

Tim Hussein und Werner Gaulke

Hybride, kontext-sensitive Generierung von Produktempfehlungen

Hybrid, context-aware generation of product recommendations

Empfehlungssysteme_Context-Awareness_E-Commerce

Zusammenfassung. In diesem Beitrag stellen wir mit Hybreed RecFlows ein modulares Framework zur Generierung von (Produkt) Empfehlungen vor. RecFlows (Kurzform für Recommendation Workflows) stellt eine Reihe etablierter Algorithmen aus dem Bereich Recommender Systems zur Verfügung sowie einen Workflow-Mechanismus, um aus diesen Algorithmen flexibel hybride Recommender zu erstellen. Darüber hinaus werden unterschiedliche Sensoren bereitgestellt, um Informationen aus verschiedenen Quellen in den Empfehlungsprozess mit einfließen zu lassen. Insbesondere werden Sensoren zur Kontext-Erfassung (z.B. der aktuelle Ort des Nutzers anhand seiner IP-Ad-

resse) implementiert. So ist es möglich, mit Hilfe von RecFlows hybride, kontextsensitive Empfehlungen zu generieren.

Summary. In this contribution, we introduce Hybreed RecFlows, a modular framework for contextaware product recommendations. RecFlows (short for recommendation workflows) contains a set of well-established recommendation algorithms as well as a workflow-mechanism for generating modular hybrid recommenders. Beyond that, Hybreed provides sensors for integrating context information (for instance the user's current location). By doing so, Hybreed RecFlows enables context-aware recommendation generation.

1. Einleitung

Mit der Hilfe Computergenerierter Empfehlungen können unüberschaubar große Mengen an Produkten, Webseiten, Diensten oder anderen Artefakten (im Folgenden generell „Items“ genannt) für den Nutzer besser zugänglich gemacht werden. Empfehlungssysteme (Recommender Systems) unterstützen den Nutzer, indem sie bestimmte Inhalte aus dem Gesamtangebot herausfiltern, die mit hoher Wahrscheinlichkeit von großem Interesse für ihn sind. Dabei hängen die Items, für die sich ein Nutzer interessiert, oft von der gegebenen Situation ab wie zum Beispiel dem aktuellen Standort, der Jahreszeit oder vielen weiteren Faktoren.

Kontextadaptive Empfehlungssysteme nutzen den Anwendungskontext, um damit den Empfehlungsgenerierungsprozess zu verbessern. Für das Konzept des Anwendungskontextes gibt es bisher keine einheitliche formale Definition.

Vielmehr existiert eine ganze Fülle von über 150 Definitionen aus verschiedenen Bereichen. Eine der bekanntesten Definitionen stammt von (Dey und Abowd, 2000), die ein breitgefasstes Verständnis von Kontextinformationen vermittelt. Dort wird Kontext definiert als „any information that can be used to characterize the situation of an entity (...)“.

Somit wird dem Designer die Entscheidung überlassen, welche Kontextinformationen für relevant erachtet werden. In diesem Artikel möchten die Autoren eine Technik vorstellen, welche sowohl Kontextinformationen aus der Interaktion mit dem System, als auch Informationen aus externen Quellen zum Beispiel die Zeit nutzt, um diese in einem Empfehlungsprozess einzusetzen.

Im weiteren Verlauf des Artikels werden zunächst vorhandene Ansätze sowie deren Einschränkungen vorgestellt. Aufbauend auf diesen Überblick stellen wir mit dem Hybreed Framework einen Ansatz für hybride Kontextadaptive Emp-

fehlungssysteme vor. Das Konzept wird anhand einer prototypischen Implementierung verdeutlicht.

2. Recommender Systems

Recommender Systems (Empfehlungssysteme) sind seit den 1990er Jahren als eigenständiges Forschungsgebiet etabliert. Ihre Wurzeln liegen in verschiedenen Disziplinen, unter anderem in der Kognitionsforschung und dem Information Retrieval. Empfehlungssysteme zielen darauf ab, dem Benutzer dabei zu helfen, geeignete Inhalte aus oft unüberschaubar großen Angeboten, beispielsweise in Webshops, zu empfehlen. Empfohlene Inhalte sind meist kommerzielle Produkte, die dem Kunden zum Kauf angeboten werden, können aber genauso gut Nachrichtenartikel, digitale Musik, Bilder oder Links auf Webseiten sein. Die dabei verwendeten Technologien können grob unterteilt werden in kollaborative und inhaltsbasierte Ansätze sowie Hybridsys-

teme. Eine Übersicht findet sich beispielsweise in (Candillier et al., 2009).

Typischer Aufbau eines Recommenders

Bild 1 zeigt den typischen Aufbau eines Recommenders (nach Burke, 2002). Neben dem eigentlichen Recommender-Algorithmus werden üblicherweise Hintergrundinformationen sowie ein jeweiliger Input benötigt, um Empfehlungen zu generieren. Die Hintergrundinformationen können dabei ganz unterschiedlicher Natur sein: Bei Collaborative Filtering Algorithmen (Sarwar et al., 2001) ist dies typischerweise eine User-Item-Matrix mit den Bewertungen bzw. Käufen aller User; bei inhaltsbasierten Recommendern beispielsweise Beschreibungen der einzelnen Produkte, beispielsweise in Form von Tags oder semantischen Modellen. Auch der jeweilige Input unterscheidet sich je nach benutztem Verfahren; beispielsweise könnten das zuletzt angesehene Produkt oder die Liste aller bisherigen Einkäufe des Nutzers als Eingabe genutzt werden.

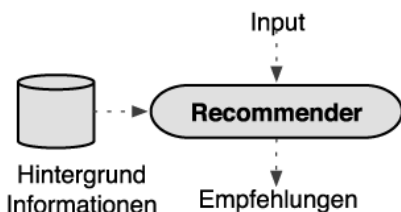


Bild 1: Typischer Aufbau eines Recommenders nach Burke (2002). Hintergrund-Informationen ist beispielsweise eine Datenbank mit den Bewertungen und Käufen aller Nutzer; Input beispielsweise das aktuell von einem Nutzer angeklickte Item, zu dem Empfehlungen nun generiert werden sollen.

