen. Die Vorhersage der verschiedenen Empfehlungssystem werden dann anhand einer Gewichtung zu einem Endergebnis zusammengefügt. Die Gewichte durch Versuche zu bestimmen, damit liegt die eigentliche Herausforderung bei diesem Ansatz.

Switching
Das System nutzt eine Bedingung oder ein Ereignis, um zwischen den verschiedenen Ansätzen zu wechseln. Der ausgewählte Ansatz übernimmt dann die Erstellung von Empfehlungen. Einfachste Methode wäre hierbei das zufällige Auswählen eines der Systeme.

Mixed
Die verwendeten Systeme erstellen unabhängig voneinander ihre Empfehlungen. Die von allen Systemen vorgeschlagenen Objekte werden dann gemeinsam in das Endergebnis übernommen und präsentiert. Diese Methode ist dann von Vorteil, wenn eine groß e Anzahl von vorgeschlagenen Objekten gefordert ist.

Feature Combination
Bei dieser Methode werden Merkmale aus den jeweilig anderen Systemen abgeleitet und als zusätzliche Eingabe für das anderen Empfehlungssystem verwendet. Zum Beispiel können, anhand von kollaborativen Informationen Klassen von Benutzern erkannt werden, die dann als Merkmal für ein inhaltsbasiertes Empfehlungssystem verwendet werden.

Cascade
Im Gegensatz zu den vorher angesprochenen Methoden verwendet die Cascade einen schrittweisen Ablauf. Zunächst wird mit einem der Systeme eine grobe Vorhersage getroffen. Das Ergebnisse dieser Vorhersage wird dann in einem nächsten Schritt als Eingabe für eines der anderen Verfahren übergeben und verfeinert.

Feature Augmentation
Bei dieser Technik werden die Objekte mit Hilfe des ersten Empfehlungssystems in Klassen eingeteilt oder bewertet. Diese Informationen über die Objekte als Ausgabe des ersten Empfehlers werden dann als neue Merkmale der Objekte verwendet und damit als Eingabe den nächsten Empfehler verwendet.

Meta Level
Bei dieser Technik wird das gelernte Model als Eingabe für das andere System verwendet. Im Gegensatz zu Feature Augmentation wird nicht die Ausgabe des ersten Empfehlers als neues Merkmal verwendet, sondern es wird das Model übergeben, dass die neuen Merkmale erzeugt hat.

Durch hybride Systeme können einige der Nachteile der einzelnen Verfahren abgeschwächt werden. Im Idealfall werden nur die positiven Eigenschaften der Grundsysteme übernommen und die Nachteile vollständig eliminiert. Häufig wird als Grundlage ein kollaboratives System verwendet und mit anderen Verfahren kombiniert, um das New Item- oder New-User Problem zu lösen. Zum Beispiel kann ein weiteres inhaltsbasiertes System eingesetzt werden, um neue Objekte zu empfehlen, die noch nicht von Benutzern bewertet wurden [11].

3.2 Zusammenfassung

In diesem Kapitel wurden die klassischen Empfehlungssysteme mit ihren Vor- und Nachteilen vorgestellt. Ein kollaboratives System ermöglicht Empfehlungen anhand anderer Nutzer zu treffen. Durch die Stärke der Cross-Genre-Empfehlung besitzt ein kollaboratives System die Möglichkeit den Nutzer neue Quellen zu entdecken 1.2. Durch das New-Item-Problem besitzt es aber den Nachteil schwer mit neuen Empfehlungselementen umzugehen. Ein inhaltsbasiertes System kann mit dem New-Item-Problem anhand der Eigenschaften der Empfehlungselemente umgehen und dieses lösen, birgt aber den Nachteil durch die Overspecialization und Limit-Content-Analysis. Diese beiden Probleme stehen mit der Anforderung, dass der Nutzer neues entdecken möchte im Konflikt. Wiederum kann ein kollaboratives System durch die Cross-Genre-Empfehlung der Overspecialization und Limit-Content-Analysis entgegenwirken.

Die Hybriden Systeme versuchen die beiden traditionellen Systeme miteinander zu verbinden um die jeweils die Schwächen des anderen abzudecken. Dazu wurden mehrere Ansätze wie die Cascade vorgestellt, die eine schrittweise Kombination der beiden traditionellen Systeme verwendet. Trotz der Verbindung muss bei beiden traditionellen Systemen weiterhin mit dem New-User-Problem umgegangen werden.

Kapitel 4

Benutzermodellierung

Das Ziel der Benutzermodellierung besteht in der Adaption von interaktiven Anwendungen an die Bedürfnisse eines Nutzers. So beschreibt es den Prozess, um aus den gesammelten Daten über den Nutzer ein hochwertiges Benutzermodell abzuleiten. Diese können anhand expliziter Erfassung, mit Hilfe von Fragebögen oder implizit durch die Beobachtung des Nutzerverhaltens erfolgen. Anschließend werden die Daten verarbeitet, um ein Benutzermodell aufzubauen und anhand die Anwendung zu personalisieren. Anwendungsbereiche sind u.a. Empfehlungssysteme, adaptive Hypermedia-Systeme, rechnergestütztes Lernen, adaptive Benutzerschnittstellen oder mobile Guide-Anwendungen [14]. Durch die Wahl eines geeigneten Benutzermodells sollen der Anforderungspunkt 3, für personalisierte Empfehlungen ohne zusätzlichen Aufwand, Punkt 4 für die individuellen Quellen, 6 Änderungen der Interessen und Punkt 7 die zeitliche Abhängigkeit der Feeds behandelt werden. Dazu werden die Vor- und Nachteile von expliziten und impliziten Feedback zur Dateneingabe des Nutzers zum System herausgearbeitet und die Arten zur Verarbeitung von zeitlichem Kontext in einem Benutzermodell.

4.1 Benutzermodell

Ein Benutzermodell, technisch auch als Benutzerprofil bezeichnet, beschreibt eine Datenstruktur eines Benutzers U zu einem bestimmten Zeitpunkt t. Daten eines Nutzermodells können zum einem aus expliziten Eingaben des Nutzer oder durch das Sammeln von impliziten Eigenschaften durch das Verhalten des Nutzers gesammelt werden. Daten innerhalb eines Benutzermodells können sein [15] [16]:

- Persönlich Daten wie Alter, Adresse, Geburtstag, Email, Geschlecht Beruf, Ausbildung etc.
- Kontakt und Freunde wie Telefonnummern, Buddy-Listen, Chat-Listen, Gruppenzugehörigkeit etc.
- Zugriffsdaten wie Dienst, die abonniert wurden, Einstellungsparameter Customization data etc.
- Gerätedaten wie Displayparameter, Verbindungsqualität und Bandbreite, vorhandene Software und Tools etc.
- Ortsdaten wie Position, Bewegungsrichtung, Geschwindigkeit, Transportmittel etc.

- Präferenzen wie Einstellungsparameter jedes Endgerätes eines Nutzers, Informationsbedarf, Interessen, Informationsdienste

Die Aufnahme der Informationen hängt immer auch vom dem Anwendungsgebiet des Systems ab. Als Datenstruktur für das Benutzermodell kann aus einfachen Attribut-Wert-Paaren, Wahrscheinlichkeitswerten, Regeln und Regelmengen, komplexeren Relationen oder Datenstrukturen wie semantischen Netzen bestehen. Anhand der Methodik zum Verarbeiten der Daten wird zwischen statischen Benutzermodellen, deren Inhalt sich eher selten ändert und dynamischen Benutzermodellen, die schnell auf Veränderungen des Benutzerkontextes reagieren, unterschieden. Zum Umsetzen eines Benutzermodells können mehrere Methoden verwendet werden. Darunter zählen statische Verfahren wie Bayes'sche-Netzwerke, Methoden des Data-Minings oder Entscheidungsbäume [24]. Ein weiterer Ansatz ist die Modellierung von Stereotypen, die bestimmte Benutzertypen vorher beschreibt und anhand von vorher festgelegten Auslösern den Benutzer zu einer bestimmten Gruppen (wie Anfänger oder Experte) zuordnet [23].

4.1.1 Explizites Feedback

Explizites Feedback ermittelt Daten für ein Benutzermodell durch direktes Nachfragen des Nutzers. Dies kann zum einen durch direktes Eingaben der Daten durch ein Formular erfolgen oder durch Nachfragen des Systems. Als Beispiel für explizites Feedback gilt das direkte Ranking des Nutzers durch eine Beliebtheitsskala [16]. Diese kann negative Werte annehmen oder positiv sein. Andere Möglichkeiten bestehen in der Zuordnung des Nutzers zu bestimmten Kategorien, zum Beispiel kann dies ein Lieblingssong innerhalb eines Benutzermodelles für Musik sein. Anhand dieser aufgenommen Daten erstellt das Modell Präferenzen die im Benutzermodell aufgenommen werden.

Vorteile

- **Polarity**
  Der Nutzer hat direkt die Möglichkeit dem System positives und negatives Feedback zu geben. So hat er direkte Kontrolle dem System mitzuteilen was er mag und was nicht [18]. Dies kann über ein numerische Skala oder einem boolischen Wert wie ja/nein entsprechen.

- **Keine Domain Relevanz**
  Bei expliziten Feedback muss nicht weiter auf die Interpretation des Anwendungsgebiet eingegangen werden. Vom User eingegebene Daten können direkt als negativ oder positiv bewerte werden und müssen nicht interpretiert werden [18].

- **Transparenz**
  Durch das direkte Einwirken des Nutzers auf die verwendete Datenbasis des Empfehlungssystems gibt, ist es dem Nutzer möglich einen Einblick in die Entscheidungsfindung des System zu haben. Somit kann er direkten Fehlentscheidungen gegebenenfalls entgegenwirken [18].

Nachteile

- **Sparsity**
  Ein Nutzer muss erst einen großen Teil an Objekten bewerten. Somit kann es vorkommen, dass durch viele nicht bewertete Objekte, ähnlich des kollaborative Ansatzes, lücken entstehen [18].

- **Voreingenommenheit**
  Bei einer zu starken Abweichung von abgegeben Bewertung zwischen dem Nutzer und der Community kann bei einem Teil der Nutzer zur Voreingenommenheit zu dem Objekt führen. Dies stellte Harper [19] bei seiner Studie zu MovieLens fest. Ein anders Beispiel ist der Kritische Blick zu einem häufigen gut bewerteten zu Onlineprodukten.

- **Transparenz**
  Durch die Transparenz ist es möglich das Empfehlungssystem zu manipulieren. Da direkte Eingaben durch den Nutzer erkennbar sind, kann dieser genau die Veränderungen feststellen und das System versuchen zu Beeinflussen. Besonders bei kollaborativen System, kann das Einfluss auf andere Nutzer haben [18].

4.1.2 Implizites Feedback

Implizites Feedback ermittelt Daten für ein Benutzermodell indirekt durch das Verhalten eines Nutzer. Durch Aktionen innerhalb der Anwendung werden indirekt die Präferenzen des Nutzers ermittelt. Tabelle 4.1 klassifiziert grundlegende Aktionen eines Nutzers die für implizites Feedback verwendet werden können. Durch diese Nutzeraktionen können Interessen eines Nutzer durch ein Segment, an Objekten oder durch Klassifizierung implizit geschlossen werden.

Verhalten	Segment	Objekt	Klasse
Ausführen	ansehen auflisten scrollen suchen finden	auswählen	durchsuchen
Aufbewahren	Drucken	Bookmarken speichern löschen kaufen Email	abbonieren
Referenzieren	kopieren und einfügen	verlinken antworten weiterleiten zitieren	
Beschriften	markieren	veröffentlichen	organisieren
Erstellen	editieren schreiben	Author	

Tabelle 4.1: Einteilung von impliziten Nutzeraktionen in einer Anwendung nach [1]

Als Beispiel kann die Aktion eines Nutzer durch Anklicken eines Newsartikels als positives Interesse gewertet werden. Oder wenn der Nutzer bestimmte Artikel in einem Onlineshop kauft kann geschlossen werden, das der Nutzer ein starkes Interesse an dem Produkt hat. Negatives Feedback wird beispielsweise implizit durch die Tätigkeit des Löschen abgebildet oder durch die Kombination von einer Auflistung von Elementen und dem Überscrollen des Nutzers. Durch die Annahme, dass Suchergebnisse nach der Priorität zu der Suchanfrage dem Nutzer gelistet werden, kann implizit geschlossen werden, dass der Nutzer das Suchergebnis nicht als relevant

zu seiner Suchanfrage empfindet, wenn er das Ergebnis überscrollt [39]. Auch das Organisieren von Daten ermöglicht die Annahme, dass der Nutzer ein Interesse an der Kategorisierung zu einem bestimmten Thema hat [35].

