Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Seminarausarbeitung

Andreas Winschu

Content Recommendation in eLearning-enabled OSN

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Andreas Winschu

Content Recommendation in eLearning-enabled OSN

Seminarausarbeitung eingereicht im Rahmen des Master Seminar

im Studiengang Master of Science Angewandte Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Kai von Luck, Prof. Dr. Gunter Klemke
Betreuer: Prof. Dr. Schmidt

Eingereicht am: 29. März 2013

Inhaltsverzeichnis

1 Einführung in das Themengebiet

1.1 Motivation

In der letzten Dekade entwickelten sich Online Social Networks (OSN) zu einem allgegenwärtigen Medium. Sie förderten die soziale Interaktion zwischen nahen bis hin zu entfernten Bekanntschaften und wuchsen zu aus dem Alltag nicht auszuschließenden Unterhaltungsplattformen. Ihre kollaborativen Funktionen können aber fernab von Unterhaltungszwecken verwendet werden und so hielten die OSNs ebenfalls Einzug in die tägliche Arbeitsroutine von Studenten und Wissenschaftlern. Eines der Intentionen dieser Zielgruppe ist es sich über neue Erkenntnisse auf den jeweiligen Lernfeldern auszutauschen und sich gegenseitig bei dem Lernprozess zu unterstützen.

Auf der anderen Seite existieren für die Bildungszwecke speziell sog. eLearning Content Management Systems (LCSMS), welche deutlich strukturierte Inhalte biten. Diese Systeme erlauben physikalisch verteilten Nutzern auf strukturierten Content zuzugreifen und auf einem festgelegtem Lernthema durch Intragruppenkommunikation zu interagieren. Im Allgemeinen sind diese Systeme jedoch an einen Instruktor, welcher Kurse erzeugt, Arbeitsresultate korrigiert und den Lernprozess unterstützt, gebunden. Üblicherweise ist weiterhin die Nutzung eine LCMSs an das zugehörige Angebot der entsprechenden Institution beschränkt.

Die Arbeit der Internet Technologies Research Group konzentriert sich darauf, die beiden obigen Paradigmen zu verbinden, sodass der Lernprozess und die Entstehung von Lerngruppen zu einem festem Bestandteil des sozialen Internet Ekosystems wird. Bei dieser Integration von OSN mit LSMS soll vor Allem die Notwendigkeit eines Moderators entfallen. Ein solcher Schritt erlaubt dem Benutzer ein selbst motiviertes Lernen auf Gebieten von persönlichem Interesse sowie eine geeignete Teamauswahl aus einer immensen Menge an möglichen Kandidaten. Das Ganze führt zu folgenden Herausforderungen:

1. Wie unterstützt man einen effektiven Teambildungsprozess?
2. Wie liefert man relevante Inhalte für die Lerngruppe?
3. Wie erleichtert man einen konsistenten Lernprozess inklusive Feedback und Korrektur?

Die erste Fragestellung wurde in Brauer und Schmidt (2012) adressiert. Um das Problem der Gruppenformation leichter anzugehen unterteilt sich der jetzige Ansatz in zwei Schritte. Zunächst wird der Graph des sozialen Netzwerks traversiert, wobei versucht wird eine minimale Anzahl an passenden Kandidaten, welche zu dem grob spezifizierten Thema des Gruppenformationsprozess Initiators passen, zu finden. Basierend auf diesen Kandidaten, wird in einem zweiten Schritt die Kanditatenkonstellation für eine erfolgreichere Lerndynamik in der Gruppe optimiert.

Diese Ausarbeitung wird sich mit dem zweiten Problem, der passenden Content Empfehlung für die Gruppen beschäftigen. Content Recommendation ist ein weit erforschtes Feld der Informatik und fällt unter das Gebiet des Information Retrieval. Hierbei werden Heuristiken, Erkenntnisse aus der Wahrscheinlichkeitstheorie sowie Verfahren zu maschinellem Lernen herangezogen.

Eins der ersten und am Meisten bekannten Prinzipien in der Content Recommendation in einem Produktivsystem ist Amazon's „User who bought this, also bought...“ Verfahren. Eine breite Welle an Veröffentlichungen und Erkenntnissen zu diesem Thema löste der Netflix Preis in 2006, welcher in einem Wettbewerb das beste Recommendation System prämierte.

2 State of the Art

2.1 Recommendation Problem

In den meisten Fällen ist Content Empfehlung mit irgendeiner Art Bewertung von Content Elementen verbunden. Diese Bewertungsmöglichkeit wird meist durch eine Funktion User Interface der Applikation bereitgestellt (e.g. Punktevergabe für Filme, Bücher), kann aber auch implizit erfolgen (Dauer der Itembetrachtung, Anzahl der Kommentare etc.). Einfach ausgedrückt lässt sich die Problematik der Content Empfehlung lediglich auf die reine Vorhersage von unbekannten Bewertungen reduzieren, also solchen die ein Benutzer auf die von ihm noch nicht bewerteten Inhalte in Zukunft vermutlich abgeben würde.

(LOUPPE, 2010, Kap. 1) definiert das Ganze formell. Wir definieren U als Menge der User und I als Menge der möglichen Content Items. Weiterhin soll r die Nutzenfunktion sein, welche den Nutzen eines Items i zu einem User u angibt, so dass folgende Abbildung gilt: $r : U \times I \rightarrow V$. Dabei weist die V Menge eine totale Ordnung auf. Somit besteht das Empfehlungsproblem für jeden User $u \in U$ in der Findung von neuen Items $i*$, welche die Nutzungsfunktion r maximiert:

$$i* = argmax(u, i)$$

Angenommen, es sollen Filme bewertet werden und die Bewertungsfunktion weist jedem Film eine Bewertung von 1-5 zu, so ergibt sich eine Bewertungsmatrix aus Abb. 1.