2.1 Collaborative Filtering

Collaborative Filtering (CF) Methoden gelten als die am häufigsten implementierten Empfehlungs-Algorithmen. Grundsätzlich basieren sie auf der Annahme, dass Personen genau dann ähnliche sind, wenn sie Items ähnlich bewerten. Items wiederum werden als ähnlich betrachtet, wenn sie von Personen mit ähnlichem Geschmack ähnlich bewertet werden.

Man unterscheidet hier zwischen User-based und Itembased Collaborative Filtering. Bei ersterem werden die Nutzer miteinander verglichen: Aus ihren Bewertungen bzw. Käufen wird ein Vektor

wie in der folgenden Tabelle erstellt: Der Benutzer U_2 könnte somit durch seinen Bewertungsvektor $V_2 = (0;1;5;2)$ repräsentiert und die Ähnlichkeit zu einem anderen Nutzer U_n durch ein Vergleich mit dessen Vektor $V_n = (1;1;2;4)$ gemessen werden. Übliche Ähnlichkeitsmaße sind in diesem Fall der Kosinus des Winkels zwischen den beiden Vektoren oder eine Korrelationsmessung nach Pearson.

Aus den Bewertungen der zu U_2 ähnlichsten Nutzer können nun fiktive Bewertungen errechnet werden (beispielsweise per gewichtetem Mittelwert), die einer tatsächlichen Bewertung durch U_2 wahrscheinlich sehr nahe kommen. Die Items mit den höchsten fiktiven Bewertungen sind Grundlage für Empfehlungen an den Benutzer.

Bezogen auf Bild 1 stellt die Präferenzmatrix hier die Hintergrunddaten zur Verfügung, während der aktuelle Benutzer als Eingabe fungiert. Beim Itembased Collaborative Filtering hingegen wird untersucht, welche Items sich gleichzeitig in den Warenkörben der Nutzer befinden und daraus auf eine gewisse Ähnlichkeit geschlossen. Bekannt geworden ist dieses Verfahren insbesondere durch die Verwendung auf Amazon.com („Kunden, die X gekauft haben, haben auch Y gekauft“). Ein großer Vorteil ist hierbei, dass sich Ähnlichkeiten teilweise vorberechnen lassen, was bei sehr großen Angeboten ein immenser Vorteil sein kann.

Auch daher wird es in Webshops häufig eingesetzt. Die Ideen gehen allerdings nicht auf Amazon zurück, sondern auf (Goldberg et al., 1992. Userbased CF) bzw. (Sarwar et al., 2001. Itembased CF). Collaborative Filtering Methoden zeigen allerdings dann Schwächen, wenn erst wenige Bewertungen abgegeben bzw. Käufe getätigt wurden („Cold Start Problem“) oder die in Tabelle 1 gezeigte Matrix sehr spärlich besetzt ist („Sparsity Problem“). Auch können derartige Systeme unter Umständen Mainstream-Items gegenüber Nischenprodukten bevorzugen (Stichwort „Long Tail“ Produkte¹). Unter anderem aus diesen Gründen wird es häufig mit anderen Ansätzen kombiniert.

	U_1	U_2	U_3	U_n
I_1	3	0	0	1
I_2	4	1	3	1
I_3	0	5	0	2
I_n	0	2	4	4

Tabelle 1: Beispiel für eine User-Item-Matrix beim Collaborative Filtering. Für jeden User (U) wird notiert, wie er ein bestimmtes Item (I) bewertet hat. In diesem Fall reicht die Bewertungsbreite von 1–5. 0 bedeutet, dass keine Bewertung abgegeben wurde.

2.2 Inhaltsbasierte Recommender

Inhaltsbasierte („content-based“) Recommender nutzen Attributbeschreibungen der Items aus, um Ähnlichkeiten zu ermitteln und auf dieser Basis Empfehlungen zu generieren. Beispiele für derartige Attribute wären PS-Zahl und Verbrauch bei PKWs oder Zoom und Auflösung bei Digitalkameras. Nutzer-Aktionen werden üblicherweise daraufhin untersucht, an welchen Attributen Interesse besteht. Es existiert eine Vielzahl an Techniken, mit denen hier gearbeitet werden kann; so kommen häufig zum Beispiel Neuronale Netze, Entscheidungsbäume oder Vektorbasierte Verfahren zum Einsatz.

Eine weitere Technik, die häufig angewendet wird und mit der wir gute Resultate im Rahmen unserer Forschung erzielt haben, ist Spreading Activation. Auch hier werden inhaltliche Attribute ausgenutzt, um Empfehlungen generieren zu können: In diesem Fall werden die Items und ihre Eigenschaften auf einen Graphen mit Knoten und Kanten abgebildet, wobei die Kanten die jeweiligen Beziehungen charakterisieren (Beispielsweise würden die DVDs „Toy Story“ und „Findet Nemo“ beide über eine Beziehung zum Knoten „Animationsfilm“ verfügen).

Spreading Activation nun bezeichnet eine Menge ähnlicher Algorithmen, die ursprünglich aus dem Bereich der Künstlichen Intelligenz und der Kognitionsforschung stammen. Im Allgemeinen geht man dabei von einer bestimmten Menge von Items aus, die miteinander

¹ Chris Anderson hat in seinem 2006 erschienenen Buch „The Long Tail“ anschaulich beschrieben, wie sich Märkte durch elektronische (=günstige) Vertriebswege radikal ändern und Nischenprodukte für Betreiber von Shops und Marktplätzen äußerst rentabel werden.

über Relationen verbunden sind. In diesem semantischen Netz können nun den einzelnen Items Gewichtungen zugewiesen werden, welche über die Relationen an verwandte Items propagiert werden. Auf diese Weise können verwandte Items identifiziert werden.