Vorteile

- **Nutzerreichweite**
  Anders als bei explizitem Feedback gelten die Regeln zum Sammeln für implizite Daten für alle Nutzer eines Systems. Somit entstehen weniger Leerstellen innerhalb der bewerteten Objekte.

- **Fehlende Transparenz**
  Anders als bei expliziten Feedback hat der Nutzer keine Möglichkeit direkt auf das Feedback einzuwirken. Somit ist das System sicherer gegenüber potenzieller Beeinflussung und Manipulation durch andere Nutzer.

- **Hohe Datengenerierung**
  Das implizite Feedback ermöglicht es ohne viel Beanspruchung des Nutzers Daten zu sammeln.

Nachteile

- **Sensibilität gegenüber Rauschen**
  Implizites Feedback ist anfällig für Rauschen innerhalb der gesammelten Daten. Das Rauschen kann durch den Nutzer, beispielsweise durch falsches Klicken eines Item, durch das gewählte Model zum bestimmen der Nutzerpräferenzen oder durch die Tools des System, beispielsweise Zeitmessung zum Lesen eines Artikels.

- **Einseitiger Feedback**
  Ein Empfehlungssystem kann mit impliziten Feedback nur feststellen was der Nutzer vielleicht mögen könnte. Daher kann nur positives Feedback gesammelt werden. Beispielsweise kann in einem Musikempfehlungssystem angenommen werden, dass der Nutzer durch die Anzahl des Hörens eines Songs diesen vielleicht mag, aber unmöglich feststellen das er einen Song nicht mag wenn er diesen nicht hört.

- **Domain Relevanz**
  Das Domainwissen, daher das Wissen über den Anwendungsbereich ist bei impliziten Feedback relevant um die Daten des Feedbacks zu interpretieren.

Vergessensfunktion

Der Anforderungspunkt 7 aus 1.2 stellt als Bedingung an den RSS-Aggregator, dass es sich bei den RSS-Feeds um ein zeitabhängiges Medium handelt. Zum einem steht die Welt nicht still und es passieren immer wieder neue Ereignisse über die Berichtet wird. Ob es nun das Leben einer privaten Person oder einer Weltnachrichtenzeitung ist. Zum anderen können sich die Interessen des Nutzers über die Zeit ändern (Anforderungspunkt 6 1.2). Lathia zeigte in [], das nicht Kontextbezogene Algorithmen schlechter auf einer iterativen Datenbasis zur Vorhersage arbeiten, wenn ausschließslich vergangene Bewertungen für die Vorhersage von neue Empfehlungen verwendet werden. Dem Benutzerprofil muss durch den zeitlichen Kontext von News

für die Aktualität des Nutzerprofil darauf achten, dass die Daten zur Bestimmung der Empfehlungen sich in einem aktuellen Zeitraum befinden. Bei expliziten Feedback kann der Nutzer direkt auf die Objekte einwirken und negatives Feedback geben. Bei impliziten Feedback ist dies durch den Nachteil des einseitigen Feedbacks nicht möglich. Zu Verbesserung des Benutzerprofils müssen alte Daten wieder Vergessen werden. Das Benutzerprofil muss somit auch gesammelte Feedbackelemente wieder löschen können. Die bereinigt die Daten und ermöglicht eine klarere aktuelle Abbildung des Nutzers. Wenn ein Nutzer konstant zu einem Thema interessiert ist, wird dieser auch weiterhin Artikel zu diesem Thema lesen und diese wieder in das Benutzerprofil aufgenommen.

Wenn der Nutzer aber sein Interesse durch den schnellen Wandel von Nachrichten mit Aktualisierungen und neuen Ereignissen ändert muss das Benutzerprofil darauf reagieren. Damit ältere Artikel auch wieder aus dem Benutzerprofil herausfallen müssen diese aus dem Profil gelöscht werden.

Zur Betrachtung von zeitlichen Kontexten in Empfehlungssystemen liefert der Artikel [36] einen Überblick der verschiedenen Herangehensweisen. Zum einen die *Heuristischen-* und *Modellbasierten* Ansätze. Diese unterteilen sich wieder in *continious-time-aware*, *categorial-time-aware* und *time-adaptive*.

- **Continious-time-aware**
  Bei dieser Ausprägung wird die Information der Zeit als eine kontinuierliche Variable definiert. Der Grundgedanke von Systemen die diesen Ansatz verwenden ist die Vergabe von unterschiedlichen Gewichten zu Objekten in der Abhängigkeit ihres Alters. Ding und Li beschreiben in [36] als eine Lösung eine Exponentialfunktion zur Gewichtung von alten und neuen Bewertungen.

- **Categorial-time-aware**
  Im Categorial-time-aware Ansatz wird die Zeit als eine diskrete Variable angesehen. So kann die Zeit T diskret zum Beispiel als $T = \{week,weekend\}$ definiert werden und ermöglicht so eine unterschiedliche Gewichtung in der Bestimmung der Empfehlungselemente.

- **Time-adaptive**
  In diesem Ansatz werden die Parameter oder Daten für eine Berechnungsfunktion dynamisch über einen bestimmten Zeitraum justiert. Anders als bei Continous- und Categorial-time-aware wird die Rating-Rate nicht Ziel des Zeitkontextes gesetzt. So wird angenommen, dass ältere Vorlieben des Nutzer weniger für aktuelle Empfehlungen von Bedeutung für den Empfehlungsprozess sind.

4.2 Zusammenfassung

Durch die Einführung von Benutzermodellen ist es einem Empfehlungssystem möglich Interessen eines Nutzers abzubilden und auf Grundlage derer Empfehlungen für den Benutzer zu erstellen. So wird dem Nutzer eine umständliche andauernde Anfrage an das System abgenommen. Die größte Problematik des Benutzermodells besteht in der richtigen Abbildung des Benutzers, sowie die Wahl zum Erfassen des Feedbacks. Das explizite Feedback ermöglicht durch den Vorteil der Polarity direkt Einfluss auf das Benutzerprofil zunehmen. Der Nutzer hat außderm durch den Vorteil der Transparenz jederzeit Einblick in die Element, die ihm empfohlen

werden. Mit expliziten Feedback kann die Anforderung 2 1.2 mit individueller Quellenwahl gelöst werden. Sollte sich eine Quelle für den Nutzer vom veröffentlichten Inhalt in eine Richtung entwickeln die ihm nicht mehr gefällt kann direkt Einfluss darauf genommen werden. Somit kann mit explizitem Feedback auch der Punkt vier zur Änderung der Interessen aus den Anforderungen abgedeckt werden. Das das explizite Feedback kein Domainwissen benötigt, kann vom Vorteil sein, das der RSS-Aggregators durch individuelle Quellenwahl verschiedene Themenbereiche abdeckt. Zu den Nachteilen für explizites Feedback zählt die Sparsity. Durch die Sparsity muss der Nutzer erst eine große Menge an Objekte bewerten, damit sinnvoll Daten für das Profil erfasst werden können. Auch sollten bei der Erfassung des Feedbacks auf Beeinflussende Punkte zu den Quellen beachtet werden, da explizites Feedback durch die Transparenz dafür anfällig ist. Das implizite Feedback kann diesen Punkt durch die fehlende Transparenz abdecken und eine Manipulation der Empfehlungen verhindern. Auch die Nutzerreichweite ermöglich eine umfassendere Behandlung gegenüber der Sparsity.

Kapitel 5

Lösungsstrategie

Nach dem Vorstellen der verschiedenen Arten von Empfehlungssysteme und deren Gegenüberstellung der Vor- und Nachteile wird nun für den RSS-Aggregator ein Empfehlungssystem ausgearbeitet und unter Betrachtung der Vor- und Nachteile von expliziten und impliziten Feedback ein Benutzermodell erstellt. Die Auswahl des Empfehlungssystems und des Benutzermodells wurde anhand der Anforderungspunkte getroffen 1.2:

- **Empfehlungssystem**
  Das Empfehlungssystem wurde anhand der Anforderungspunkte 1 Verarbeitung von textuellen Daten, 2 der Nutzer möchte neues entdecken, 4 die individuelle Quellenauswahl und 5 die verschiedenen Themenbereiche erarbeitet.

- **Benutzermodell**
  Für das Benutzermodell wurden die Anforderungspunkte drei personalisierte Empfehlungen ohne Aufwand, sechs änderung der Interessen und sieben Aktualität erstellt.

5.1 Empfehlungssystem

Aus den Vor- und Nachteilen 3.3 ist erkennbar, dass die Anforderungen von einem reinen klassischen System nicht komplett gelöst werden können. Das kollaborative System bietet den Vorteil der Cross-Genre Empfehlung und ermöglicht die Entdeckung neuer RSS-Quellen. Dies unterstüzt den Anforderungspunkt 6. Leider hat ein rein kollaboratives System den Nachteil des New Item-Problem welches es für das System schwierig macht den Anforderungspunkt der möglichen Interessenänderung und der schnelllebigen Art des Mediums umzugehen. Der inhaltsbasierte Ansatz bietet den Vorteil anhand der Inhaltsanalyse neue Artikel direkt zuordnen und in den Empfehlungsprozess einfließen zu lassen. Dieser Vorteil ermöglicht es das Problem mit der schnelllebigen Art des Mediums zu umzugehen. Ein weiterer Vorteil ist die Nutzerunabhängigkeit. Der Nutzer kann die Anwendung auch unabhänging von anderen Nutzer verwenden. Dies ist ein unterstützender Punkt für die Freiheit des Nutzers seine Quellen individuell zusammenzustellen und sich unabhängig zu informieren. Leider bringt ein rein inhaltsbasiertes System die Nachteile der Overspecialization und der Limited-content-analysis, die einen Nachteil zur Erüllung des Anforderungspunkt unter 6 liefert. Als letzten Punkt besitzen beide klassischen System die Schwäche des New-User Problems. Aus diesen Gründen wurde sich für ein hybrides Empfehlungssystem entscheiden das die Vorteile der beiden klassischen Systeme verbindet. Die Entitäten wurden dazu in zwei Empfehlungsobjekte unterteilt, RSS-Quellen und RSS-Feeds wobei RSS-Feeds von den RSS-Quellen abhängen. Ausgehend von dieser Abhängigkeit wurde sich

für ein *kaskadiertes System* 3.1.3 entschieden. Burk evaluierte für ein kaskadiertes System, dass zuerst ein stärkeres System die Vorfilterung abdeckt und ein schwächeres die Feingranulierung übernimmt [11]. Aus diesem Grund konzentriert sich ein kollaboratives System in der ersten Kaskade auf das Finden neuer RSS-Quellen durch den Vorteil der Cross-Genre-Empfehlung und ein inhaltsbasiertes System übernimmt in der zweiten Kaskade die Rangbestimmung der Feedartikel für die Auflistung.

5.1.1 1. Kaskade kollaboratives System

In der ersten Kaskade soll der Vorteil der Cross-Genre Empfehlung des kollaborativen System genutzt werden. Dies ermöglicht dem Nutzer für ihn interessante Quellen zu finden, die ihm noch nicht bekannt sind. Ein inhaltlicher Ansatz wäre in der ersten Kaskade durch Anforderungpunkt 3 von Nachteil. Die Nutzer haben die Möglichkeit eine Quelle zu abonnieren oder nicht zu abonnieren. Auf dieser Grundlage wird ein boolisches Modell für die Merkmalsvektoren verwendet. Zur Berechnung der Empfehlung für einen Nutzer wurde die am häufigsten verwendete *gewichtete Summe* 3.4 mit einer kosinusbasierten Ähnlichkeitsfunktion 3.8 verwendet. Der Vorteil der *angepassten gewichteten Summe* die unterschiedliche Interpretationen der Bewertungsskala zu umgehen spielt bei dem boolischen Modell keine entscheidende Rolle, da eine Quelle vom Nutzer abonniert und nicht abonniert wird. Das Problem des New-User-Problems und Kaltstartproblems soll in der Anwendung durch die Auswahl der Webquellen umgangen werden. Entweder trägt der Nutzer selbst eine Quelle ein oder er hat die Wahl aus bereits eingetragenen Quellen.