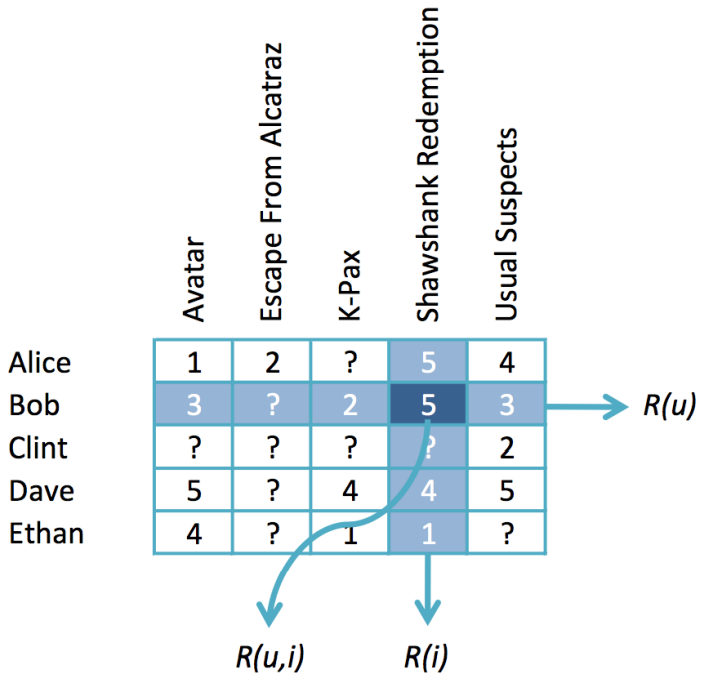

Abbildung 1: Rating Matrix LOUPPE (2010)

LOUPPE (2010) definiert weiterhin die Menge aller User, welche eine Bewertung an einem Item i abgegeben haben als $U(i)$und die Menge aller Items i, welche durch einen bestimmten User u bewertet wurden als $I(u)$. Somit wird die Menge aller Ratings an einem Item i, als $R(i)$, die Menge aller Ratings, abgegeben durch einen User, als $R(u)$ und die Gesamtmenge aller Ratings als $R(u, i)$ bezeichnet.

Mit dieser Notation definiert der Autor eine allgemeine Blaupause für einen Bewertungsalgorithmus, welcher das Empfehlungsproblem im Grunde auf die Vervollständigung der Bewertungsmatrix herunterbricht.

Algorithm 1 Insert(k)

Eingabe: ein User u
Ausgabe: Ein Item $i*$, welches empfohlen werden soll
Metode:

1. Für Alle von u nicht bewerteten Items i
   berechne eine Bewertungsvorhersage $R(u, i)$ durch einen bestimmten Algorithmus
2. Empfehle das Item mit der höchsten Bewertungsvorhersage

In der Realität ergeben sich die Probleme dadurch, dass r nicht für den gesamten Raum $U \times I$ definiert ist. Im Gegenteil ist die Anzahl an abgegebenen Bewertungen enorm klein im Vergleich zu den Millionen von möglichen Benutzern und Items. Die Herausforderung besteht nun darin r auf den gesamten Raum zu extrapolieren.

Algorithmen Typen

Die verschieden Bewertungsalgorithmen messen sich also gegenseitig in der Kunst die unbekannten Bewertungen auf der Bewertungsmatrix vorherzusagen. Dabei werden, wie bereits erwähnt, Verfahren aus einer breiten Menge an Disziplinen herangezogen. Diese Verfahren werden in ihrem Grundgedanken in zwei Oberkategorien unterteilt: *content-based* Ansätze und *collaborative-filtering* Verfahren.

content-based Recommendation konzentriert sich darauf Empfehlung auf Basis von den Elementen zu machen, für welche der Benutzer sich in der Vergangenheit interessiert hat. Typischerweise existiert dabei ebenfalls ein Profil zu den Interessen des Benutzers, dessen Attribute sich durch die Interaktion ergeben, aber unter Anderem durch den Benutzer selbst manuell vervollständigt werden können. Solche Systeme erfordern weiterhin eine Ähnlichkeitsfunktion, welche zwei beliebige Content Elemente in Korrelation stellen kann. Im Fall von Dokumenten kann etwa die sog. term frequency herangezogen werden, bei Filmen oder Büchern, kann versucht werden eine gemeinsame Übereinstimmung an Charakteristiken zu finden. Schließlich werden diejenigen Elemente vorgeschlagen, welche durch die Ähnlichkeitsfunktion in einer stärkeren Beziehung zu der Profilhistorie stehen.

In (LOUPPE, 2010, Kap. 2.2) wird die *content-based* Methode folgender Maßen formalisiert: Die Vorhersage der Bewertung $Ru, i*$ für das zu empfehlende Item $i*$ wird durch alle Bewertungen $R(u, ik)$, getätigt von dem User u an allen mit $i*$ ähnlichen Items $ik \in I(u)$, getroffen.

Bekannte Produktivsysteme, welche Empfehlungen dieser Art nutzen, sind The Internet Movie Database oder Rotten Tomatoes.

Der *collaborative filtering* Methode hingegen liegt im Kern ein ganz anderes eher sozialeres Prinzip nahe. Diese Verfahren treffen die Annahme, dass Menschen, von denen bereits bekannt ist, dass sie bestimmte ähnliche Interessen haben sehr wahrscheinlich auch in übrigen Interessen, von denen sie gegenseitig noch nichts wissen, übereinstimmen werden. Grob ausgedrückt geht man davon aus, dass „gleiche Menschen, gleiches mögen“. Im Gegensatz zu *content-based* Ansätzen werden hier also absolut keine Inhaltsinformationen über die Items in Betracht gezogen. Das Einzige, was über ein Inhaltselement bekannt ist, ist lediglich eine Referenz darauf. Empfehlung wird demnach nur auf Basis einer Ähnlichkeitsfunktion zwischen den Usern getroffen und so etwas wie Benutzerinteressenprofile existieren nicht in der eigentlichen Form.

LOUPPE (2010) formuliert das Ganze in etwa wie folgt: Die Vorhersage der Bewertung $Ru, i*$ für das zu empfehlende Item $i*$ wird durch alle Bewertungen $R(u, ik)$, getätigt von allen mit dem User u ähnlichen Usern $uk \in U$, welche das Item $i*$ bewertet haben, getroffen.

Bekannte Vertreter dieser Art von Empfehlungssysthemen im Produktiveinsatz sind Amazon oder Last.fm

Selbstverständlich sind Empfehlungssysteme nicht auf eins der beiden Paradigmen beschränkt. Wie auf der einen Seite, die Empfehlungen durch *content-based* Verfahren langweilig wirken können, weil diese immer das gleiche Thema behandeln, so können *collaborative filtering* Ansätze sich in ihrer Annahme der „gleichen Interessen“ gänzlich irren. Daher werden in der Praxis durchaus oft beide Verfahren in Kombination eingesetzt.

2.2 Collaborative Filtering

Einleitung

Wie zuvor gezeigt, halten sich Collaborative Filtering Methoden nicht an den tatsächlichen Inhalt der Empfehlungsitems, sondern analysieren lediglich die User zu User Beziehungen. Bei dieser Analyse besteht wiederum eine Unterteilung in zwei Kriterien, und zwar in der Hinsicht, ob das Verfahren versucht ein abstraktes Model aus dem vorliegenden Bewertungsmuster zu lernen. Ist das der Fall, so wird ein collaborative filtering Algorithmus zu den sog. „model-based“ Verfahren zugeordnet, anderenfalls zu der Kategorie der „memory-based“ Verfahren.