Will man inhaltsbasierte Empfehlungen mit kollaborativen Ansätzen vergleichen, fällt auf, dass ihre Vor- und Nachteile weitgehend komplementär zueinander sind: Während inhaltsbasierte Empfehlungssysteme von Anfang an konstant gute Empfehlungen liefern, ist man bei kollaborativen Ansätzen auf eine gewisse Nutzerbasis und Bereitschaft zur Kollaboration angewiesen. Im Umkehrschluss kommen diese Ansätze aber auch mit deutlich weniger Modellierungsaufwand aus und sind häufig besser skalierbar. Daher erscheint es nur logisch, Hybridsysteme in Betracht zu ziehen, die die jeweiligen Vorteile kombinieren.

2.3 Kontext-sensitive und hybride Recommender

In den letzten Jahren sind häufig Ansätze bezüglich der Kombinierbarkeit von Recommendern untersucht worden (Burke, 2007) sowie die Möglichkeiten zur Integration von Kontextinformationen in den Empfehlungsprozess (Baldauf et al. 2007). Einen Ansatz, um Kontext in Form von ausgewählten Tags eines Benutzers, auszunutzen stellen (Shepitsen et al., 2008) vor.

Die gewählten Tags werden verwendet, um aus Menge der möglichen Empfehlungen vorab Teilbereiche zu filtern. Ein inhaltsbasiertes Modell für Kontextadaptive Empfehlungen wurde (Kim und Kwon, 2007) eingeführt. In vier getrennten Modellen werden Produkte, Aufenthaltsorte, Bestellhistorie und Benutzervorlieben gespeichert. Aus der Menge der Produkte werden mit Hilfe der Benutzervorlieben und der Bestellhistorie relevante Produkte ausgewählt. Einen Ansatz um vorhandene Algorithmen für kollaboratives Filtern um Kontext zu erweitern zeigen (Adomavicius et al., 2005).

Zu den gegebenen Benutzern und deren Bewertung werden Kontextinformationen als zusätzliche Dimension hinzugefügt. Empfehlungen werden hier erzeugt, indem die Bewertungen der Benutzer unter dem vorhandenen Kontext

neu bewertet werden. Die vorgestellten Systeme zeigen exemplarisch, dass Kontext in Empfehlungssysteme integriert werden kann, um diese zu verbessern. Jedoch handelt es sich um spezielle Lösungen, die nicht ohne weiteres auf alle Anwendungsfelder übertragen werden können. Im Folgenden wird ein Rahmenwerk vorgestellt, mit dem dies auf generischem Wege erreicht werden kann. Über die Integration von Kontext hinaus bietet die vorgestellte Lösung zusätzlich die Möglichkeit Empfehlungsalgorithmen flexibel miteinander zu kombinieren um leistungsstarke Hybridsysteme zu erhalten.

3. Das Hybreed Framework

Beim Hybreed Framework handelt es sich um eine modulare Plattform, welche unter anderem Kontextinformationen für den Einsatz in hybriden Empfehlungsprozessen nutzbar macht. Die durch Hybreed generierten Empfehlungen können in Endanwendungen integriert werden. Durch die Modularität sind vielfältige Einsatzszenarien möglich, da das Rahmenwerk alles nötige für die Generierung von Empfehlungen vorgibt, aber die zugrundeliegenden Daten und die Konfiguration des Systems von dem Entwickler der Endanwendung bestimmt werden können. Beispiele für den Einsatz sind zum Beispiel E-Commerce Systeme oder die Anzeige kontextabhängiger Empfehlungen oder Werbeeinblendungen auf Smartphones. Um für den Einsatzzweck die passende Konfiguration zu bilden, können die vorhandenen Komponenten beliebig zusammen gestellt werden.

3.1 Kontext-Erkennung mit Hilfe von Hybreed Sensors

Allgemein wird der Begriff „Kontext“ in der Informatikforschung oft als eine Liste von Faktoren angenommen, die die aktuelle Situation beschreiben. Dies können zum Beispiel Ort und Zeit sein, wie Sie von (Schilit et al., 1994) definiert werden. Auf einem eher abstrakten Level bewegt sich die Definition von (McCarthy, 1993) und (Dey und Abowd, 2000), welche keine expliziten Faktoren als Inhalt des Kontextes angeben. Neben diesen eher freien Ansätzen, gehen (Burrell, 2000) und

(Baldauf, 2007) im Feld des Ubiquitous bzw. Wearable computings den Weg, dass Faktoren vorab fest definiert und beschränkt sind. (Haake et al., 2010) stellen auf Basis der Definition von Dey und Abowd ein allgemeines Kontextmodell vor, welches für eine Anwendung benötigtes Kontext- und Adaptionswissen auf vier Ebenen modelliert.

Auf diesem Modell basiert auch Hybreed². Die relevanten Kontext-Informationen können demnach externe Daten enthalten sowie Informationen, die während der Verwendung des Systems entstehen. Beispiele für externe Informationen wären zum Beispiel Ort und Zeit, während bei der Systemverwendung Daten wie die Navigationshistorie oder Aufenthaltsdauern auf den einzelnen Sichten anfallen.

Um die vorgestellten Informationen nutzen zu können, wird das Hybreed Sensors Modul verwendet. Das Modul verwendet Sensoren um Kontextinformationen zu erfassen. Mit Sensoren werden spezialisierte Komponenten bezeichnet, die einzelne Werte messen und zur Verfügung stellen. Ein Sensor könnte zum Beispiel den aktuellen Standort bestimmen und dafür die aktuelle IP des Benutzers verwenden oder aber auf die Daten eines GPS Empfängers zurückgreifen.