5.1.2 2. Kaskade inhaltsbasiertes System

Zur Auflistung der abonnierten Feeds wird in der zweiten Kaskade ein inhaltsbasiertes System verwendet werden. Dies soll die Rangbestimmung der Feeds der Quellen für den Nutzer bestimmen. Durch den Vorteil das New-Item-Problem zu lösen, wurde sich für die Feingranulierung in der zweiten Kaskade für ein inhaltsbasiertes System entschieden. Feeds werden von abonnierten Quellen häufiger aktualisiert und müssen. Aus diesem Grund muss die zweite Kaskade mit mehreren neuen Objekten am Tag umgehen können. Das es sich bei Feeds um kurze textuelle Zusammenfassungen handelt wird im inhaltsbasierten Teil die unter 2 beschriebenen Technik zum aufbereiten der Feeds verwendet. Mit der Tf-Idf (2.2)-Gewichtung soll die Erkennung der Schlüsselwörter weiter gesteigert werden. Folgend werden die Feeds mit dem kosinusbasierten Ähnlichkeitsmaß in Relation zueinander in Beziehung gesetzt. Anhand der Schlüsselwörter sollen die Themen der Feeds wie unter Anforderung 5 gelistet beschrieben werden. Das New-User-Problem soll durch eine Auflistung der gewählten Feeds nach einer zeitlichen Ordnung nach dem Datum abgedeckt. Liegen noch keine Daten zu gelesenen Artikel im Benutzerprofil vor, soll mit dem Datum eine simple Ordnung der Feeds gewährleistet werden.

5.2 Benutzermodellierung

Das Benutzermodell muss durch den zweiteiligen Aufbau des kaskadierten Empfehlungssystems mit zwei Empfehlungsobjekten umgehen können, zum einen die RSS-Quellen und die RSS-Feeds. RSS-Quellen sollen es dem Nutzer ermöglichen neue Quellen zu finden und somit neue thematische Feeds zu erhalten. Als Anforderungspunkt der individuellen Quellenwahl wurde sich für explizites Feedback entschieden, da es den Vorteil der Transparenz besitzt und

dem Nutzer so ermöglicht auch direkten Einfluss durch die Polarity auf die Empfehlungen zu nehmen. Implizites Feedback eignet sich für diesen Bereich durch die Sensibilität gegenüber Rauschen und dem einseitigen Feedback nicht direkt zur Wahl der Quellen, da der Nutzer weiterhin die Kontrolle über die Quellen haben soll. Zur Feedbackerfassung in der zweiten Kaskade bietet sich das implizite Feedback an. Durch das Kaskadierte Empfehlungssystem wird im Vorverarbeitungsschritt potenzielle Quellen ausgewählt. Damit besteht für den Nutzer schon ein grobes Interesse an den Veröffentlichungen der Feeds. Diese sollen dann in der zweiten Kaskade durch die Rangordnung weiter verfeinert werden. Explizites Feedback ist für den Nutzer deshalb im zweiten Verfeinerungsschritt nicht geeignet. Durch das potenzielle Wachstum von Feeds von mehreren Quellen ist die Sparsity ein Nachteil für den Nutzer zum Anordnen der Feeds. Implizites Feedback eignet sich durch nicht bewusste Sammlung des Nutzer dadurch besser zur Datenerfassung. Daher kann das Problem des einseitigen Feedback vernachlässig werden. Auch die Sensibilität der gegenüber dem Rauschen kann dabei vernachlässigt werden, da falsche Ordnungen vom Nutzer hingenommen werden können, wenn weiterhin alle Feeds der Quellen dargestellt werden ud diese sich in einem zeitlich aktuellen Rahmen befinden. Die Abdeckung zum Nachteil der Transparenz kann durch das explizite auswählen der RSS-Quellen abgedeckt werden. Der Nutzer erfährt immer noch von welcher Quelle der RSS-Fedd kommt und kann nachvollziehen warum dieser in der Liste auftaucht. Das Wissen über die Domain zur Modellbildung findet hier in einem überschaubaren Rahmen zur Ordnung der Feeds statt.

5.2.1 Benutzermodell

Daniel Billsus und Michael J. Pazzani beschreiben in ihrem Artikel einen Ansatz für ein Benutzerprofil zur News-Klassifikation das anschließend in NewsDude Anwendung fand [27]. Als Grundlage für das Benutzermodel entwickelten sie ein zweiteiliges Benutzerprofil, dass zum einen durch ein *Short-term*-Profil als erste Stufe für kurzzeitige Interessen des Nutzers zu einem bestimmten Thema aufzeichnet und ermöglichen soll mehrere zusammenhängende Informationsstränge zu verfolgen. Vom Nutzer nicht gesehene Artikel werden dann basierend auf den gesehenen Artikel durch einen Nearest-Neighbor-Algorithmus klassifiziert. Durch Verwendung des Kosinus-Ähnlichkeitsmaß und einem TF-IDF-Derivats ordneten die Autoren ähnliche Newsartikel in Vectorspace- Modellrepräsentation zu bestehenden Artikeln zu. Durch einen minimalen Schwellwert t_{min} werden zu stark ähnliche Artikel gefiltert. Auf diese Weise soll verhindert werden, dass der Nutzer nicht mehrere Artikel empfohlen bekommt, die er schon kennen könnte. Der Vorteil des *Nearest-Neighbor* Ansatzes ist, dass keine große Datenbasis von mehreren Artikel zum aufgreifen von neuen Informationssträngen benötigt wird. Neue Interessen schon mit dem ersten gelesenen Artikel in die Bewertung mit ein [27]. Das *Short-term*-Profil arbeitet mit implizitem Feedback und bestimmt die Artikel über Techniken eines inhaltsbasierten Empfehlungssystems. Damit eignet sich eine Umsetzung mit diesem Profil in der zweiten Kaskade des Benutzerprofils.
Als zweiten Teil modulierten Daniel Billsus und Michael J. Pazzani das *Long-term*-Profil. Dieses soll die allgemeinen Interessen des Nutzers thematisch abdecken, wenn keine Artikel als Folgeartikel von gelesenen Artikeln aus dem *Short-term*-Profil gefunden wurden. Die Autoren verwendeten dazu eine Vorkategorisierung von Artikelthemen, die durch die Höhsten über Tf-Idf gewichteten Terme aller Newsartikel einer Kategorie beschrieben wurde. Auf diese Weise ordneten Billsus und Pazzani Feature-Terme in das, für alle Nutzer gültige *Long-term*-Profil einer Kategorie. Die Aufbereiten Feature-Terme verwendeten die Autoren dann als Trainingsmenge für einen naiven Bayesklassifikator. Dieser klassifiziert die Artikel folgend in zwei Klassen,

Interessant und *Nicht Interessant*. Die Wahrscheinlichkeit das eine Newsgeschichte zu einer Klasse j gehört, wird anhand der Vorkommen der Feature-Terme innerhalb des Artikel mit $p(class_i|f_1,f_2,...,f_n)$ klassifiziert.

$$p(class_j) = \sum_{i}^{n} p(f_i|class_j)^{N_i} \tag{5.1}$$

wobei N_i die Frequenz von f_i beschreibt. Nachfolgend wurden zur Verbesserung der Wahrscheinlichkeitsbestimmung zwei weitere Kriterien zur Qualitätssicherung der Klassifikation für die Schlüsselwörter eingeführt.

1. Feature f_j kommt i-mal im Artikel vor
2. $p(f|Interessant) \geq p(f|\neg Interessant)$ um zur Klasse Interessant zu gelten

Zum einen werden die Artikel mit einer nicht genügend großen Anzahl an Schlüsselwörtern gefiltert. Zum andern muss der Schätzung des Klassifikators für die Klasse *Interessant* größer sein, als für die Komplimentäre Klasse, als Verstärkung, dass auch wirklich Interessante Artikel klassifiziert werden. Algorithmus 1 von Daniel Billsus und Michael J. Pazzani zeigt die Klassifikation eines vom Nutzer ungesehenen Artikels.

Algorithm 1: Benutzermodell nach Daniel Billsus und Michael J. Pazzani

Data: story u
Result: score
if $\exists d : d \in \{short\text{-}term\text{-}sotries\} \vee cosinus\text{-}similarity(d,u) \geq t_{min}$ **then**
  score = nearst-neighbour-prediction(u,$\{short - term - stories\}$)
else if $\exists\{f_1,f_2,\ldots,f_n\} : \forall f \in \{f_{u1},\ldots,f_{un}\} p(f|c) \geq p(f|\neg c)$ **then**
  score = native-bayes-prediction(u,$\{all - strories\}$)
else
  score = default

Das *Long-term*-Profil von Daniel Billsus und Michael J. Pazzani stützt sich auf mehrere vordefinierte Kategorien die durch einen inhaltsbasierten Ansatz klassifiziert werden. Durch die Verwendung eines kollaborativen Ansatzes wurde das *Long-term*-Profil für die erste Kaskade abgeändert. Zur Empfehlung von neuen Quellen wird ein nutzerbasiertes Verfahren mit einer gewichten Summe (Formel 3.4) und dem Kosinusähnlichkeitsmaßes (Formel 3.8) verwendet.

Explizite und Implizite Kriterien

Aus der Abwägung von Vor- und Nachteilen wurde sich für die ersten Kaskade für explizites und für die zweite Kaskade für implizites Feedback entscheiden. Die Informationeneingabe des Nutzers sollten aus dem Anforderungspunkt 3 1.2 so einfach wie möglich sein. Die Tätigkeiten, die ein Nutzer innerhalb der ersten Kaskade tätig ist die Auswahl der Quellen. Somit wird vom Nutzer die Auswahl der Quellen als explizites Feedback aufgenommen. Der Nachteil des impliziten Feedback, dass das entwickeln des Modells aufwendig ist, wurde auch dieses so simpel wie möglich gehalten. Deshalb wird nur der Punkt des Lesens eines Feeds aus Tabelle 4.1 verwendet.

Vergessensfunktion

Die Autoren Cansheng Ji und Jingyu Zhou [22] beschreiben in ihrer Studie mehrere Recommendation-Features für einen RSS-Reader. Dazu wird ein Feature der *Inverse-Update-Frequency* eingeführt (Formel 5.2):

$$IUF(src_i) = 1 - \frac{n(src_i)}{n_{max}} \tag{5.2}$$

wobei $n(src_i)$ die Anzahl veröffentlichen Feeds der RSS-Quelle und n_{max} die maximale Anzahl an veröffentlichen Feeds pro Tag beschreibt. Dies verfolgt die Theorie, dass die Qualität einer RSS-Quelle von der Artikel-Update-Frequenz abhängt. Daher eine niedrige Veröffentlichkeitsrate ist ein Zeichen von hoher Qualität und eine hohe Veröffentlichkeitsrate eine niedriges Qualitätsmaß für die RSS-Quelle. Auf Grundlage der Veröffentlichkeitsrate einer RSS-Quelle wird die Vergessensfunktion des Benutzerprofils erstellt. Zum einen sollen RSS-Quellen mit einer niedrigen Veröffentlichkeitsrate länger im Profil bestehen können, da potenzielle neue Artikel der RSS-Quelle länger benötigen um veröffentlicht zu werden. Zum anderen sollen Artikel mit einer hohen Atrikelfrequenz schneller wieder aus dem Benutzerprofil gelöscht werden, da die Quelle potentiell viele Artikel produziert und eine Dopplung von Themen stattfinden kann. Dadurch wird die IUF als Skalierungsfaktor einer RSS-Quelle für die Löschfunktion genommen.
Im Artikel von Cansheng Ji und Jingyu Zhou [22] wurde die IUF in Abhängigkeit von der täglichen Veröffentlichung abgeleitet. Um für eine RSS-Quelle den Veröffentlichkeitsfaktor zu bestimmen wird ein *Moving-Average* über die Zeit bestimmt. Der Moving-Average ermöglicht eine iterative Berechnung des Mittelwerts über einen Zeitraum. Dies ermöglicht eine Ermittlung der Veröffentlichkeitsfrequenz einer RSS-Quelle. Formel 5.3 beschreibt den iterativen Moving-Average über einen Zeitraum t [34]:

$$m^{(n)}(t) = m^{(n)}(t-1) + \frac{x(t)}{n} - \frac{x(t-1)}{n} \tag{5.3}$$

Das Vergessen der Artikel in den einzelnen Profilen soll über eine Datumsdifferenz zum Erscheinungstag und dem aktuellen Datum realisiert werden. Für die Bestimmung der Lebensdauer eines Artikels wird eine exponentielle Halbwertkeitsfunktion aus der Physik genommen. Diese beschreibt den Zerfall eines Teilchensystems über die Zeit. Durch die Abhängigkeit, dass Newsartikel auch über die Zeit an Aktualität verlieren bietet sich dieses Verfahren an. Desto älter ein Artikel ist, desto weniger aktuell ist dieser. Daher spielt die Differenz zwischen dem Aktuellen Datum (d_{crr}) und dem Erscheinungsdatum (d_{feed}) einen Wert innerhalb der Gewichtung von Artikeln. Zusätzlich wird der Rang des Empfohlenen Artikels innerhalb der Empfehlung betrachtet. Hinzu kommt als Faktor die vorher genannte IUF 5.2 unter Verwendung des Moving-Average zu einer RSS-Quelle.

$$f(d_{crr},d_{feed},src_i) = e^{-IUF(moveavg(src_i))\cdot\frac{1}{d_{crr}-d_{feed}}} \tag{5.4}$$

Als statischer Wert wird 0.1 gewählt. Unterschreiten Feeds im Nutzerprofil den Wert von 0.01 werden diese aus dem Profil gelöscht. Abbildung 5.1 zeigt den Verlauf der Gewichtungsfunktion.