Memory Based Verfahren

„Memory-based“ Verfahren nutzen die Gesamtmenge aller Ratings R um Empfehlungen zu treffen. In dem ersten Schritt wenden diese Verfahren verschiedene statistische Techniken und heuristische Analysen an, um die Menge aller ähnlicher User N(u) zu dem vorliegenden User u zu identifizieren.

Diese ähnlichen User sind auch unter dem Begriff „Neighbors'" bekannt und daher werden „memory-based" Verfahren auch oft als „neighborhood-based" bezeichnet. Sobald diese Nachbarn gefunden wurden, geschieht die Empfehlungsberechnung für die neuen Items gegenüber dem suchenden User mittels Aggregierung der Nachbarbewertungen an diesen Items, so dass (siehe (LOUPPE, 2010, Kap. 2.2)) für die Berechnung der Empfehlung folgende Formell gilt:

$$R(u,i) = h(R(uk,i)|uk \in N(u))$$

wobei h die Aggregationsfunktion ist. Bei dieser Aggregation entsteht in dem simplen Fall ein Bewertungsdurchschnitt in den die stärke der Ähnlichkeit zu den Nachbarn, sowie die Höhe der Bewertungen gewichtet eingeflossen sind.

Ein bekannter Algorithmus, welcher nach dieser Mehtode vorgeht ist „k Nearests Neighbors (kNN)". Seine Vorgehensweise wird in der Abb. 2 deutlich. Hier sieht man das Alice, Bob, Clint und Ed sich in ihren Empfehlungen gegenüber drei bestimmten Filmen einig sind, also Nachbarn sind. Um nun Alice eine weitere Empfehlung zu machen werden die Empfehlungen der Nachbarn auf die weiteren Filme, welche Alice noch nicht bekannt sind, aggregiert. In diesem Fall würde Alice also, die höchste Empfehlung für „American Gangster" erhalten.

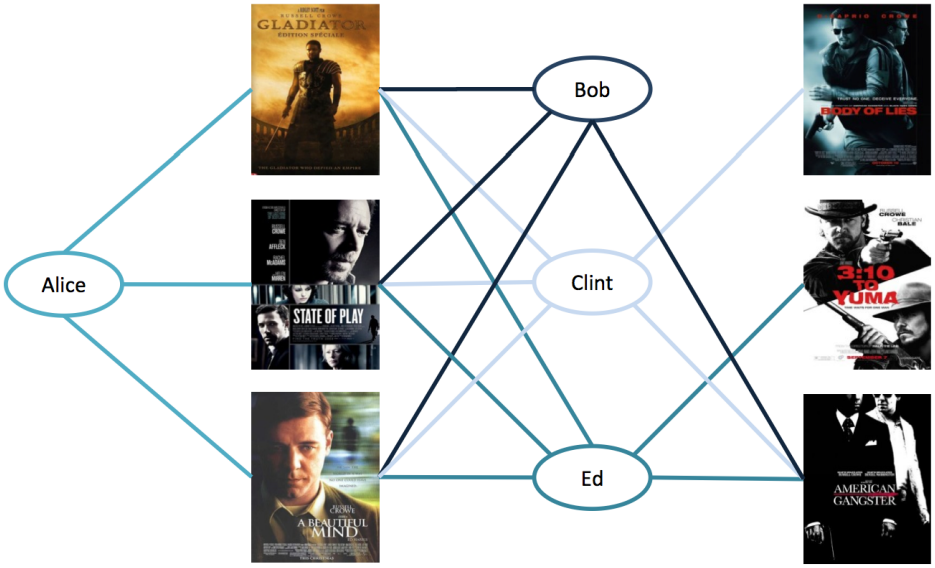

Abbildung 2: k Nearests Neighbors LOUPPE (2010)

Model Based Verfahren

Model-based Collaborative Filtering Verfahren gehen hingegen nicht direkt über die im Speicher vorhandenen Bewertungen und Beziehungen zwischen den Benutzern vor, um eine neue Empfehlung für ein dem User unbekanntes Item zu tätigen. Sie versuchen über die Zeit ein allgemeines Muster, also ein Modell, über das Bewertungsverhalten der User zu lernen, welches

dann angewendet wird um Vorhersagen über die potentielle Bewertung eines neuen Items zu treffen.

Die typischen *Collaborative Filtering* Techniken dieser Klasse sind *Latent Factor Aproaches* aus der Disziplin des maschinellen Lernens. Der Hauptansatz bei dieser Vorgehensweise liegt darin, durch Analyse der beobachteten Bewertungen in R versteckte Eigenschaften der User und Items - *Latent Factors* - durch Inferenz herauszuchristalisieren. (LOUPPE, 2010, Kap. 2.3.3) vergleicht diese `latent factors` mit Gensequenzen, welche die eigentlichen Eigenschaften kodieren.

Letztendlich stecken hinter dieser Kodierung ganz u.a. ganz offensichtliche Attribute, wie im Fall von Filmen, die Menge an Action oder Comedy oder die allgemeine Qualität des Films. Es können aber auch total komplexe Aussagen sein, wie z.B. ob ein User eher dazu neigt hohe Bewertungen zu vergeben oder wie viel Menge an Action er in einem Film erwartet. Der wichtige Unterschied zu *content-based* Methoden ist, dass diese Eigenschaften nicht explizit in dem Item/User Profil ausgefüllt werden und auch letztendlich dem Mensch unklar bleiben, sondern lediglich aus dem Bewertungsmuster und Userbeziehungen identifiziert werden und nur als ein Vektor von eindeutigen Faktoren vorliegen.

Das Prinzip des *Collaboartive filtering* unter *latent factor* Ansätzen basiert formell darauf sowohl User als auch Items in einen gemeinsamen Raum aus den *latent factors* zu projizieren, so dass User - Item Interaktionen als Skalarprodukt in diesem Raum modelliert werden können. Jeder User u wird zu einem Vektor von Faktoren $P(u) \in \mathbb{R}^f$ sowie jedes Item zu dem Vektor von Faktoren $Q(i) \in \mathbb{R}^f$ assoziiert. Die Elemente von $Q(i)$ messen den Spiegel an entsprechenden Eigenschaften des Items i während die Elemente in P das Interesse von u an diesen Eigenschaften festhalten. Das Vektorprodukt dieser beiden Vektoren repräsentiert das Gesamtinteresse eines Users $u$an einem Item i, was nichts Anderes ist, als die Bewertung $R(u, i)$ selbst. In diesem Kontext kann man das Problem des Lernens eines *latent factor* Models darauf herunterbrechen die beste Übereinstimmung zwischen dem User-Item Raum und $\mathbb{R}^f$ zu finden (vgl. (LOUPPE, 2010, S.21)).