Die erfassten Daten eines Sensors werden von diesem in eine einheitliche Repräsentation überführt, welche in ein gemeinsames Modell abgelegt werden. Im Fall vom Hybreed werden die Daten im RDF/OWL Format repräsentiert, welches geeignet ist sowohl einfache als auch komplexe Modelle zu verwalten. Das Format ist somit unabhängig von den gespeicherten Daten. Eine typische Ausgabe eines Sensors sieht wie folgt aus:

```
<rdf:Description
  rdf:about="location:current">
  <location:lat rdf:datatype="xsd:double">51.51</location:lat>
  <location:lng
    rdf:datatype="xsd:double">7.45
  </location:lng>
</rdf:Description>
```

In diesem Fall gibt der Sensor einfach die aktuellen Koordinaten als latitude (lat) und longitude (lng) zurück. Sensoren können Daten von angebundenen Hardware-Sensoren verwerten, wie zum

² Auf Details bezüglich der Modellierung wird an dieser Stelle verzichtet. Diese finden sich in (Haake et al., 2010).

Beispiel Temperatur oder GPS Empfänger. Daneben können externe Web Services als Informationsquelle dienen oder eigene Implementierungen erzeugt werden, die die gewünschten Kontextdaten liefern.

Die Ausführung mehrerer Sensoren wird durch eine zentrale Einheit, der Sensing Engine koordiniert. Diese führt die die einzelnen Sensoren aus, was durch ein standardisiertes Interface ermöglicht wird, dass alle Sensoren implementieren müssen. Nach der Ausführung werden die gelieferten Daten der Sensoren von der Sensing Engine in das Zustandsmodell eingefügt oder aktualisiert, sofern diese dort schon vorhanden waren (s. Bild 2). Die Ausgabe besteht letztendlich aus einem Modell mit allen Kontext Informationen der verwendeten Sensoren.

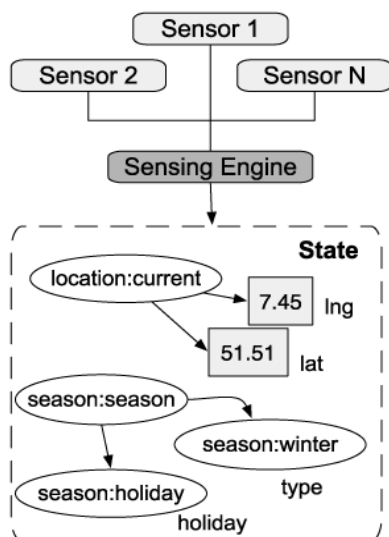


Bild 2: Die Sensing Engine aggregiert die einzelnen Sensor-Inputs zu einem gemeinsamen Zustand (State), der später den Recommendern zur Verfügung steht.

3.2 Kontext-sensitive Empfehlungen mit Hilfe von Hybreed RecFlows

Ziel von RecFlows ist es, etablierte Recommender nutzen zu können, sie um Eigenentwicklungen zu ergänzen und sie gegebenenfalls in Verbindung mit Kontext-Informationen (Ort, Zeit, etc.) zu komplexen Recommendation Workflows zu verknüpfen. Zu diesem Zweck wurde ein allgemeines Interface Hybreed Recommender kreiert, welches beliebige Recommender integrieren soll (s. Bild 3).

Als Datenquelle kann eine beliebige Implementierung gewählt werden, beispielsweise eine Datenbank-Anbindung zu einer Tabelle mit Produkt-Bewertun-

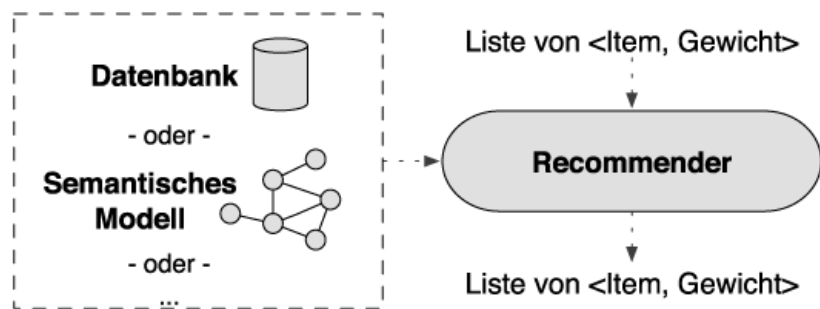


Bild 3: Ein Hybreed Recommender stellt ein Interface für beliebige Recommender-Algorithmen dar.

gen oder eine semantische Repräsentation von Items und ihrer Eigenschaften in Form einer Ontologie. Beides wurde in RecFlows bereits erfolgreich eingesetzt. Die in der jeweiligen Datenquelle abgebildeten Items bilden gleichzeitig die Menge der möglichen Empfehlungen. Wir werden später zeigen, wie diese Menge durch Kombination verschiedener Recommender schrittweise gefiltert oder erweitert werden kann.

Als Eingabe in den Recommendation-Prozess wird eine Liste von Items erwartet, zusammen mit einem jeweils optionalen Parameter „Gewicht“, welcher das Gewicht angibt, mit dem das jeweilige Item in den Empfehlungsprozess eingehen soll. Ergebnis dieses Prozesses ist wiederum eine Liste von Items mit ihrem jeweiligen Gewicht. In diesem Fall repräsentiert das Gewicht die Stärke der Empfehlung. Das Gewicht ist in beiden Fällen eine reelle Zahl zwischen 0 und 1. Wird keines angegeben, gilt automatisch der Wert 1.

Innerhalb des Recommenders findet gegebenenfalls eine Umkodierung der Eingaben statt, so dass vorhandene Recommendation-Bibliotheken genutzt werden können. Bild 4 zeigt, wie ein existierender User-based Collaborative

Filtering Recommender aus der Apache Mahout Bibliothek³ integriert wurde.

Aufbau eines RecFlows

Wie gezeigt sind die einzelnen Hybreed Recommender gleichförmig aufgebaut. Bild 5 zeigt, wie zwei Recommender hintereinander geschaltet werden können, um z. B. das Ergebnis eines Collaborative Filtering Prozesses mittels einer Spreading-Activation-Technik zu erweitern⁴.