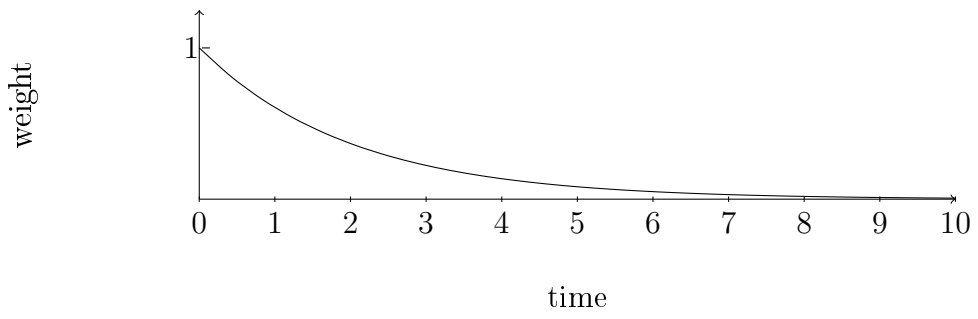

Abbildung 5.1: Vergessenfunktion 5.4

5.3 Zusammenfassung

Anhand der Anforderungen wurde ein kaskadiertes Empfehlungssystem mit einer ersten kollaborativen Kaskade zur Entdeckung von neuen Quellen und einem inhaltsbasierten in der zweiten Kaskade entwickelt. Ausgehend von Daniel Billsus und Michael J. Pazzani [27] wurde das Benutzerprofil 1 als Grundlage genommen, im *Long-term*-Profil angepasst und durch eine Vergessensfunktion (Formel 5.4) erweitert. Der Vorteil des Models ist die Anordnung von ähnlichen folgenden Geschichten. Dieses Benutzermodell ermöglicht es dem Nutzer von gewählten RSS-Quellen nun verschiedene Informationsstränge zu verfolgen und nicht gesehene Artikel implizit zu bewerten. Durch explizites Feedback in der ersten Kaskade hat der Nutzer kompletten Zugriff auf die verwendeten RSS-Quellen. Durch das Löschen kann er direkt negatives Feedback zu dem Quellen geben. Auf dieser Grundlage besitzt er weiterhin komplette Kontrolle über die abonnierten Quellen, kann aber auch neue entdecken. ïż£

Kapitel 6

Evaluierung

In diesem Kapitel wird das Empfehlungssystem auf seine Empfehlungsqualität, das Nutzermodell und der Learning Algorithmus mit verschiedenen Kriterien untersucht. Es wurde für die Simulation ein System prototypisch implementiert. Im ersten Teil wird auf die Erstellung des Datensatzes zum Testen des kaskadierten Systems begonnen.

Dazu wurde zum einen ein Datensatz für die Webquellen von *rss-nachrichten.de* und ein Datensatz von Feedartikeln erstellt. Folgend werden die Kriterien zur Evaluierung des Benutzermodells beschrieben und der Simulationsablauf vorgestellt. Dazu wird ein Benutzer-Zielmodell P_u erstellt und über k Simulationsrunden die Entwicklung des Systembenutzerprofils zum Artikelranking untersucht. Das Benutzer-Zielmodell soll den Nutzer mit seinen Interessen abbilden und als Entwicklungsziel für das Empfehlungssystem dienen. Innerhalb der Simulation wird eine Teilmenge an Newsartikel durch P_u und das Empfehlungssystemprofil P_s gerankt. In jedem Schritt wird durch die Normalized Distance-based Performance Measure die Distanz zwischen diesen beiden Rankings berechnet. Nach jedem Schritt wird vom Zielmodell ein simuliertes Feedback an das System gegeben. Anschließend wird das verwendete Benutzermodell mit dem bestehenden Profil von Newsdude verglichen, die RSS-Quellenempfehlung mit einem inhaltsbasierten und kollaborativen System verglichen. Zum Schluss werden die Ergebnisse vorgestellt, diskutiert und ausgewertet.

6.1 Datensatz

Als Datenbasis wurde über vier Wochen die RSS-Quellen von *www.rss-nachrichten.de*[1], *www.rssverzeichnis.net*[2] und *www.rssverzeichnis.de*[3] gesammelt und täglich abgerufen. Jeder Quelle wurde dazu einer Kategorie von der jeweiligen Seite zugeordnet. Insgesamt wurden 1258 RSS-Quellen mit 136 Kategorien gesammelt. Durch Überschneidungen wurden einige Kategorien händisch zusammen gelegt, um nicht zu viele thematische Dopplungen zu haben. Nach zusammenführen der Kategorien der RSS-Quellen wurden über vier Wochen insgesamt 57986 Feeds gesammelt. Weiterhin wurde der Datensatz von Feeds bereinigt die entweder keinen Titel, Text oder Datumsangabe besaßen. Durch die Bereinigung sollten die Inhalte durch das inhaltsbasierte System mit einer Tf-Idf erkannt werden. Nach der Bereinigung blieben 44102 Feeds für die Simulation übrig. Die Kategorien der RSS-Quellen reichen von kleinen thematischen Blogs wie Programmierung, preisvergleichen und Hobbygärtnern bis zu großen Newsonlinezeitungen,

[1] www.rss-nachrichten.de - Stand 6.10.2016
[2] www.rssverzeichnis.net - Stand 6.10.2016
[3] www.rssverzeichnis.de - Stand 6.10.2016

Kategoriename	Anzahl Sources	Anzahl Feeds
Reisen	45	1899
Bildung-Beruf	27	1884
nachrichten-medien	11	1754
Haustiere	7	1746
Finanzen	30	1665
⋮	⋮	⋮
Anhänger-Garagen	1	10
Essen und Trinken	1	10
Heimwerker	1	10
Schiffe - Boote	1	8
Organisation	1	6

Tabelle 6.1: Fünf höhsten und niedrigsten Kategorie

wie spiegel.de, tagesschau.de oder zeit.de. Tabelle 6.1 zeigt die fünf größten Kategorien mit ihrer Anzahl der Quellen und der Feeds. Abbildung 6.1 zeigt die Verteilung von Anzahl der Feeds pro Kategorien von Größten zur Kleinsten Klasse.

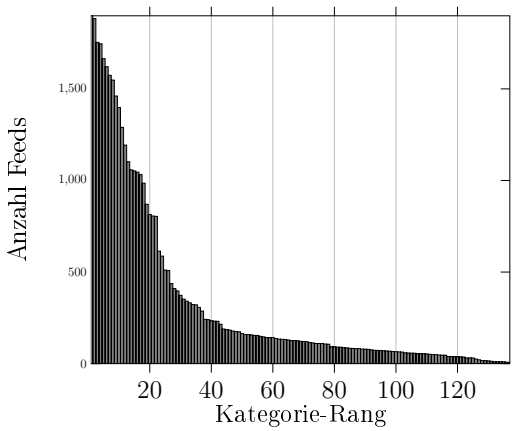

Abbildung 6.1: Verteilung der Kategorien des Datensatzes

6.2 Implementierung

6.2.1 Backend

Zur Verarbeitung der Texte des RSS-Aggregator wurde ein Web-Server entwickelt der es den Nutzern erlaubt medienunabhängig auf die Feeds zuzugreifen. Die Aufgaben des Servers belaufen sich auf das Verwalten der RSS-Quellen, das Aktualisieren, das Parsen und das Aufbereiten des Inhalts der einzelnen Feeds für das Empfehlungssystem, sowie eine Nutzerprofilverwaltung.

6.2.2 Frontend

Das Frontend bildet die Schnittstelle für den Nutzer zu der Anwendung. Um zu gewährleisten, dass der Nutzer von unterschiedlichen Medien auf den Server zuzugreifen, wurde als Client ein

Webbrowser verwendet. Webbrowser bieten den Vorteil die Feeds über HTML5 und CSS3 zu visualisieren, Interaktionen durch Javascript und Daten über das JSON-Format Mediumunabhängig zu nutzen. Daher kann der Nutzer mit einem mobilen Gerät, wie Smartphone oder Tablet, sowie einem Desktop darauf zugreifen. Um das Layout und Styling auf allen Geräten zu garantieren wurde Bootstrap[4] von Twitter eingebunden. So lassen sich über CSS-Klassen die Ansichten für unterschiedliche Bildschirmgrößen der Medien definieren. Zum Bearbeiten des DOM (Document Object Model) wurde javascriptbasierte Framework JQuery[5] eingebunden. Das Framework ermöglicht einen einfachen Zugriff auf bestimmte Container des DOMs, diese zu bearbeiten und via POST- oder GET-Abfragen per Ajax mit dem Server zu kommunizieren. Den Zugriff des Nutzers zu seinem Profil findet über das Einloggen zu Server statt.

Aufbau

Der Aufbau wurde so gestaltet, dass der Nutzer einen intuitiven Eindruck bekommt. Über diese Menü-Leiste hat der Nutzer die Möglichkeit sich entweder unter Neuste die Liste nach Datum zu sortieren oder das Empfehlungssystem unter Empfehlungen zu nutzen. Unter dem Punkt Einstellungen besteht die Möglichkeit, die RSS-Quellen zu organisieren.

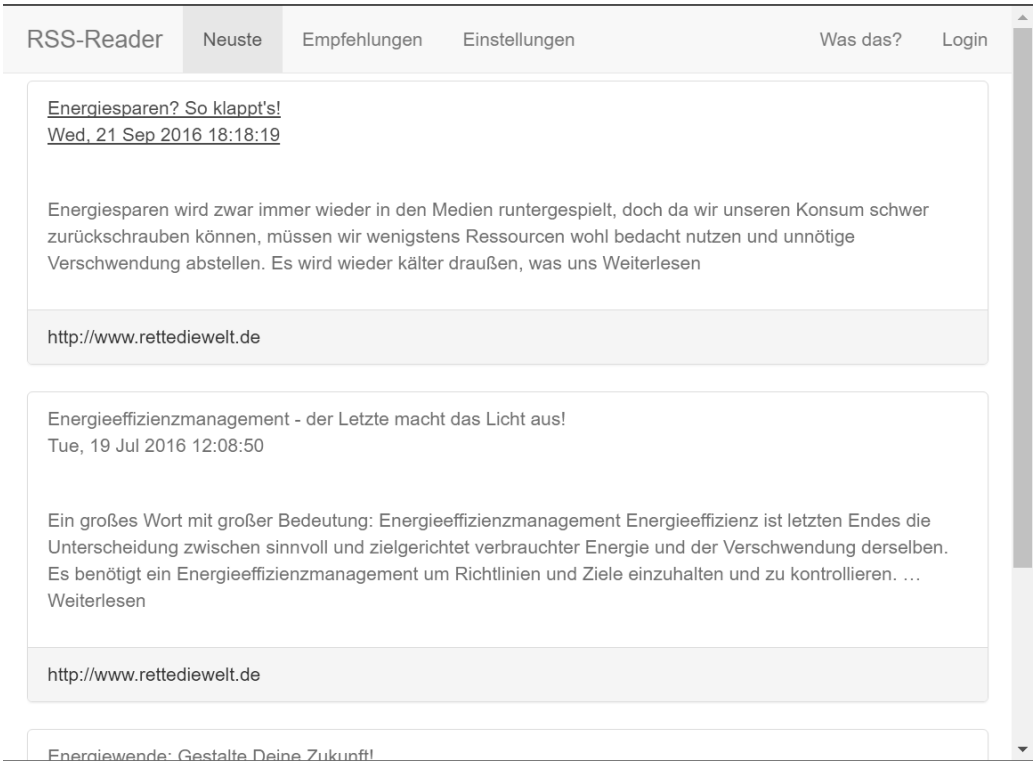

Abbildung 6.2: Frontend Feedsauflistung

Dort hat er die Optionen entweder durch eine Auflistung im Server existierender Quellen durch Checkboxen zu abbonieren oder abzubestellen oder durch einen Buttonklick eine neue Quelle einzutragen. Eine neu eingetragene Quelle wird von Server validiert und mit Channelinformationen dem Nutzer präsentiert. Nach Bestätigung der Quelle wird diese im Server eingetragen, aktualisiert und zum Nutzerprofil hinzugefügt. Sonst wurden eine Login-Funktionalität

[4]http://getbootstrap.com/ - Stand 8.11.2016
[5]https://jquery.com/ - Stand 8.11.2016

zum anmelden und eine Beschreibung der Arbeitsweise hinzugefügt. Die Abbildung 6.2 zeigt eine beispielhafte Listenansicht für den Nutzer. In der Mitte wurde der Platz für die Liste der Feeds vollständig ausgenutzt. Die Liste liefert durch ein scrollbares Feld eine geordnete Übersicht aller abonnierten Feeds. Die einzelnen Einträge werden einzeln als Blöcke dargestellt und geben anhand der Überschrift und dem Datum die ersten Informationen des Feeds. Ein Button weist dem Nutzer auf die Möglichkeit hin den Teaser aufzuklappen und zu lesen. Nachdem der Nutzer den Artikel gelesen hat, besteht für ihn die Möglichkeit über den zurück-Button wieder zur Listenansicht zu wechseln. Das Klickevent wird zum Backend gesendet. Dieses nimmt dann den Feed nach einer Lesezeit im Short-term-Nutzerprofil auf.