In Abb. 3 wird dieses Prinzip veranschaulicht. Filme und User werden in dem Raum, dessen Dimensionen die Charakteristiken der Filme und das Interesse der User an diesen Charakteristiken festhalten, projiziert. In diesem einfachen Beispiel gibt es nur zwei *latent factors* welche durch die beiden Achsen gekennzeichnet sind. Eine Achse gibt wieder, ob der Film mehr zu einem Drama oder dem Comedy Genre zuzuordnen ist und die zweite ob der Film sich eher an Frauen oder Männer als Zielpublikum richtet. Ein *model-based Collaborative Filtering* Verfahren würde auch die User anhand dieser Faktoren auf die gleichen Positionen legen. Aus dieser Abbildung sehen wir das Alice mit ziemlich hoher Wahrscheinlichkeit an „Pride and Prejudice“ interessiert wäre, während „Beverly Hills Cop“ für sie deutlich weniger in Betracht käme.

Die beiden bekanntesten Ansätze, welche nach solchen *latent factors* vorgehen, sind *Matrix Factorization* und *Restricted Boltzman Machines*. *Matrix Factorization* baut wie der Name schon sagt die komplette Matrix des Raums $\mathbb{R}^f$ im Speicher auf. *Restricted Boltzman Machines* heben das Ganze Verfahren auf eine noch höhere Stufe. Sie benutzen ein neuronales Netz um nicht nur ein Muster von Interessen der User und Eigenschaften der Items herauszubekommen, sondern auch zusätzlich ein Metamodel über dieses Bewertungsmuster an sich, welches in dem neuronalen Netz festgehalten wird.

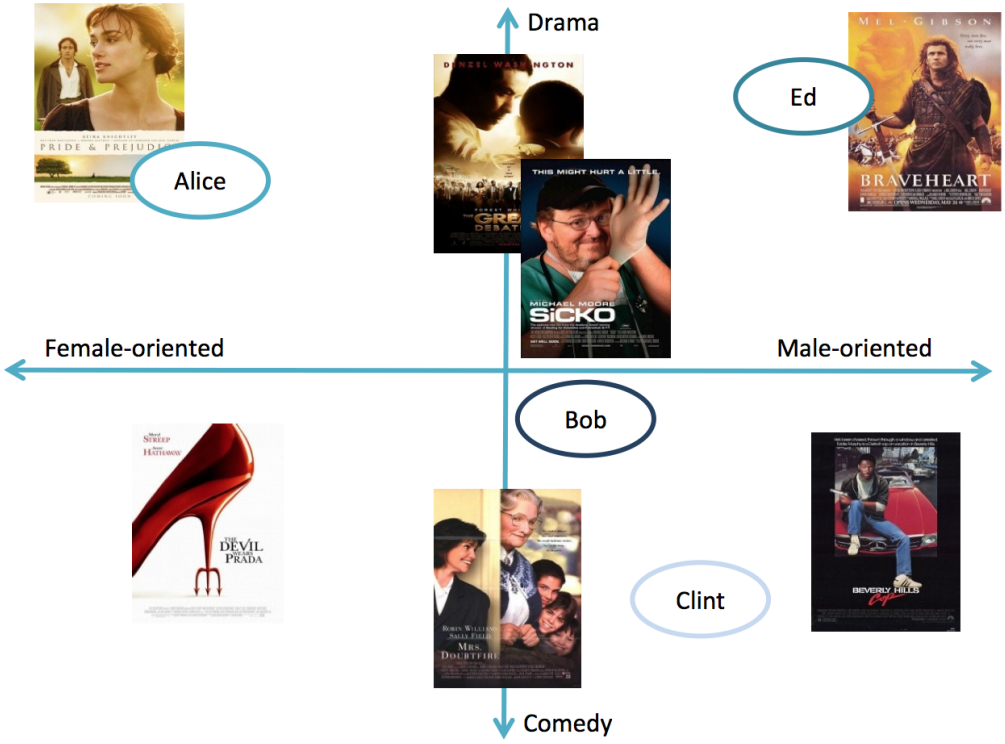

Abbildung 3: Latent Factors LOUPPE (2010)

Restricted Boltzmann Machiness

Restricted Boltzman Machines gehören zu der Klasse der sog. *Deep Belief Networks*, einer relative neuen Klasse der neuronalen Netze. Intuitiv gesagt, liegt der Zweck dieser Netze darin, ein statistischen Verhalten einer Teilwelt zu modellieren. D.h. dass eine *Boltzmann Machine* mit verschiedenen Musterverteilungen aus der realen Welt trainiert werden kann und dann in der Lage ist, ein internes Model dieser Muster selbständig zu inferieren. Auf Basis dieses Models, kann die *Boltzmann Machine* in Zukunft zu einem gewissen Teil unvollständige Muster vervollständigen.

In der Abb. 4 ist eine einzelne Restricted Boltzmann Machine dargestellt. Wie jedes neuronale Netz, besteht diese aus mehreren Knoten (Neuronen), welche miteinander verbunden sind. Jedes Neuron hat einen Zustand und eine mathematische Funktion, welche angibt, wann es an- oder ausgeht oder wie man es in der Terminologie der neuronalen Netze sagt, „feuern“ kann. Das besondere an einer Restricted Boltzman Machine ist, dass dieses Netzwerk aus zwei Teilen besteht: *hidden units* (braun) und *visible units* (blau). Die *hidden units* sind verantwortlich für die Musterkennung, also das eigentliche Training aus verschiedenen Muster Samples. Das Resultat dieses Lerntraining propagiert nach oben in die *visible units* und kann später dort abgelesen werden.

In (LOUPPE, 2010, Kap. 4.2.3) wird dieser Prozess bildlich illustriert (siehe Abb. 5). Die innere Welt einer *Restricted Boltzmann Machine* wird anhand des Szenarios der Bildvervollständigung