Durch das Hybreed Recommender Interface ist sichergestellt, dass jede verwendete Hintergrund-Datenquelle in der Lage ist, sich auf eine übergebene Liste von Items

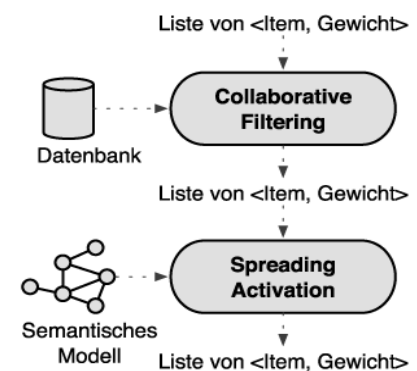


Bild 5: Die Ergebnisse eines Collaborative Filtering Recommenders.

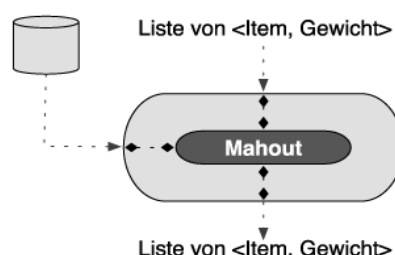


Bild 4: Dieses Beispiel zeigt wie existierende Recommender-Algorithmen eingebettet werden können. Innerhalb eines Hybreed Recommenders wird ein existierender Algorithmus ausgeführt. Input, Hintergrund-Daten und entstehende Recommendations müssen jeweils konvertiert werden (gestrichelte Linie).

zu reduzieren oder um diese zu erweitern. Auf diese Weise können die Empfehlungen auch die Menge der zu berücksichtigenden Items steuern (s. Bild 6).

Erweiterungen der jeweiligen Modelle geschehen analog; allerdings muss hierbei eine Ausgangs-Datenquelle existieren, aus der die entsprechenden Inhalte wieder hergestellt werden können. Das (Wieder) Hinzufügen von Inhalten eignet sich

³ <http://mahout.apache.org>

⁴ Spreading Activation dient im Wesentlichen dazu, semantisch verbundene Items zu identifizieren. Details finden sich beispielsweise in Hussein & Ziegler 2008.

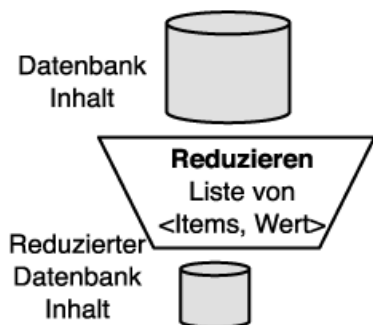


Bild 6: Die Inhalte einer Datenbank werden auf eine Menge von übergebenen Items begrenzt.

also nur für vormals reduzierte Datenquellen, beispielsweise wenn ein Recommender Items empfiehlt, die bei einem anderen vorher sprichwörtlich „durchs Raster gefallen“ sind. Dies wird im Folgenden genutzt, um komplexe RecFlows zu realisieren.

3.3 Integration von Kontext-Informationen

Die Sensoren können frei implementiert werden, wobei einige prototypisch bereits implementiert wurden⁵. Auf eine detaillierte Vorstellung dieser Sensoren muss an dieser Stelle aus Platzgründen verzichtet werden, allerdings sollte deutlich geworden sein, dass das Ergebnis eines solchen Sensors ebenfalls eine Liste unterschiedlicher Items (z.B. Städte) mit einem dazu gehörigen Wert zwischen 0 und 1 sind; beispielsweise um eine Abstufung in der Gewichtung zu erreichen: Duisburg würde z.B. mit 1 bewertet werden (kommt uneingeschränkt in Frage), Köln beispielsweise nur mit 0.5, da es etwas weiter entfernt vom aktuellen Standort ist.

Diese Informationen können nun von einem Recommender berücksichtigt werden. Hier bieten sich regelbasierte Algorithmen an, die in einem ersten Schritt eine Filterung der Hintergrund-Informationen vornimmt, so dass beispielsweise für einen Konzertveranstalter nur Tickets für nahegelegene Events angeboten werden.

Auf diese Weise können nahezu beliebige Kontext-Informationen in den Empfehlungsprozess eingebunden werden. RecFlows gibt auch hier in erster Linie ein

stabiles Interface vor, innerhalb dessen dem Nutzer freie Hand in Bezug auf die zu implementierenden Sensoren gegeben wird.

4. Prototypische Umsetzung

Wir haben den fiktiven Web-Shop *DISCOVER* implementiert, und dort unterschiedliche RecFlows integriert. Bild 7 zeigt einen Screenshot dieses Shops, auf dem verschiedene Bereiche hervorgehoben sind, die allesamt durch unterschiedliche RecFlows versorgt werden.

4.1 Aufbau des Portals

Um in dem Testportal flexibel verschiedene Recommender Kombinationen mit unterschiedlichen Visualisierungen testen zu können, wurde großes Augenmerk auf eine modulare Implementierung gelegt. Um dies zu verdeutlichen soll im Folgenden exemplarisch erläutert werden, wie die Ausgabe auf Bild 6 zustande kommt.

Die Generierung einer Seite im Portal lässt sich in zwei Phasen unterteilen: In der ersten Phase werden die Empfehlungen generiert und in der zweiten Phase

für die Ausgabe aufbereitet und ausgegeben. Klickt der Benutzer zum Beispiel den Menüpunkt „Browse Products“ an, wird die Anfrage entgegengenommen und aufgelöst. Es wird zunächst überprüft, welche Zielseite der Benutzer angefordert hat. Für jede Seite des Portals wird vorab konfiguriert, welche Elemente dargestellt werden sollen, sowie welche RecFlows für die Generierung von Empfehlungen ausgeführt werden müssen. Nachdem für die angeforderte Seite nun die Berechnungen der Recommender angestoßen wurden, wird die Ausgabe dieser vorbereitet.