6.3 Evaluierung

Zur Evaluierung des kaskadierten Systems werden die beide Teilkomponenten durch die Abhängigkeit von RSS-Quellen und Feeds einzeln untersucht.

- 1. Kaskade - RSS-Quellen
- 2. Kaskade - Feeds

Die Aufgabe der ersten Kaskade ist die Empfehlung von neuen RSS-Quellen. Dazu wird die Messung mit Precision und Recall durchgeführt. Die Aufgabe der zweiten Kaskade ist die Anordnung der, von den gewählten Quellen veröffentlichen Feeds. Dwi H. Widyantoro [39] verwendete zum Vergleichen des Rankings die Normalized Distance-based Performance Metrik. Beide Distanzen sind in zwei Simulationen durchgeführt worden um beide Kaskaden zu testen.

6.4 Benutzersimulation

Zur Evaluierung der Benutzermodelle und des Empfehlungssystem wurde sich aus Gründen der Reproduzierbarkeit für eine Offlineevaluierung entschieden. Dazu wurde sich an die Evaluierung von Dwi H. Widyantoro [39] zur Auswertung des Benutzermodells orientiert. Dazu führte er fünf verschiedene Kriterien wie Accuracy, Normalized Distance-based Performance Measure, Spearman Rank-order Correlation Coefficient, Average Document Score und Precision versus Recall ein. Diese werden nachfolgend näher beschrieben. Das Experiment lernt ein Benutzerprofil zu einem Set an Dokumenten und vergleicht dieses mit einem Ziel-Benutzersystem P_u. Durch den Vergleich wird die die Entwicklung des System-Benutzermodells P_s bewertet.

Accuracy

Accuracy bestimmt den Wert in wieweit der simulierte Nutzer und das System innerhalb der gewählten Dokumente von D_{news} entsprechen. Die beiden Testset D_{pref} und D_{retr} werden von D_{news} durch die jeweiligen Profile gefiltert. [39] beschreibt die formale Definition der accuracy.

$$Accuracy(D_{pref},D_{retr}) = \frac{1}{D_{retr}} \sum_{d \in D_{retr}} f(d), \tag{6.1}$$

$$f(d) = \begin{cases} 1 & d \in D_{pref} \\ 0 & otherwise. \end{cases} \tag{6.2}$$

Das Empfehlungssystem und der simulierte Nutzer bewerten die Dokumente und nehmen die n höchsten. Die accuracy gibt dann den prozentualen Anteil gleicher Dokumente vom simulierten Nutzer und des Systems an.

Normalized Distance-based Performance Measure (ndpm)

Y.Y. Yao [39] beschreibt in seinem Artikel eine neue Distanzberechnung zwischen Nutzerbewertung und Systembewertung eines Ranking vor. Aus Studien geht hervor, dass die Entwicklung von Nutzerbewertungen über Zeit besser durch relative als absolute Bewertungen ist. Die Distance-based Performance Measure ermöglich ein relatives Vergleichen von Dokumentpaaren durch P_u und P_s erstellte Rankings. Dazu definiert Y.Y. Yao eine Relation $\succ$ als Relevanz für Nutzer innerhalb eines Rankings auf eine Menge an Dokumenten $D = \{d1,d2,d3,d4\}$.

$$d \sim d' \Leftrightarrow \text{Nutzer bevorzugt } d \text{ vor } d' \Leftrightarrow Score(P,d) > Score(P,d') \tag{6.3}$$

Anhand der Relation wird durch eine Funktion $u : D \rightarrow R$ wird das Ranking mit einer Ordinalität belegt. Anhand dieser Relation wird eine Distanz zwischen zwei Ranking bestimmt, die beschreibt wie ähnlich diese sich sind. Sei $\succ_u$ und $\succ_s$ die Rankingrelation vom Nutzermodell und Systemmodell, dann wird die Distanz zwischen zwei Dokumenten $d,d' \in D$ zu diesen beiden Ranking berechnet mit,

$$\delta \succ_u , \succ_s (d,d') = \begin{cases} 2 & if(d \succ_u d' \wedge d' \succ_s d) \vee (d \succ_s d' \wedge d' \succ_u d) \\ 1 & if(d \succ_s \vee d' \succ_s d) \wedge d \sim_u d' \\ 0 & \text{sind gleich} \end{cases} \tag{6.4}$$

Basierend auf dem Ranking des Nutzers und des Systems ist die Distanz 2 wenn sich die Bewertung der Profile für zwei Dokumente widersprechen. Wenn die Bewertung von gleichen Dokumentenpaaren kompatibel (daher d, d durch das Nutzerprofil als relevant oder nicht relevant zugeordnet wurde) wird dies mit 1 bewertet. Wenn die Ordnung für zwei Dokumente gleich ist wird dies mit 0 bewertet. Die gesamte Distanz zwischen zwei Rankings berechnet durch Summierung der Dokumentenpaare berechnet. Durch dieses Maß soll das relative empfinden des Nutzer abgebildet werden.

$$\beta(\succ_u , \succ_s) = \sum_{d,d'} \delta \succ_u , \succ_s (d,d') \tag{6.5}$$

Und anschließend normalisiert.

$$ndpm = \frac{\beta(\succ_u , \succ_s)}{2 \cdot count(docpairs)_{\succ}} \tag{6.6}$$

Die Leistung des Systems ist dann am besten, wenn $ndpm = 0$, also die Rankings der beiden Profile komplett übereinstimmen.

Spearman Rank-order Correlation Coefficient

Definiert ein monotones Maßfür ordinal Messdaten, welches auf deren Rängen basiert. Somit wird die Entwicklungskorrelation eines Dokuments zu allen Dokumenten zwischen dem Zielnutzerprofil und Systemprofil bestimmt. Sei U_i der Rang des Dokuments d_i zwischen allen anderen d's bewertet auf dem Zielnutzerranking und dem Systemranking. So ist der SpearmanRankkoeffizient r_s definiert durch die Gleichung

$$r_s = 1 - \frac{6 \sum_{i=1}^{|D_{retr}|} (U_i - S_i)^2}{|D_{retr}|3 - |D_{retr}|} \tag{6.7}$$

Dies soll als Vergleichwert zur Npdm dienen.

Average Document Score

Wertet die Bewertung eines Profils zu einem Dokument. Für einen Nutzer soll es die absolute Entscheidung zu einem Dokument widerspiegeln. Für ein Dokumentset D_{retr} wird der average document score berechnet durch

$$Score(P_u,D_{retr}) = \frac{1}{|D_{retr}|} \sum d \in D_{retr} \text{ Score}(P_u,d_i)) \tag{6.8}$$

Die Dokumente des Empfehlungssets werden durch den simulierten Benutzer bewertet und ergeben im besten Fall 1. Anhand dieser Metrik wird die absolute durchschnittliche Empfehlung der Dokumente bestimmt.

Precision versus Recall

Precision und Recall sind klassische Leistungsparameter aus dem Information Retrieval. Sie beschreiben die Relevanz von empfohlenen Dokumenten anhand binärer Werte. Tabelle 6.2 zeigt schematisch die Parameteraufteilung zur Berechnung der Precision und des Recalls. Dazu werden Dokumente in zwei Klassen eingeteilt, relevant und nicht relevant. Als zweiter Punkt werden die Dokumente in *empfohlene* und *nicht empfohlene* eingeteilt. Formel 6.9 zeigt die Berechnung von der Precision durch das Verhältnis von relevanten und empfohlenen Elementen. Recall (Formel 6.10) beschreibt die Wahrscheinlichkeit, dass ein Objekt relevant ist und somit wie wahrscheinlich es ist das es empfohlen wird.

	Relevant	Irrelevant
Erhalten	a	b
Nicht erhalten	c	d

Tabelle 6.2: Parameter für die Berechnung von Precision und Recall

Die Berechnung der Precision (Formel 6.9)) und des Recalls (Formel 6.10) ergeben sich aus [40]:

$$precision = \frac{a}{a+b} \tag{6.9}$$

$$recall = \frac{a}{a+c} \tag{6.10}$$

wobei a die Anzahl der relevanten empfohlenen Elemente, $a+b$ die Anzahl der empfohlenen Elemente und $a+c$ die Anzahl der gesamten relevanten Elementen im gesamten Dokumentenkorpus sind. Precision definiert damit den Anteil aller relevanten Elemente innerhalb der empfohlenen Elemente. Und Recall den Anteil aller relevanten Elemente aller Möglichen Elemente.

6.4.1 Aufbau der Simulation

Zur Evaluierung des Empfehlungssystem wurde sich am Offlineaufbau von Dwi H. Widyantoro orientiert [39]. Dazu wurde in Schritt 9 ein Feedback für Quellen hinzugefügt sowie die Berechnung des Precision / Recalls in Schritt 7. Da die Artikel jeweils von den RSS-Quellen abhängen wurde nicht wie in Schritt 3 zufällig Dokumente aus dem Dokumentenkorpus gewählt, sondern diese jeweils von dem Zielbenutzer gewählten RSS-Quellen. Abbildung 6.3 zeigt eine Übersicht des Simulationablaufs.

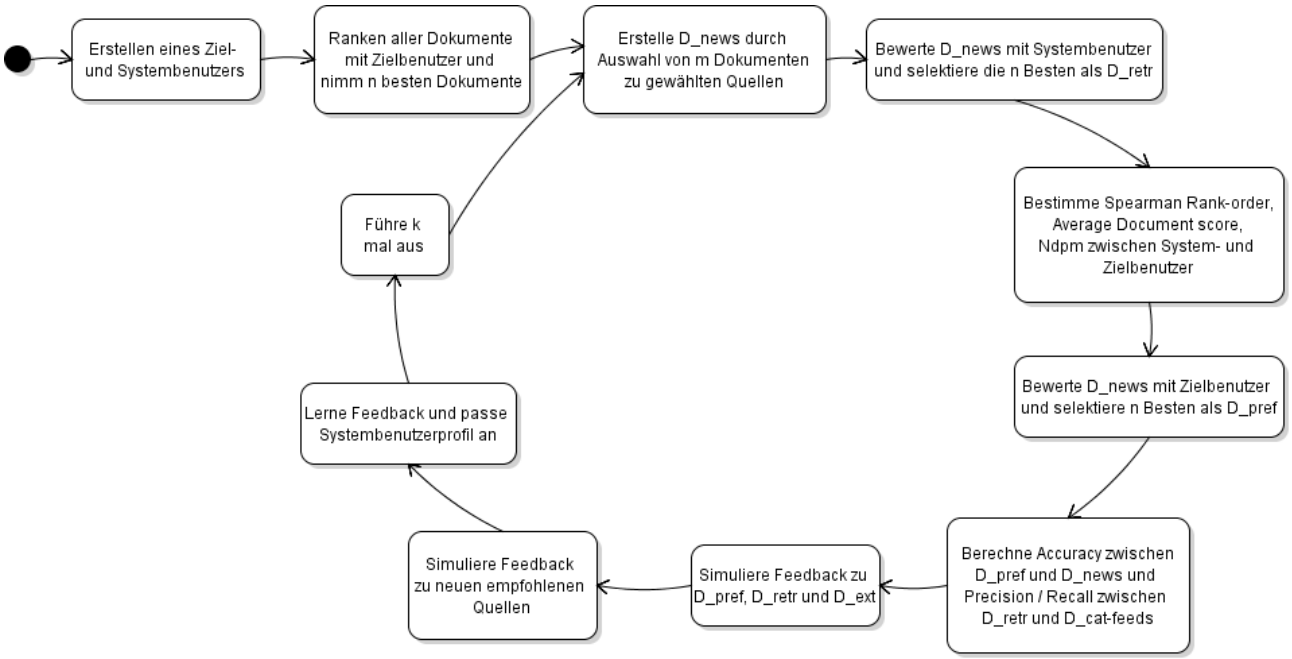

Abbildung 6.3: Simulationsablauf

1. Erstellung eines zufälligen Ziel Benutzerprofils P_u und einem System-Benutzerprofils P_s
2. Ranke alle Artikel der Dokumentensammlung mit P_u und nimm die n besten Elemente D_{ext}
3. Erstelle ein Set D_{news} durch zufälliges nehmen von m Dokumenten von aller Zielbenutzer selektierten RSS-Quellen
4. Bewerte mit Benutzerprofil P_s die Dokumente und selektiere die n Besten D_{retr}.