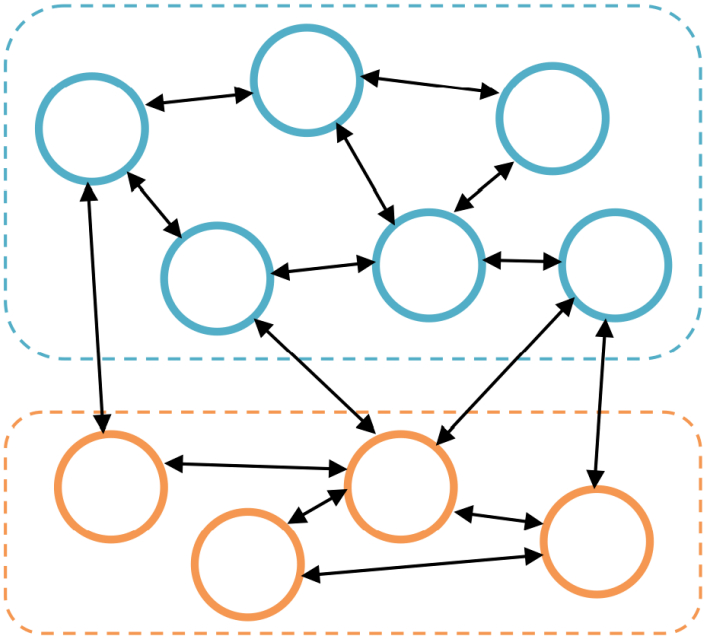

Abbildung 4: *Restricted Boltzmann Machine LOUPPE (2010)*

dargestellt. In der ersten Reihe sieht man die Training Muster. Diese werden über mehrere Iteration den Restricted Boltzmann Machines beigebracht. In der zweite Reihe sehen wir die gleichen Muster, jedoch enthalten diese kleine Abweichungen und sind teilweise nicht vollständig. Wenn man nun nach einer Trainingsphase mit den vollständigen Mustern der Maschine ein solches unvollständiges Muster präsentiert, kann man aus den *visible units* das richtige vollständige Muster ablesen. In der unteren Reihe sieht man das von der *Restricted Boltzmann Machine* vervollständigte Resultat.

Wenn man sich nun vorstellt, dass die Verteilung der verschiedenen geometrischen Muster aus dem obigen Beispiel auf die User Ratings eines Bewertungssystems korrespondiert, so erhält man den Ansatz für ein *Collaborative Filtering* Verfahren, welches *Restricted Boltzmann Machines* zum Nutzen macht. Ein Pixel entspricht dabei einem Item und seine Intensität der Bewertung, welche ein Benutzer einem Item gegeben hat. Gemäß dieser Vorstellung, ist die Empfehlung eines Items, lediglich die Vervollständigung eines kaputten Rating Musters, welche nun auf den gesamten Datensatz angewendet werden werden muss.

Allgemeine Schwierigkeiten

Insgesamt stehen sich Empfehlungssysteme folgenden drei Schwierigkeiten gegenüber:

- sparsity - Es wurden zu wenige Bewertungen, gemessen an der Gesamtmenge der Elemente und User abgegeben, so dass keine Nachbarn gefunden werden können bzw. das gelernte Model nicht die Realität wiederspiegelt. Hier versuchen die Verfahren oft einen

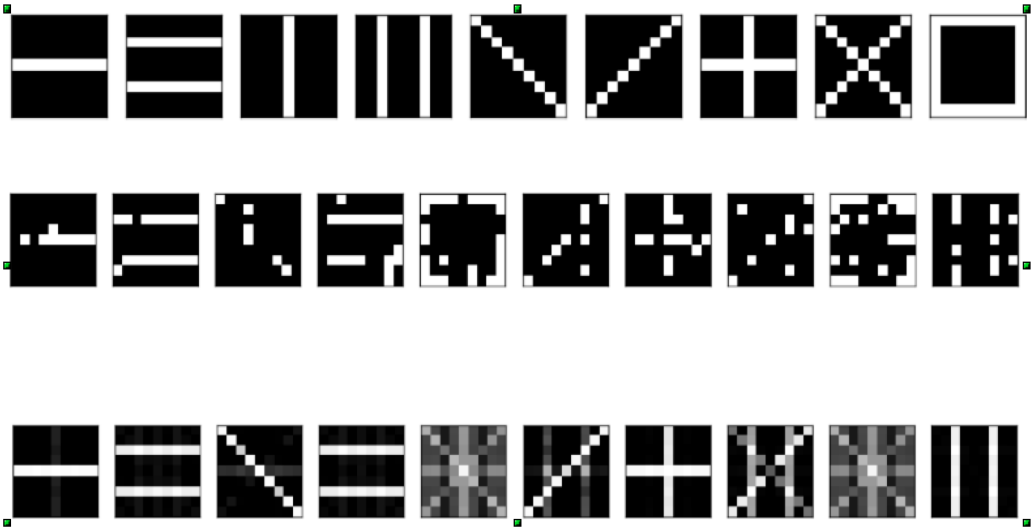

Abbildung 5: Die Innere Sichtweise einer *Restricted Boltzmann Machine* LOUPPE *(2010)*

impliziten User Feedback einfließen zu lassen, was jedoch eine weitere Herausforderung darstellt.

- diversity - Besonders gute Inhalte werden immer wieder empfohlen. „'Rich get richer" Syndrom
- scalability - Mit wachsender Benutzerzahl (parallele Empfehlungsanforderungen) und Datenmenge, steigt der Ressourcenverbrauch des Algorithmus so enorm, dass keine Empfehlung in adäquater Zeit vorhergesagt werden kann

2.3 Diaspora Social Network

Unsere Lösung zu einer Integration von eLearning Systemen und Social Netwoks wird auf dem Open Source Social Network namens *Diaspora* [1] basieren. Die Alternative läge darin eins der populäreren Netzwerke, wie *Facebook* oder *Google +* zu betrachten und mittels deren public API's auf die Netzwerkinfrastrukturen zuzugreifen. Obwohl die API's durchaus umfangreich genug sind und das obige Vorhaben sicherlich ermöglichen, hat die Entscheidung zu einem Open Source Social Network einen anderen Hintergrund.

Zunächst ein Mal sollen mögliche Datenschutzbedenken vermieden werden, was bei den populären OSN nicht gewährleistet ist, da deren Kerngeschäft in der Auswertung personenbezogener Informationen liegt und diese sich nicht damit rühmen mit solchen vertraulich umzugehen.Das zweite Kontra Argument liegt in den Sicherheitsrestriktionen der öffentlichen Schnittstellen dieser OSN. Ein Programm welches die public API dieser Netzwerke nutzt hat niemals Zugriff auf den gesamten Social Graph. Z.B. kann dabei nur auf Daten von Benutzern zugegriffen werden,

[1] http://diasporaproject.org/

welche auch dieses Recht dem Nutzer einer OSN Schnittstelle gewährt haben. Zwar, wie (Brauer und Schmidt, 2012, Kap. 3) beschreibt, kann das Vorhaben eh nur realisiert werden, indem Benutzerprofile um die notwendigen Metainformationen für Content Recommendations erweitert werden. Dennoch liegt die Intention der Arbeit im Mindstone Projekt möglichst viel von der OSN und LSMS Infrastruktur nutzen zu können.