Um die Empfehlungen auszugeben werden neben der Liste der empfohlenen Elemente weitere Informationen wie zum Beispiel Anzahl, Bildgröße und Detailgrad der Ausgabe benötigt. Um die Darstellungen der Empfehlungen, wie zum Beispiel Tag Wolken oder die Listenansicht, einfacher wiederverwenden zu können, wurde die Ausgabe mit Hilfe der Templating Engine Apache Tiles modularisiert. Mit Hilfe von Tiles können Ansichten einer Web Anwendung aus vordefinierten Bausteinen zusammen gesetzt werden. Anstelle also jede Ansicht von Grund auf zu definieren, ist diese eine Komposition von wiederverwendbaren Elementen. Im Falle der Recommender bestehen die



Bild 7: Dieser DISCOVER Screenshot zeigt unterschiedliche Bereiche, die jeweils durch eigenständige RecFlows angetrieben werden. (1) liefert Kategorieübergreifende Empfehlungen, (2) zeigt anstehende lokale Events, welche zum Surfverhalten des Nutzers passen, (3) eine über Facetten eingrenzbarer Menge an empfehlenswerten Produkten und (4) besonders passende Highlights aus dieser Auswahl.

⁵ Für Sensoren existiert ein eigenes Sub-Projekt Hybreed Sensors (<http://www.hybreed.org/sensors>).

Bausteine zum einen aus dem benötigten HTML Code und zum anderen aus einer Anbindung, welche aus dem Verarbeiteten RecFlow die generierten Empfehlungen entnimmt und für die View, durch hinzufügen von zusätzlichen Daten wie Bildern oder Beschreibungen, aufbereitet.

Für das Beispiel bedeutet dies, dass nach dem Berechnen der Empfehlungen, durch die Templating Engine aufgelöst wird, welche Bausteine für die Darstellung verwendet werden. Diese entnehmen die Empfehlungen um sie für die Darstellung so aufzubereiten, wie es Abbildung 6 schließlich darstellt.

Gesamt betrachtet lässt sich bei dieser Bearbeitungsweise eine klare Trennung nach dem Model-View-Controller Softwaremuster erkennen. Bei diesem Muster wird unterschieden, zwischen einem Modell, das die Geschäftsdaten verwaltet, einem Controller, der durch die Verarbeitung von Aktionen Änderungen im Modell auslöst sowie die View, welche schließlich die Inhalte des Modells darstellt. Bezogen auf das Portal, kann die Verarbeitung der Anfragen dem Controller, die Ergebnisse der RecFlows dem Mo-

del und die Aufbereitung sowie Komposition der Ausgabe der View zugeordnet werden. Das so entstandene Portal lässt sich leicht an die Anforderungen anpassen. Die Darstellung von Empfehlung und die Generierung durch RecFlows ist flexibel kombinierbar, was die Evaluation von verschiedenen Einsatzszenarien und Kombinationen ermöglicht. Im Folgenden Abschnitt wird nun näher auf die in diesem Szenario verwendeten Kombinationen der verwendeten Recommender eingegangen.

4.2 Beispielhafte RecFlows

Um die praktische Anwendung der Kombinationsmöglichkeiten von Recommendern zu verdeutlichen, wird anhand zweier Beispiele erläutert, wie eine Kombination aussehen kann und welche Vorteile diese mit sich bringt. Die Beispiele sind der e-commerce Welt entnommen und empfehlen möglichst passende Produkte. Die Beispiele wurden jeweils im DISCOVER Portal realisiert.

Ein RecFlow für lokale Events

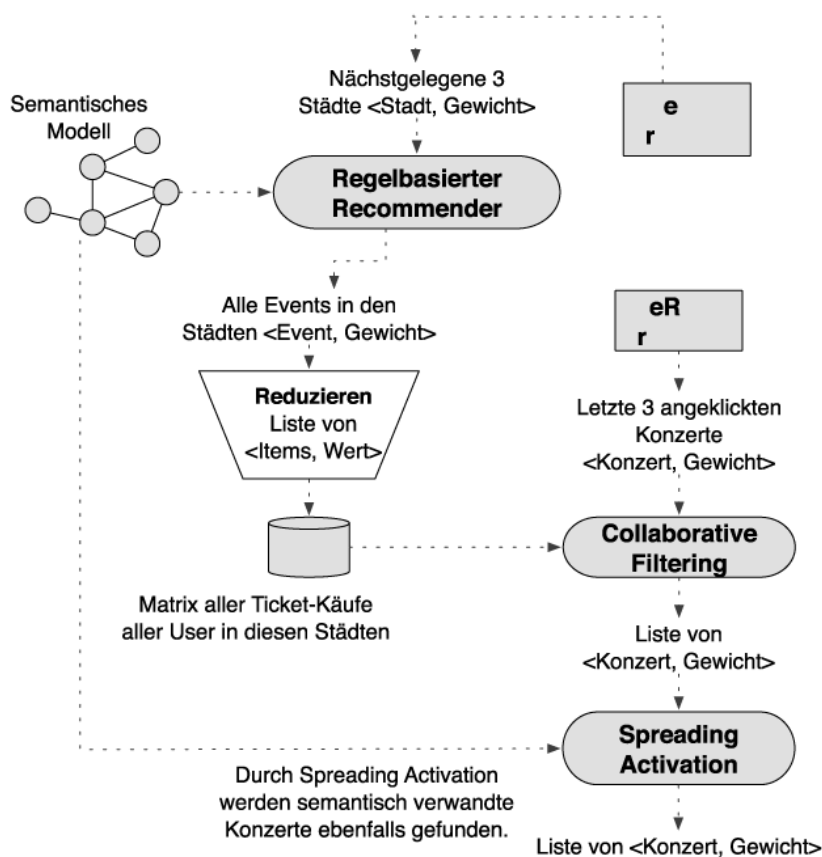


Bild 8: Exemplarischer RecFlow, welcher Konzert-Tickets in der Nähe des Nutzers passend zu seinem Musikgeschmack filtert.