5. Bestimme Spearman Rank-order Correlation Coefficient, Average Document Score und Normalized Distance-based Performance Measure zwischen den beiden Profilen.
6. Bewerte D_{news} mit dem Zielbenutzermodell und selektiere die n Besten Dokumente D_{pref}.
7. Berechne die Accuracy zwischen D_{pref} und D_{retr} und Precision / Recall zwischen D_{retr} und $D_{cat-feeds}$
8. Simuliere Feedback von Dokumenten D_{pref}, D_{retr} und D_{ext}
9. Simuliere Feedback zu neuen empfohlenen und bestehenden Quellen.
10. Lerne Feedback zu D_{retr} und $D_{sources}$ und passe P_s an
11. Führe Schritt 3 bis 10 k-mal aus

6.4.2 Zielbenutzer

Um die Entwicklung des Benutzermodells untersuchen zu können, verwendete Dwi H. Widyantoro mehrere Zielbenutzer. Diese definieren das Offlinebenutzermodell als Ideal für die Entwicklung des Systembenutzerprofils P_s. Durch die unter beschrieben 6.4 Metriken zwischen den beiden Profilen wird die Entwicklung des Benutzermodells bewertet. Dazu wurde jedem Zielbenutzer eine Höhste favorisierte Kategorie zugeordnet. Ausgehend von der Lieblingskategorie wurden zu allen Kategorien ein Tf-Idf-gewichteter Textkorpus erzeugt. Durch die Kosinusmetrik wurden die Abstände zwischen den Kategorien bestimmt. Die Abstände wurden dann genutzt um für jeden Nutzer eine Ordnung der Kategorien zu erzeugen.
Um die Rang-Ordnung der Feeds im Detail zu bestimmen wurden die Kosinusdistanzen zwischen den einzelnen Feeds zum Kategorie-Merkmalsvektor bestimmt. Als letzter Punkt wurde für den Zielnutzer auf die gleiche Weise wie bei den Feeds eine Wertigkeit für die RSS-Quellen erstellt. Den Rang der Quellen wurde anhand ihrer Feeds und der Distanz zum Kategorie-Merkmalsvektor bestimmt. Somit gibt es eine Ordnung der Feeds für den Zielbenutzer um das inhaltsbasierte System zu testen, zum anderen gibt es ein favorisierte Ordnung der Quellen um die Empfehlung der Quellen zu bewerten.

6.4.3 Feedbacksimulation

Das Feedback soll dem System vermitteln wie der Benutzer auf die Anordnung der Rangliste reagiert. Zum einem wird dem System P_s die Auswahl der gelesenen Feeds simuliert zum anderen die Auswahl der neu empfohlenen RSS-Quellen ausgewählt. Dwi H. Widyantoro stelte zwei Funktionen für seine Feedbacksimulation auf. Zu einem führte er zur Erzeugung eine Sättigungsfunktion mit $\frac{1}{e^{3*Score(P_u,Dretr)}}$ ein [38]. Diese sollte das Interesse des Nutzer an einer Kategorie simulieren und gegebenenfalls negatives Feedback liefern. Zum anderen wählte er für positives Feedback die Funktion

$$fb(P_u,d_i) = \frac{2}{1 + e^{-5*Score(P_u,d_i)}} - 1 \tag{6.11}$$

unter der Abhängigkeit der Bewertung des Zielbenutzers zum Dokument d_i. Da das Benutzermodell in der Bachelorarbeit nur positives Feedback verwendet, wird für den Zielbenutzer nur die Funktion zu 6.11 genutzt. Als Grundlage für das Feedback wurde die jeweilige Rangordnung der Feeds und Quellen des Zielbenutzer genommen. Das Ziel ist, dass wenn der Zielnutzer

Feeds bzw. Quellen mit einer hohen internen Bewertung zu seiner Lieblingsquelle bzw. -feed bekommt, eine höhere Wahrscheinlichkeit besteht, dass er diese auswählt, zum anderen soll eine geringere Wahrscheinlichkeit bestehen wenn Feeds oder Quellen mit weniger Interesse zur Auswahl gestellt werden.

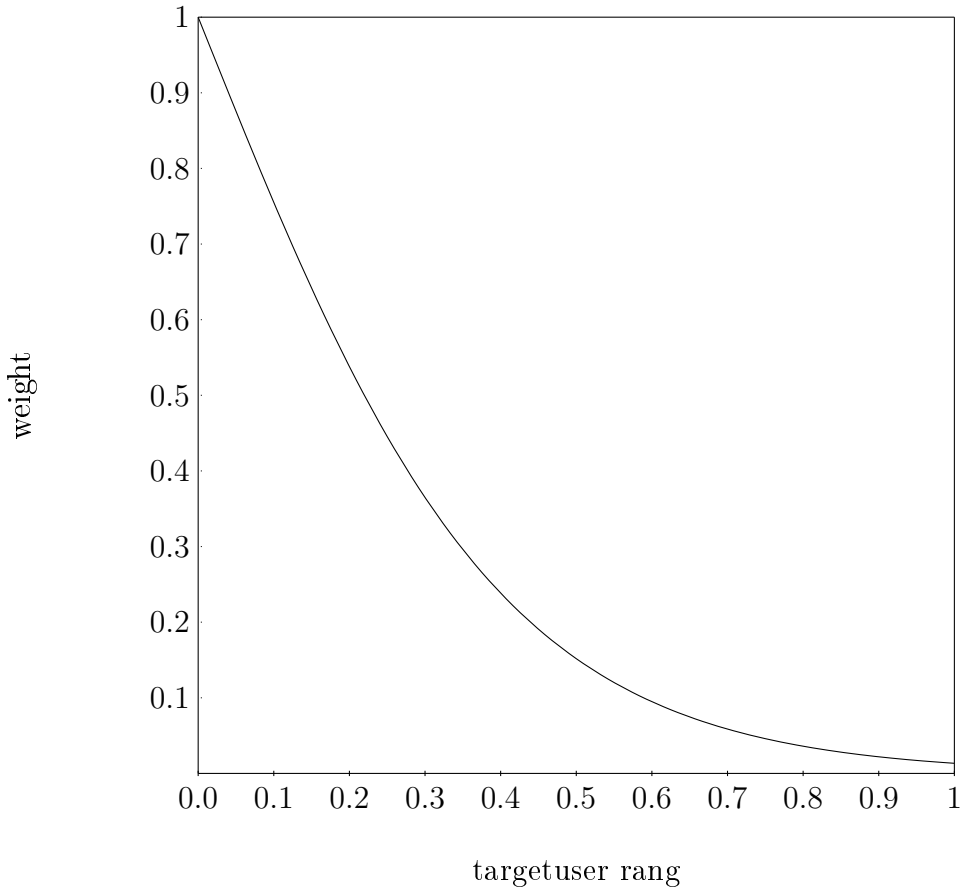

Abbildung 6.4: Bewertung des normalisierten internen Ranges des Zielbenutzers

Zum anpassen der Funktion auf den Rang des Zielbenutzer wurde der Rank mit $r(feed) = \frac{position(feed)}{countAllFeeds}$ normalisiert. Weiterhin wurde die Funktion invertiert um bei hohen Rängen eine hohe Wahrscheinlichkeit und bei geringen eine geringe Wahrscheinlichkeit zu erzielen.

$$fb(P_u,d_i) = \frac{2}{1 + e^{5*Score(P_u,d_i)}} \tag{6.12}$$

Abbildung 6.4 visualisiert die Funktion 6.12 der Berechnung der Wahrscheinlichkeit zur Auswahl der Feeds bzw. Quellen durch den Zielnutzer. Anhand dieser Funktion wählt der Zielbenutzer durch einen zufälligen gleichverteiltem Schwellwert die Feeds bzw. Quellen aus.

6.5 Vergleich zu anderen Empfehlungssystemen

Für das kaskadierte Empfehlungssystem wurden drei weitere Systeme getestet. Zum einen das System von NewsDude [26], ein inhaltsbasiertes System mit Kosinusmetrik unter Verwendung von Tf-Idf und ein kollaboratives mit gewichteter Summe und Kosinusähnlichkeitsmaß .

6.5.1 Newsdude

Das Original von NewsDude beinhaltet zum einen das unter 5.2.1 beschriebene Benutzermodell. Unter Verwendung eines inhaltsbasierten Empfehlungssystems wird dabei ein *Short-term*- und ein Long-term-Benutzermodell angelegt. Um Empfehlungen zu treffen wird dazu erst nach ähnlichen Dokumenten im *Short-term*-Nutzerprofil geschaut. Werden dort keine Empfehlungselemente mit ausreichender Ähnlichkeit gefunden wird im Long-term-Nutzermodell mit einer Kategorisierung „Interessant“ oder „nicht Interessant“ verwendet. Ein Profil besteht aus vom Nutzer bewerteten Texten die im Profil zu Termen transformiert abgelegt werden. Über das Kosinus-Ähnlichkeit werden dann ähnliche Dokumente bestimmt. Das *Long-term*-Nutzerprofil basiert auf einem Native Bayes-Klassfikator der anhand der Klassen „*Interessant*“ und „ *nicht Interessant*“ bei einer Übereinstimmung von mindestens drei Wörtern die betreffende Klasse zuweist.
NewsDude verwendet zum Empfehlen von einzelnen Artikeln keine Kanäle wie es durch die Anforderung 4 gefordert ist, sondern benutzt den gesamten Pool an gesammelten Nachrichten. Um in der Simulation mit den anderen Systemen vergleichbar zu sein wurde die Funktionsweise des *Long-term*-Profils für die Empfehlung der neuen Quellen mit einem inhaltsbasierten Ansatz erweitert. Quellen werden als Termvektoren mit Tf-Idf-Gewichtung formuliert und dem Kosinus-Ähnlichkeitsmaß zu bereits abonnierten Quellen verglichen und empfohlen. Zusammengefasst wird die Rangordnung der Feeds und der Quellen über das Benutzerprofil mit dem inhaltsbasierten System umgesetzt.

6.5.2 Inhaltsbasiertes mit Kosinusmetrik

Als Baseline-Empfehlungssystem wird ein inhaltsbasiertes System verwendet. Dies hat keine Unterteilung zwischen der Rangordnung der Feeds und den Quellen. Das Baseline-Verfahren erstellt wie bei dem NewsDude-Ansatz die Quellen als Termvektoren mit Tf-Idf-Gewichtung auf und vergleicht ähnliche Quellen anhand des Kosinusmaßes. Zur Rangordnung in der Listenordnung werden die Feeds absteigend nach dem Datum sortiert.

6.5.3 Kollaboratives

Als letztes wurde ein kollaboratives System implementiert. Auch dieses empfiehlt dem Zielbenutzer nur Quellen und sortiert die Feeds nach Datum absteigend für den Nutzer. Für die Simulation wurden zu jeder Kategorie Zielbenutzer erstellt. Ausgehend von der Entwicklung der Zielbenutzer abonnieren diese nach ihren Präferenzen neue Quellen. Mit dem Bernoullischen Model, ein Nutzer hat eine Quelle abonniert oder nicht, einer gewichteten Summe und dem Kosinus-Ähnlichkeitsmaß es werden die Ähnlichkeiten zu den Benutzer erzeugt.

6.6 Ergebnisse

Folgend werden die Ergebnisse der Simulationen zu den einzelnen System gegeneinander in Bezug gesetzt. Dazu wurden das inhaltsbasiert System als Baselinesystem für NewsDude und ein kollaboratives System als Vergleich für das kaskadierte System implementiert. Der ersten Teil konzentriert sich auf die Entwicklung der Nutzermodell zum erkunden der Quellen. Dies wurde mit dem Precision / Recall gemessen. Um einen besseren Vergleich zu erzielen wurde zu beiden Systemen der *F-Measure* berechnet [41]. Das F-Measuere verbindet Precision und Recall zu einem Wert und ermöglicht so einen besseren Vergleich zwischen zwei Empfehlungssystemen.

$$F_1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{6.13}$$

Der zweite Teil evaluiert die Rangordnung der Feeds innerhalb der zweiten Kaskade. Dazu wurden die relativen Kriterien zur Rangordnung mit der Ndpm, Spearman, dem Average-Document-Score und der Accouracy getestet.