Diaspora eignet sich für das Vorhaben ideal. Es legt von Kern auf Wert auf Datenschutz, indem es den Usern die Kotrolle über deren Inhalte überlässt. Es besitzt zwar keine öffentliche API, jedoch ist es komplett Open Source. Zusätzlich ist es mit dem Framework *Ruby on Rails* [2] realisiert. *Ruby* gilt als eine besonders erweiterungsfreundliche Sprache und erlaubt uns somit den Eingriff in die gesamte *Diaspora* Infrastruktur, während *Rails* das defacto Standardframework für Erstellung von Webapplikationen in *Ruby* ist. Die Anbindung unserer Integrationssoftware erfolgt mittels einer sog. *Rails Engine*, welche im Grunde eine vollständige *Rails* Applikation, innerhalb einer *Rails* Applikation ist und an jedem Punkt im Framework eingreifen kann. Näheres zu Daspora unter Diaspora (2013), sowie zu Rails unter Rails (2013c) und insbesondere zu Rails Engines in Rails (2013a)

2.4 Neo4J

Webapplikationen Entwickler haben in den letzten Jahren festgestellt, dass deren Anforderungen oftmals viel einfacher zu realisieren sind, wenn sie auf die Datenspeicherung in einem relationalem Schema verzichten. So entstanden verschiedene Datastores, welche sich einem bestimmten Problem in besonderer Weise widmen unter dem Stichwort NoSQL - Not Only SQL. Oft werden dabei die *ACID* Garantien zugunsten von Partitiontolerance aufgegeben und ein loseres dynamisches Schema Model gewählt, welches erlaubt die Daten ohne vorherige explizite Schemadefinition zu speichern. (Siehe Cartell (2011)). Das Motto der NoSQL Bewegung liegt darin sich von dem “one size fits all” Paradigma wegzubewegen, sondern je nach Situation eine möglichst passende kostengünstige Kombination aus Datastores zu wählen, auch wenn man dabei stellenweise in weniger kritischen Bereichen auf gewisse Garantien verzichtet.

Neo4j ist eins der Vertreter dieser NoSQL Bewegung. Dieses adressiert das Problem der stark miteinander verknüpften Daten. Relationale Datenbanken brachten uns zwar ein Werkzeug um redundante Datenhaltung durch Normalisierung zu vermeiden, indem eindeutige Datensätze mittels Primär-/Fremdschlüssel Beziehung referenziert werden. Jedoch sind die Verbundsoperationen auf diesen Daten, welche die Verknüpfungen wieder zusammenführen verhältnismäßig teuer und lösten schon sofort nach ihrer Einführung einen wahren Harwareupgrade Boom aus.

Neo4j hat laut (Michael Hunger, 2012, Kap.19) ein intuitives, reichhaltiges graphenbasiertes Model der Datenrepräsentation. Anstatt Daten in Tabellen, Zeilen und Spalten zu halten, arbeitet man mit einem Graphen aus Knoten, Beziehungen und Eigenschaften. Es besitzt ein Traversierungsframework, sowie domainbezeogene Anfragesprachen, um mit dem Graphenmodel zu arbeiten (e.g *Cypher*). Es gibt die *ACID* Garantien nicht auf, bietet aber dennoch horizontale Scalierung über mehrere physikalische Instanzen. Davon abgesehen kommt diese Datenbank

[2] http://rubyonrails.org/

mit mehreren Milliarden Knoten/Beziehungen/Eigenschaften auf einer einzelnen Maschine ohne Probleme aus.

Gerade OSN's sind zu der Erkenntnis gelangt, dass deren Daten nicht nur sehr stark miteinander verknüpft sind, sondern dass auch die performante Auswertung dieser Verknüpfungen eine Alltagsoperation deren Kerngeschäfts ist. Zu dem liegt die Idee das Recommendation Problem mit Hilfe von Grpahendatenbanken zu lösen gerade auf der Hand und wurde zu einem weiteren Trend. Aus diesen beiden Gründen untersucht das Mindstone Projekt den Einsatz des Neo4J Graph Store für die angemessene Lerninhalt Empfehlung.

Die bekannte Statistiken und Trends Sammlung *The Ruby Toolbox* [3] gibt als populäre Bibliothken für die Arbeit mit der *Neo4J* Datenbank in *Ruby* die *Ruby* Gems *neo4j* [4], *neography* [5] und *architec4r* [6] an.

Da die *Neo4J* Datenbank in *Java* geschrieben ist, ist ein driverbasierter Zugriff nur mittels *JRuby* möglich. Diesen Ansatz wählt das *neo4j* gem. Neben einem Low Level Interface bietet dieses Gem ebenfalls einen abstrakten Mapper, welcher den Zugriff ähnlich dem aus *Ruby on Rails* bekanntem *ActiveRecord* Pattern (siehe Rails (2013b)) für relationale Datenbanken möglich macht. Neography nutzt die *Neo4J* REST API kann mit jeder *Ruby* Implementierung verwendet werden. Neben den Low Level Operationen bringt diese Bibliothek zwar eine Abbildung von Nodes und Relationships auf Objekte mit, jedoch kein so reicheres *ActiveRecord* ähnliches Datenmodel, wie das *neo4j* Gem. *Arhitect4r* wählt einen dritten Weg. Es nutzt ausschließlich die *Cypher* Query Language (siehe (Technology, 2012, Kap.15))von *Neo4j*. Es bietet ein ähnlich reiches Datenmodel wie das *neo4j* Gem.

[3] https://www.ruby-toolbox.com/
[4] https://github.com/andreasronge/neo4j
[5] https://github.com/maxdemarzi/neography
[6] https://github.com/namxam/architect4r

3 Related Work

3.1 Einleitung

In Brauer und Schmidt (2012) wurde ein Weg gezeigt, wie Lerninteressenten dynamisch zu Gruppen mit gleicher Stärke finden können. Um nun anschließend die in dieser Arbeit angesprochene nächste Problematik der passenden Lerninhalt Empfehlung für eine solche OSN eLearning Gruppe anzugehen, werden hier vergleichbare Arbeiten betrachtet, wobei die Strukturierung wie folgt ist:

Zunächst werden aktuelle Arbeiten gelistet, welche sich mit dem Recommendation Problem generell beschäftigen. Weiterhin werden ausgewählte Arbeiten präsentiert, dessen Kernbeitrag zu dem Recommendation Problem besonders das OSN Umfeld adressiert bzw. adressieren könnte. Und schließlich wird auf eine Ausarbeitung eingegangen, welche Ansätze dazu liefert, welche ein OSN adaptiertes eLearning System heranziehen könnte, um besonders gut von der collective Intelligence deren Nutzer profitieren zu können, um sich so damit von den üblichen eLearnnin Systemen abzugrenzen.

3.2 Recommendation Problem

Wie sich bereits in 2 andeutete ist das grundlegende Recommendation Problem ein weit erforschtes Thema. Dieses konnte kürzlich um 2009 erneut eine ganz besondere Aufmerksamkeit genießen, ausgelöst durch den Netflix Preis (LOHR (2009)).