Am Beispiel der lokalen Events wird nun erläutert, wie RecFlows in DISCOVER realisiert wurden. Bild 8 zeigt einen Ablauf, der passend zum Musik-Geschmack des Nutzers Konzert-Tickets in dessen Nähe empfiehlt. Dieser Recflow nutzt die Kontext-Information des aktuellen Standorts aus, um mit einem Regelbasierten Recommender gezielt Veranstaltungen in umliegenden Städten auszuwählen. Diese Veranstaltungen bilden ein reduziertes Datenmodell. Dieses Modell wird zusammen mit allen Ticketverläufen als Datenquelle für einen Collaborative Filtering Recommender verwendet. Auf Grundlage der zuletzt angeklickten Konzerte werden hier Empfehlungen generiert, die wiederum als Eingabe in einen Spreading Activation basierten Recommender gelten.

In diesem RecFlow wurden Kontext-Informationen und Recommender-Eigenentwicklungen ebenso genutzt wie etablierte vorhandene Recommender-Bibliotheken (Collaborative Filtering aus dem Apache Mahout Projekt).

Analog wurde eine Reihe weiterer RecFlows in DISCOVER implementiert, beispielsweise für Produkte passend zur Jahreszeit oder Veranstaltungstipps im aktuellen Ort passend zum Wetter.

4.3 Ein RecFlow für Artikel passend zu Ereignissen

Neben dem zuvor beschriebenen Recflow, der abhängig von der aktuellen Position Veranstaltungen empfiehlt, wird nun vorgestellt, wie Kalenderereignisse für die Empfehlungen von passenden Produkten ausgenutzt werden können. Bestimmte Ereignisse, wie zum Beispiel Weihnachten oder Ostern, lassen oft Nachfragen nach gewissen Produktgruppen entstehen, die außerhalb der Saison auf kaum Nachfrage treffen. Um diese Produktgruppen automatisiert empfehlen zu können, wird eine Kombination von Regelbasierter Empfehlung mit Collaborative-Filtering vorgestellt.

Zunächst werden im ersten Schritt mit Hilfe einer Regel Produkte oder Produktklassen ausgewählt, um als Grundlage für einen Item basierten Recommender zu gelten. Dieser wählt aus der reduzierten Produktmenge die Elemente aus, die nach den Bewertungen am besten zu dem aktuellen Benutzer des Portals passen.

Diese Kombination bietet mehrere

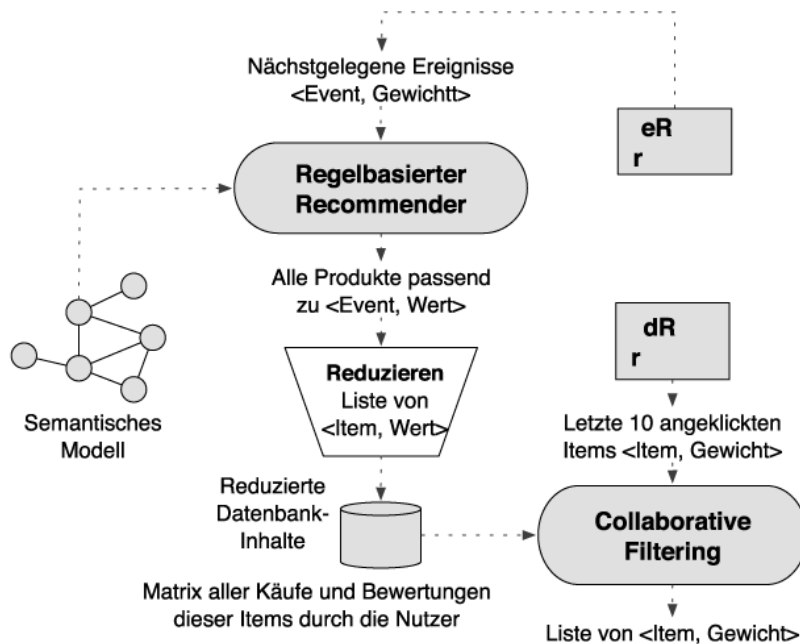


Bild 9: RecFlow, um Items zu identifizieren, welche dem Benutzer gefallen könnten und die im Zusammenhang mit aktuellen Events stehen.

Vorteile. Zum einen können gezielt Produktklassen passend für das Ereignis ausgewählt werden, ohne die Daten der Produkte erweitern zu müssen. Besonders auf kurzfristige Ereignisse kann so schnell reagiert werden. Zum anderen können generische Recommender verwendet werden, die nicht speziell für einen Zweck implementiert werden.

Lediglich die Regel des Regelbasierten Verfahrens muss ausgetauscht werden. Dieses Beispiel zeigt anschaulich, wie RecFlows Recommender gezielt für bestimmte Szenarien kombiniert werden können um darauf abgestimmte Empfehlungen zu erhalten.

5. Diskussion und Ausblick

In diesem Artikel haben wir mit Hybreed RecFlows ein modulares Framework vorgestellt, mit dessen Hilfe mit verhältnismäßig geringem Aufwand hybride Recommender aus vorgefertigten Bausteinen zusammen gesetzt werden können. Dabei wird auch die Einbindung von Kontextinformationen unterstützt, welche mit Hilfe des Hybreed Sensors generiert werden können. Dabei können nahezu beliebige Kontextsensoren eingesetzt werden, so lange sie ein vorgegebenes generisches Interface implementieren.

RecFlows wurde und wird zur Zeit prototypisch in unterschiedlichen Szenarien eingesetzt. Beispielsweise wird zur Zeit

ein System zur intelligenten Navigationsunterstützung in Fahrzeugen umgesetzt. Mit Hilfe von Hybreed werden dabei kontextabhängig bestimmte Zusatzinformationen wie z.B. günstige Tankstellen entlang der Route (bei leerem Tank) oder gut bewertete Hotels (bei langen Dienstreisen) angeboten.