6.6.1 Precision / Recall

Zur Bestimmung der Leistung der ersten Kaskade wurde der Precision / Recall der Empfehlungen der RSS-Quellen simuliert. Von beiden System wurden diese über 30 Runden berechnet und anschließend verglichen. Die kollaborativen Ansätze wurden mit 40 Nutzern getestet.

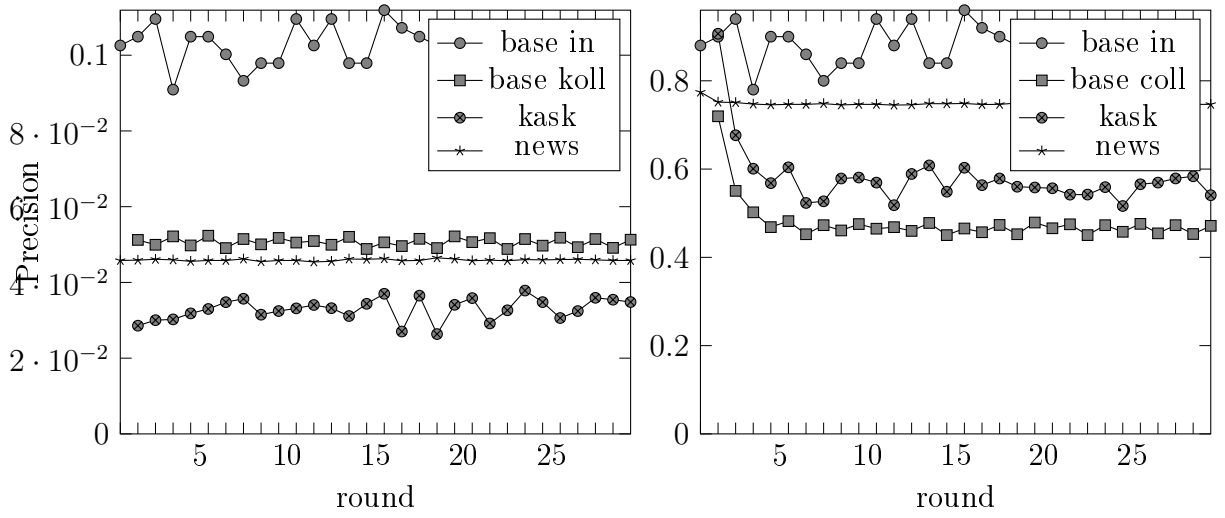

Abbildung 6.5: Recall (links) und Precision (rechts) des kaskadierten System und der NewsDude mit den jeweiligen Baseline-Systemen

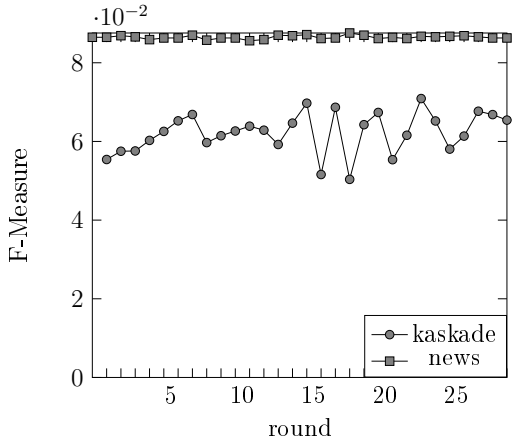

Abbildung 6.6: F-Measure NewsDude und kaskadiertes System

Abbildung 6.5 zeigt zum einen links den Recall und rechts die Precision der einzelnen Systeme. Bei der Precision ist das inhaltsbasierte Baselinesystem am Besten mit Werten zwischen

0.8 bis 0.9. Folgend darauf liegt das System von NewsDude mit 0.8 bis 0.7 dich gefolgt des Kaskadierten 0.7 bis 0.5. Am schlechtesten ist das kollaborative Baselinesystem mit rund 0.5. Der Recall (links) liegt bei dem Kaskadierten System zwischen 0.02 und 0.04, für das kollaborative bei 0.05, bei dem inhaltsbasierten bei 0.05 und 0.11 und bei dem NewDudesystem bei 0.05 und 0.11. Der Versuch wurde mit 40 Zielbenutzern getestet. In jeder Runde wurden dem Zielbenutzer drei Quellen empfohlen. Zur Darstellung wurde der Mittelwert der Nutzer gebildet. Kurz nach dem Anfang ist ein Abfall der Precision zu erkennen. Dies ist durch die erste Wahl des Zielbenutzer zu seiner Quelle zu erklären. Da der Zielbenutzer nur ein Quelle ausgewählt hat wird ihm ein Großteil an relevanten Dokumenten empfohlen. Nach Hinzunahme von weiteren Quellen durch andere Zielbenutzer fällt die Precision und pendelt sich für das kaskadierte System bei 0.6 ein.
Im F-Measere 6.6 ist noch einmal deutlich der Unterschied zwischen den beiden Systemen innerhalb des Precision und Recalls zu sehen. Der F-Measure liegt bei NewsDude ungefähr 0.17 wogegen der Wert für das Kaskadierte System bei 0.1 liegt.

6.6.2 Lernrate

Zur Betrachtung der Lernrate wurde für das kaskadierte und einfache kollaborative System mit 40 Zielnutzern und das System von NewsDude mit dem inhaltsbasierten System über 30 Runden durchgeführt. Die Ergebnisse der Systeme wurden über alle vierzig Nutzer gemittelt.

Accouracy

Abbildung 6.7 zeigt die Entwicklung der Accouracy zwischen dem kaskadierten Benutzerprofil (links) und des Benutzerprofils von NewsDude (rechts) mit den jeweiligen Baseline-Systemen.

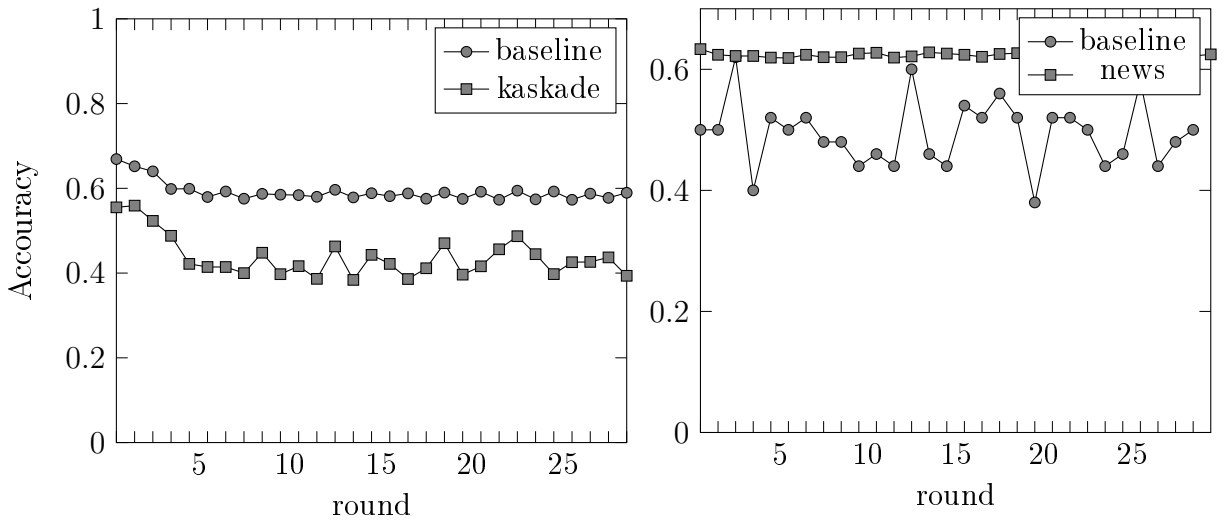

Abbildung 6.7: Accouracy des kaskadierten Systems (links) und NewsDude (rechts) mit den jeweiligen Baseline-Systemen

Zu erkennen ist, dass bei dem kaskadierten System die absolute Accuracy zum Vergleichsystem abnimmt. Dies deckt sich mit der Entwicklung zur Precision. Nach rund 5 Runden pendelt sich die Genauigkeit bei 0.6 ein. Anders bei dem inhaltsbasierten und NewsDude-System. Hier liegt die Accuracy bei ungefähr 0.5.

Normalized Distance-based Performance Measure

Die Ergebnisse der Simulation zur Normalized distance performance measure ist in Abbildung 6.8 dargestellt. Die Diagramme sind wieder für das kaskadierte System (links) und NewsDude (rechts) mit den jeweiligen Baseline-Systemen über die Runden aufgetragen. Über die Runden entwickelt der ndpm des kaskadierten Systems zwischen 0.7 und 0.6 und besitzt eine kleine Abweichung zum Referenzsystem. Somit ist das relative Ranking mit der zweiten Kaskade besser als das Referenzsystem. Aber im Vergleich ähnlich zur Entwicklung des NewsDude-Systems. Beide Systeme ranken die Feeds nach gleichen Verfahrensweise. Das kann ein Grund für den ähnlichen Verlauf sein.

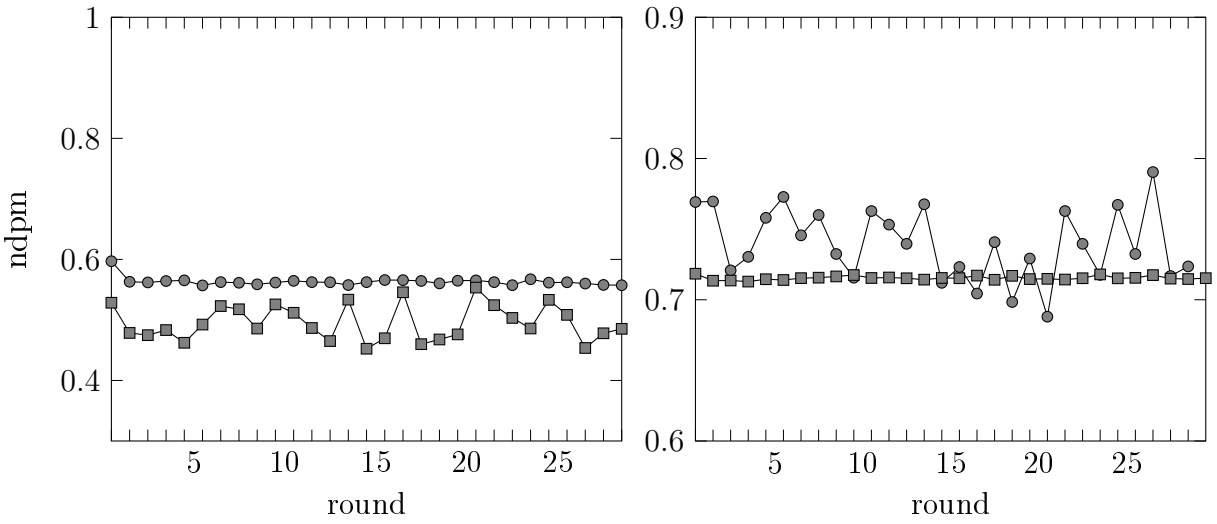

Abbildung 6.8: Ndpm Ergebnisse von kaskadierten (links) und NewsDude (rechts) mit den jeweiligen Baseline-Systemen

Spearman rank-order correlation coefficient

Abbildung 6.9 zeigt die Entwicklung des Spearman-Koeffizienten. Die Entwicklung des Spearman-Koeffizienten stüzt die Entwicklung des Ndpm. Auch hier liegen das kaskadierte System und das NewsDude-System im gleichen Wertebereich.

Average Document Score

In Abbildung 6.10 sind die Ergebnisse des Average Document Score abgebildet. Hier ist ein groös er Unterschied zwischen den Baseline-System zu erkennen. Der Average Document Score ist von den Baseline Systemen sehr niedrig über die Runden bewertet worden und bewegt sich beim kollaborativen Baselinesystem bei rund 0.1 und beim inhaltsbasierten Baselinesystem bei rund 0.05. Zurück zu führen ist das auf die Unterschiedliche Bewertungsweise zwischen Ziel- und Systembenutzer. Der Systembenutzer ordnet die Artikel nach ihrem Datum an. Der Zielbenutzer belegt seine Präferenz aber nach den Inhalt der Artikel. Anhand dieser unterschiedlichen Bewertung können die niedrigen Werte erkärt werden.