Das Online Filmportal Netflix war an die Grenzen ihres Recommendaton Algorytmus gelangt und hat einen Award, prämiert mit einer hohen Geldsumme,an das Team, welches eine Verbesserung um 10% gegenüber dem von Netflix implementierten eigenen Algorithmus liefern konnte verliehen, gemessen an dem sog. Root Mean Square Error. Um diesen Maß zu ermitteln stellte Netflix ein Training Datensatz aus 100.408.507 Filmbewertungen bereit zu dem die Firma ebenfalls ein Qualifying Datensatz auslieferte, welches aus 2.827.131 Ratings der gleichen User bestand, die nicht im Training Set auftauchten. Die neuen Recommendation Algorithmen sollten also möglichst gut, die realen Bewertungen der Nutzer aus dem Qualifying Set, basierend auf den Bewertungsmustern im Training Set treffen.

Das Siegerteam der Firma Belkor nutzte ein kombiniertes Verfahren aus 107 verschiedenen Recommendation Techniken und veröffentlichte deren Erkenntnisse unter anderem in Bell und Koren (2007), Bell u. a. oder Koren (2009) wobei die Vor- und Nachteile der jeweiligen Methoden genannt wurden. Es gibt zahlreiche andere Publication auf diesem Gebiet, welche schon vor dem Netflix Preis behaupteten, ein bestimmtes Problem, wie z.B. sparsity, diversity oder scaleability besonders gut zu lösen (Linden u. a. (2003), Awerbuch u. a. (2005), Drineas u. a. (2002) , Kleinberg und Sandler (2003), Kleinberg und Sandler (2004)). Es besteht jedoch kein Bedarf mehr aus wirtschaftlicher Sicht die Verfahren weiterhin besonders gut zu optimieren. In der Realität funktionieren in bestimmten Fällen auch einfache modellose Neigborhood Search Methoden um Content Empfehlungen zu treffen. Allerdings ist die Wahl des jeweiligen Verfahren für die passende Domain entscheidend.

Ein relativ simples Verfahren wird beispielsweise in Yildirim und Krishnamoorthy (2008) vorgestellt. Dieses setzt genau wie der Gruppenbildungsprozess aus Brauer und Schmidt (2012) auf Random Walks auf, was dem Ganzen eine zusätzliche Aufmerksamkeit verleiht. Außerdem setzt das Verfahren stärker auf die Item zu Item Beziehungen, erstellt also ein Metamodell wie in 2.2 beschrieben, welches User und Item Ähnlichkeiten aufdeckt. Solche Vorgehensweise eignet sich besonders gut für das OSN eLearning Umfeld, da sowohl der thematische Bezug zu den Lerninhalten gegeben ist, als auch der sozialen Aspekt aus dem OSN betrachtet wird.

Yildirim und Krishnamoorthy (2008) beschreibt den Ansatz als eine Erweiterung des *item based top-N recommendation* Algorithmus (Deshpande und Karypis (2004)). “Wir schlagen ein Model vor, bei dem Benutzer Random Walks im Item Raum gemäß der Item Ähnlichkeit ausführen. Hierbei wird eine initial normalisierte Itemähnlichkeitsmatrix als Übergangswahrscheinlichkeitsmatrix interpretiert. Somit schreitet ein User, welcher gerade ein Item betrachtet, sehr warhscheinlich zu einem sehr ähnlichen Item in dem nächsten Schritt seines Random Walks. Die aggregierte Wahrscheinlichkeitsverteilung eines User über die Items während des Radom Walks zu wandern stellt die finale Bewertungsvorhersage für die Items. Die Methode hat drei wesentliche Schritte: (i) Bilden der initialen Übergangswahrscheinlichkeitsmatrix, (ii) Abrufen der aggregierten Warhscheinlichketisverteilungen und (iii) Interpretieren dieser Wahrscheinlichkeiten um Bewertungen vorherzusagen.”

3.3 OSN und Collaborative Recommendation

Im vorherigen Kapitel wurde gezeigt, dass Collaborative Recommendation ein weit erforschtes Gebiet ist und es wurde auf Arbeiten hingewiesen welche grundlegende Ansätze zu Content Empfehlung liefert. In diesem Abschnitt werden nun zwei Projekte vorgestellt, welche den Fokus bei der Collaborative Reocemmendation auf die Domain der OSN’ legen.

So behauptet Konstas u. a. (2009) das Ergebnis eines Random Walk mit Restarts (RWR) Algorithmus signifinkant verbessert zu haben, indem es mit Daten aus einem Social Graph angereichert werden konnte. In dieser Arbeit wurden als Testdatensätze die öffentlich zugänglichen Information über Musikstücken und “Fans” des Portals last.fm herangezogen. Ferner wird als Datenmodel ein Graph gewählt, da diese Struktur analog zu den hoch verknüpften Daten des sozialen Netzwerks ist.

Konstas u. a. (2009) stellt das Verfahren wie folgt vor:

“Wir untersuchen einen Datensatz, basierend auf den Daten von dem last.fm OSN, welches einen sozialen Graphen zwischen Usern, Musiktracks und Tags bildet und effektive Freundschaftsverknüpfungen sowie kolloborative Annotationen enthält.

Wir evaluieren ein Random Walk mit Restarts (RWR) Modell auf diesem Datensatz und zeigen, dass das Hinzuziehen von Freundschaftsbeziehungen und sozialer Tags die Performanz eines Item Empfehlungssystems steigern kann.

Wir zeigen, dass die RWR Methode eine Standard Collaborative Filtering Methode, welche wir an dem gleichen Datensatz evaluieren, an dem Ergebnis übertrifft.

Wir zeigen, dass unsere RWR Methode keinerlei Training benötigt und es erfolgreich schafft das Wissen, welches in dem OSN vorhanden ist zu managen. ”

Zu einer ähnlichen Schlussfolgerung kommt ebenfalls Dalia Sulieman (2012). Die Forscher stellen ein semantisches social recommender System vor, welches wiederum auf einem Datenmodel operiert, bei dem User durch ein soziales Netzwerk verbunden sind. Die Besonderheit des Verfahrens stellt die Tatsache dar, dass User und Items zusätzlich durch eine Taxonomy beschrieben sind. Angewendet und verglichen werden zwei verschiedene Verfahren, basierend entweder auf Depth First Search (DFS)oder Breath First Search (BFS). Letztendlich kommt die Arbeit ebenfalls zu dem Schluss, dass die zusätzliche Information aus der Taxonomy, den social Annotations und Tags von Vorteil ist und so herkömmliche Lösungen in dieser Domain übertrifft. Außerdem hilft die zusätzliche Information das Netzwerk so wenig wie möglich zu scannen.

Dalia Sulieman (2012) präsentieren die Lösung wie folgt:

“ Der semantische Teil unseren vorgeschlagenen Modells setzt auf drei fundamentale Aspekte auf:

1) User Präferenzen: gruppiert in einem Benutzerprofil, welches alle möglichen Informationen, wie Aktivitäten und Interessen, über die Benutzer enthält. Das Benutzerprofil hat mehrere Ausprägungen, wie vektorbasierende und konzeptionelle Repräsentation.