Darüber hinaus ist ein kontextadaptiver Touristenführer für das Apple iPhone fast fertig gestellt, welcher zum aktuellen Ort Wikipedia-Informationen sowie passende Bilder, Videos und ggf. Events vorschlägt. Parallel wird an einer Anbindung von Hybreed an intelligente Hauselektronik gearbeitet, bei der sich elektronisch konfigurierbare Möbel und Haushaltsgeräte an die jeweilige Umgebung anpassen.

Trotz ihrer grundsätzlichen Unterschiedlichkeit haben all diese Szenarien einige Gemeinsamkeiten, so dass sie sich allesamt mit Hilfe des Hybreed Frameworks umsetzen lassen.

Literatur

- G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin. Incorporating contextual information in recommender systems using a multi-dimensional approach. *ACM Transactions on Information Systems*, 23(1):103–145, 2005.
- M. Baldauf, S. Dustdar, and F. Rosenberg. A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing*, 2(4):263–277, 2007.
- R. Burke. Hybrid recommender systems: Survey

and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.

- R. Burke. *Hybrid Web Recommender Systems*, volume 4321 of *Lecture Notes in Computer Science*, pages 377–408. Springer, 2007.
- J. Burrell, P. Treadwell, and G. Gay. Designing for context: usability in a ubiquitous environment. In *Proceedings on the 2000 conference on Universal Usability (CUU '00)*, pages 80–84, New York, NY, USA, 2000. ACM.
- L. Candillier, K. Jack, F. Fessant, and F. Meyer. *Collaborative and Social Information Retrieval and Access: Techniques for Improved User Modeling*, chapter State-of-the-Art Recommender Systems. Idea Group Publishing, 2009.
- A. K. Dey and G. D. Abowd. Towards a better understanding of context and context-awareness. In *Proceedings of the CHI 2000 Workshop on the What, Who, Where, When, and How of Context-Awareness*, The Hague, Netherlands, 2000. ACM Press.
- D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.
- J. Haake, T. Hussein, B. Joop, S. Lukosch, D. Veiel, and J. Ziegler. Modeling and Exploiting Context for Adaptive Collaboration. *International Journal of Intelligent and Cooperative Information Systems*, 19(1n02), 2010.
- T. Hussein & J. Ziegler. Adapting web sites by spreading activation in ontologies. *Proceedings of the International Workshop on Recommendation and Collaboration (in Verbindung mit IUI 2008)*, ACM, New York, 2008.
- S. Kim and J. Kwon. Effective context-aware recommendation on the semantic web. *International Journal of Computer Science and Network Security*, 7(8):154–159, 2007.
- J. McCarthy. Notes on formalizing context. In *Proceedings of the 13th international joint conference on Artificial intelligence (IJ-CAI'93)*, pages 555–560, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.
- B. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl. Item-based collaborative filtering recommendation algorithms. In V. Y. Shen, N. Saito, M. R. Lyu, and M. E. Zurko, editors, *Proceedings of the 10th international conference on World Wide Web*, pages 285–295, Hong Kong, 2001. ACM.
- A. Shepitsen, J. Gemmell, B. Mobasher, and R. Burke. Personalized recommendation in social tagging systems using hierarchical clustering. In *Proceedings of the 2008 ACM conference on Recommender Systems (RecSys)*, pages 259–266, New York, NY, USA, 2008. ACM.

B. N. Schilit, N. Adams, and R. Want. Context-aware computing applications. In D. Long, editor, Proceedings of the Workshop on Mobile Computing Systems and Applications (WMCSA 1994), pages 85–90, Santa Cruz, CA, USA, 1994. IEEE Computer Society.



1



2

1 Tim Hussein ist wissenschaftlicher Mitarbeiter am Lehrstuhl für Interaktive Systeme/Interaktionsdesign an der Universität Duisburg-Essen. Er war und ist in verschiedenen geförderten Forschungsprojekten verantwortlich für Design, Implementierung und Evaluation adaptiver Systeme.
E-Mail: tim.hussein@uni-due.de

2 Werner Gaulke ist wissenschaftlicher Mitarbeiter am Lehrstuhl für Interaktive Systeme/Interaktionsdesign an der Universität Duisburg-Essen. Er beschäftigt sich mit adaptiven Systemen und der Extraktion von Wissen aus unstrukturierten Quellen für die Verwendung in semantischen Modellen.
E-Mail: werner.gaulke@uni-due.de



Java Schritt für Schritt



Rolf Dornberger, Rainer Telesko

Java-Training zur Objektorientierten Programmierung

Leitfaden für Lehre, Unterricht und Selbststudium

2010 | 350 S. | Broschur | € 39,80 | ISBN 978-3-486-58739-5

Dieses verständlich geschriebene Buch vermittelt fundiertes Wissen über Java und Objektorientierte Programmierung bis hin zur Vertiefung komplexerer Anwendungen. Jedes Kapitel schließt mit Lernzielen und Aufgaben, die zur Wiederholung bzw. Vertiefung des Stoffinhaltes dienen.

Die Autoren legen Wert darauf, Programmieren nicht nur als das Schreiben syntaktisch korrekter Programme zu lehren, sondern auch die Philosophie der Programmierung und den Einstieg in die Objektorientiertheit zu vermitteln.

Schwerpunkte sind die Themen algorithmisches Denken, systematischer Programmentwurf und der Einsatz moderner Softwarekonzepte. Elementare Konzepte von Programmiersprachen werden unter Verwendung von Java veranschaulicht und einfache Entwicklungswerkzeuge für Java vorgestellt. Thematisiert werden auch Grundkonzepte der Objektorientierung und der Einsatz von Java für komplexere Anwendungen.

Das Buch richtet sich an Programmierneinsteiger und ist geeignet für die Lehre an Hochschulen (in der Wirtschaftsinformatik, Informatik, dem Ingenieurwesen o. Ä.), aber auch für den Informatikunterricht in der Oberstufe.

Bestellen Sie in Ihrer Fachbuchhandlung oder direkt bei uns:

Tel: 089/45051-248, Fax: 089/45051-333, verkauf@oldenbourg.de, www.oldenbourg-wissenschaftsverlag.de