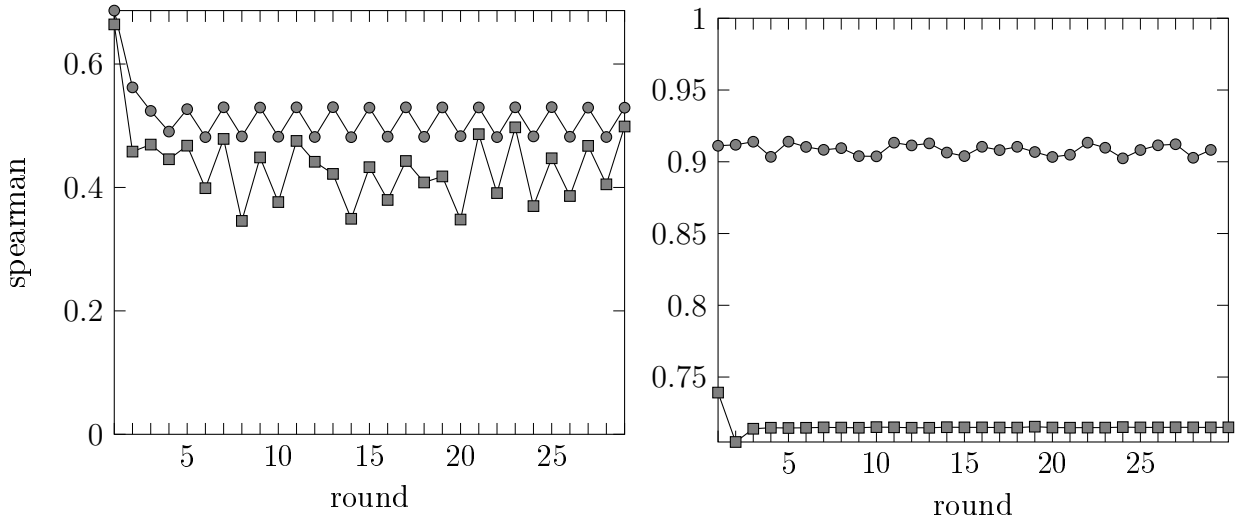

Abbildung 6.9: Spearman rank Ergebnisse von kaskadierten (links) und NewsDude (rechts) mit den jeweiligen Baseline-Systemen

6.6.3 Diskussion

Ein Teil der Ergebnisse liefern einen eine Bestätigung der theoretischen Annahmen. Durch die geringere Precision ist es für den Nutzer möglich neue Artikel zu entdecken und der Anforderungspunkt 2 kann damit abgedeckt werden. Auch liefert die Ndpm und der Spearmankoeffizient einen Beleg für die Bessere Anordnung der Empfehlungsliste. Durch die Erweiterung des kollaborativen Ansatzes konnte der Punkt zur Entdeckung neuer News, gegenüber dem inhaltsbasierten Ansatz von NewsDude den Nutzer unterstützen neue RSS-Quellen zu entdecken. Die Evaluation in dieser Arbeit wurde auf Grundlage einer Simulation erstellt. Der Vorteil der Simulation ist die Reproduzierbarkeit. Für den Zielbenutzer wurden aber sehr starke Annahmen getroffen. Diese können nur einen ersten Eindruck zu dem kaskadierten System geben. Besonders auf Grundlage des Ranking des Zielbenutzer können die Experimentellen Werte von einem Realeinsatz abweichen. Als Ausblick sollte dazu weiterführend eine Studie durchgeführt werden.

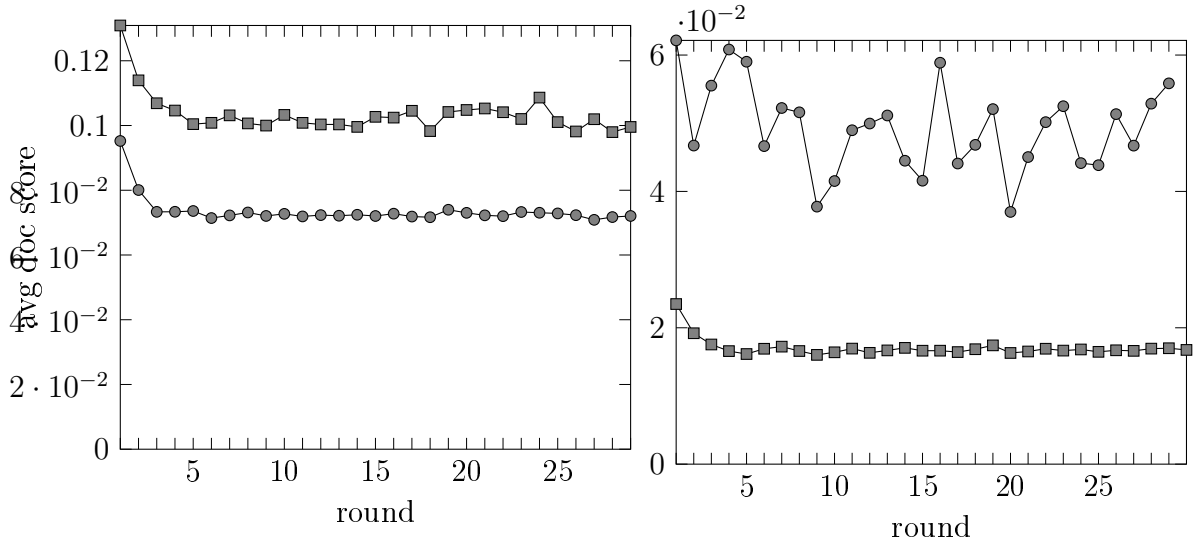

Abbildung 6.10: Average document score Ergebnisse von kaskadierten (links) und NewsDude (rechts) mit den jeweiligen Baseline-Systemen

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[1] *Introduction to Information Retrieval*
von Christopher D. Manning, Prabhakar Raghavan and Hinrich Schutze in Cambridge University Pres (2009)

[2] *Information Retrieval from Documents: A Survey*
von M. Mitra und B.B. Chaudhuri Indian Statistical Institute, Calcutta (2000)

[3] *A Survey of Text Similarity Approaches*
von Wael H. Gomaa und Aly A. Fahmy (2013)

[4] *„Is This Document Relevant? . . . Probably": A Survey of Probabilistic Models in Information Retrieval*
von Fabio Crestani, Mounia Lalmas, Cornelis J. van Rijsbergen und Iain Campbell, University of Glasgow (1998)

[5] *Term weighting approaches in automatic Text retrieval*
von Gerald Salton und Chris Buckly,
Department of Computer science Cornell University Ithaca, New York
November (1989)

[6] *Toward the next Generation of Recommender Systems: A survery of the state-of-the-art and possible extensions*
von Gediminas Adomavicius und Alexander Tuzhilin (2005)

[7] *Eigentaste: A Constant Time Collaborative Filtering Algorithm*
Ken Goldberg and Theresa Roeder and Dhruv Gupta and Chris Perkins
IEOR and EECS Departments University of California, Berkeley (2000)

[8] *Empfehlungssysteme - Grundlagen, Konzepte und Lösungen*
von Andre Klahold (2009)

[9] *Recommender systems survey*
von J.Bobadilla, F.Ortega, A. Hernando, A. Gutierrez (2013)

[10] *Machine Learning*
von Mitchell, T. McGraw-Hill, New York (1997)

[11] *Hybrid Recommender Systems: Survey and Experiments*
von Robin Burke
California State University, Fullerton
Department of Information Systems and Decision Sciences (2002)

[12] *Using collaborative filtering to weave an information tapestry*
D. Goldberg, D. Nichols, B. Oki, and D. Terry (1992)
Communications of the Association of Computing Machinery, 35(12):61-70, (1992)

[13] *Grouplens: Collaborative filtering for usenet news.*
von Joseph Konstan, Brad Miller, David Maltz, Jon Herlocker, Lee Gordon, and John Riedl.
Communications of the ACM, March (1997)

[14] *Adaptivität und Benutzermodellierung in interaktiven Softwaresystemen*
von Alfred Kobsa
AG Wissensbasierte Informationssysteme Informationswissenschaft, Universität Konstanz (1993)

[15] *Mediation of user models for enhanced personalization in recommender systems*
von Shlomo Berkovsky, Tsvi Kuflik, Francesco Ricci
Department of Computer Science and Software Engineering, University of Melbourne, Melbourne
University of Haifa, Haifa, Israel, (2007)

[16] *Generic User Modeling Systems*
von Alfred Kobsa
Department of Information and Computer Science, University of California, Irvine (2000)

[17] *Addressing Uncertainty in Implicit Preferences*
von Sandra Gadanho und Nicolas Lhuillier
Applications Research Motorola Labs Basingstoke, U.K. (2007)

[18] *Modeling User Preferences in Recommender Systems: A Classification Framework for Explicit and Implicit User Feedback* von Gawesh Jawaheer, Peter Weller und Patty Kostkova
University College London (UCL) (2014)

[19] *An Economic Model of User Rating in an Online Recommender System* von F. Maxwell Harper, Xin Li, Yan Chen, and Joseph A. Konstan
University of Minnesota, Minneapolis, MN, 55406 USA
University of Michigan, Ann Arbor, MI, 48109 USA (2014)

[20] *I like it... I like it not: Evaluating User Ratings Noise in Recommender Systems*
von Xavier Amatriain, Josep M. Pujol, and Nuria Oliver
Telefonica Research (2009)

[21] *Rate it Again: Increasing Recommendation Accuracy by User re-Rating*
von Xavier Amatriain, Josep M. Pujol, Nuria Oliver und Nava Tintarev
Telefonica Research (2009)

[22] *A Study on Recommendation Features for an RSS Reader*
von Cansheng Ji und Jingyu Zhou (2010)

[23] *Category-Based Filtering and User Stereotype Cases to Reduce the Latency Problem in Recommender Systems*
von Mikael Sollenborn and Peter Funk

Mälardalen University Department of Computer Science and Engineering Västeras, Sweden (2002)

[24] *Content-based Recommender Systems: State of the Art and Trends*
von Pasquale Lops, Marco de Gemmis and Giovanni Semeraro (2011)

[25] *Comparing two approaches of generating interest profiles for information filtering: Interese inferred from typical user actions versus rating of content*
von Junlian Zhang and Javed Mostafa (2002)

[26] *A Hybrid User Model for News Story Classification*
von Daniel Billsus and Michael J. Pazzani (1999)

[27] *A Learning agent for wireless news access*
von Daniel Billsus, Michael J.Pazzani und James Chen (2000)

[28] *User Modeling for Adaptive News Access*
von Daniel Billsus, Michael J.Pazzani (2000)

[29] *The Adaptive Web. Lecture Notes in Computer Science*
von Brusilovsky, P., Kobsa, A., Nejdl, W. (2007)

[30] *Information Filtering Based on User Behavior Analysis and Best Match Text Retrieval*
von Masahiro Morita und Yoichi Shinoda (1994)

[31] *User Modeling. In: Handbook of human factors in Web design*
von Johnson, A.; Taatgen, N.Lawrence Erlbaum
Associates 424-439 (2005)

[32] *Open User Profiles for Adaptive News Systems: Help of Harm*
von Jae-wook Ahn, Peter Brusilovsky, Jonathan Grady Daqing He und Sue Yeon Syn
School of Information Science, University of Pittsburgh (2007)

[33] *Implicit Feedback for Inferring User Preference: A Bibliography*
von Diane Kelly und Jaime Teevan
Kapitel Classification of implicite feedback techniques (2003)

[34] *Methods for determining the order of an autoregressive-moving average process: A survey*
von Jan G. de Gooijer, B Abraham, A Gould und Lecily Robinson
Department of Economic Statistic , University of Amsterdam (1985)

[35] *Query Chains: Learning to Rank from Implicit Feedback*
von Filip Radlinski und Thorsten Joachims (2005)

[36] Time-aware recommender systems: a comproehensive survey and analysis of existing evaluation protocols
von Pedro G. Campos Fernando Diez und Ivan Cantador (2013)

[37] *Spezifikation RSS 2.0* von Harvard Law
http://cyber.law.harvard.edu/rss/rss.html, Stand 6.2016

[38] *Dynamic Modeling and Learning User Profile In Personalized News Agent*
von Dwi H. Widyantoro, Reza Langari, John Yen (1999)

[39] *Measuring Retrieval Effectiveness Based on User Preference of Documents* von Y.Y. Yao Department of Mathematical Sciences (1995)

[40] *Evaluating Collaborative Filtering Recommender Systems* von Jonathan L. Hocker und Joseph A.Konstan, Loren G.Terveen und John T. Riedl
Oregon State University und University of Minnesota (2004)

[41] *Analysis of Recommendation Algorithms for E-Commerce*
von Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl
GroupLens Research Group / Army HPC Research Center Department of Computer Science and Engineering University of Minnesota (2000)