2) Domain Taxonomy: Eine Taxonomy wird definiert durch eine Sammlung von Entitäten, organisiert in einer hierarchischen Struktur (’is-a’ Hierarchie), um Objekte aus einer bestimmten Domain beschreiben zu können.

3) Semantische Ähnlichkeit Messung: wird benutzt, um die Relevanz zwischen Ontologie Konzepten zu berechnen.

In unserem Model benutzen wir eine domain-spezifische Taxonomy um das Wissen über die User und Items zu repräsentieren. Wir hängen weiterhin jeweils ein semantisches Taxonomyprofil jedem User und Item an und schließlich benutzen wir die semantische Ähnlichkeit Messung um die semantische Relevanz zwischen Benutzern und Items zu berechnen.”

Die dem ganzen Verfahren zu Grunde liegen Datenstruktur ist auch hier wiederum ein Graph, dessen Traversierung, wie bereits oben erwähnt mittels BFS bzw. DFS stattfindet. Jedoch findet hier ein Optimierung durch das Hinzuziehen von heuristiken statt um das besuchen aller Kanten und Knoten zu vermeiden. Diese Heuristiken basieren auf drei definierten Konzepten: semantische Ähnlichkeit, Zentralitätsgrad und Gewichtung von Graphenkanten.

3.4 Collective Intelligence

In dem vorherigen Abschnitt wurden Arbeiten gezeigt, welche das collaborative Filtering auf der Domain der OSN erforschten. Es zeigte sich stets, dass das Hinzuziehen der semantischen Informationen zu den Items und Userprofilen, sowie das Ausnutzen der Annotationen und Tags aus dem social Graph sich positiv auf die Qualität der Item Empfehlung auswirkt. Von daher ist für eine intelligentere Verfahrensweise eine möglichst strukturierte zusätzliche semantische Information ausschlaggebend.

Gruber (2008) zeigt Ansätze, wie man zu solchen strukturierten Information gelangen kann und sich so von einer puren Wissensammlung zu kollektiven Intelligenz bewegen könnte. Es werden mehrere Beispiele aktueller kollaborativer Systeme gezeigt, wobei der Autor zu dem Schluss kommt, dass dabei eine “collective Intelligence” nicht wirklich erreicht wird. Anstelle dieser herrscht meist ein Zustand, den der Autor als “collected Intelligence” bezeichnet. D.h. es findet eine bloße Ansammlung von Informationen, aber keine wirkliche Inferenz dieser Statt.

Obwohl man aber noch nicht zu einer “collektive Intelligence” gelangen konnte, so funktionieren die Systeme dennoch erstaunlich gut. Dies wird am Beispiel von Fragen und Antworten Systemen, welche der Autor unter FAQ-Sphere klassifiziert (Abb 6), deutlich. Dabei besteht meist ein Zusammenspiel aus mehren Webapplikationen, Feedbackverfahren und der Intelligenz der Nutzer mit dem jetzigen Zustand umgehen zu können. Z.B. haben die Benutzer es so gut gelernt ihre tatsächlichen Anfragen in Suchanfragen so zu formulieren, dass sie gezielt die passenden Antworten finden können. Auf der anderen gibt es die Suchmaschinen, welche zu einer Auswahl von Keywords durch Ranking Verfahren immer die richtigen Blog- , Foren- oder Newsgroupeinträge finden. Die FAQ Sphere wurde nicht als ein System konzepiert und dennoch funktioniert sie so gut, dass z.B. ein Entwickler zu einer Fehlermeldung erstaunlicherweise meist die richtige Problemlösung finden kann.

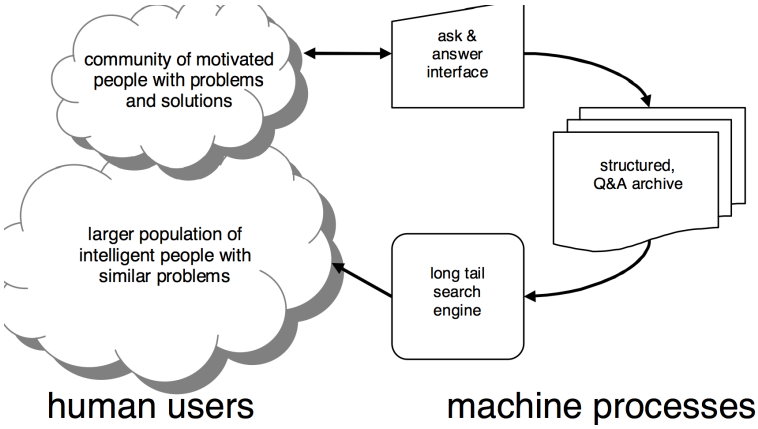

Abbildung 6: FAQ Sphere - Beispiel eines collective Knowledge Systems Gruber (2008)

Doch das Finden von richtigen Informationen, sowie das Empfehlen passenden Inhalts kann entschieden verbessert werden, indem man zu strukturierteren Daten gelangt. Der Autor nennt drei Ansätze, wie dies geschähen kann: 1) Struktur beim Ausliefern von Daten für das Web beibehalten, 2) strukturierte Daten aus unstrukturierten Eingaben extrahieren und 3) die Benutzer bei Eingaben strukturierter Daten unterstützen.

Der erste Punkt ist eindeutig und soll Entwickler anregen, die Struktur, welche in ihren Datastores zu Grunde liegt, auch bis zu dem Endpunkt, z.B. mit Hilfe von HTML5 semantischen Annotationen auszugeben, so dass andere Systeme diese Inhalte auch maschinell verarbeiten können.

Bei dem zweiten Punkt referenziert der Autor mehrere Text- und Datamining Ansätze, welche aus normalen Prosatexten recht erfolgreich strukturierten Informationen gewinnen können. So sollen

Benutzereingaben, wie z.B. Antworten auf Fragen in Newsgroups gleich strukturiert abgespeichert werden können.

Als einen in naher Zukunft vielversprechendsten Ansatz bezeichnet Gruber (2008) eine Technik, welche er als “snap to grid” tauft Dabei sollen strukturierte Daten gleich vom System gesammelt werden können. Eine simpler Weg dafür ist z.B. eine intelligentere Autovervollständigung von Benutzereingaben. Wie das funktionieren kann, wird am Beispiel einer Reiseplanungsanwendung gezeigt, welche Benutzern ermöglicht sich über die Reiseorte auszutauschen und Reisen gemeinsam zu planen. Diese Anwendung hilft z.B. dem Benutzer immer den richtigen Ort einzugeben, auch wenn es bestimmte Ortsnamen gibt, welche mehrfach auf der Welt vorkommen. Der Benutzer braucht jedoch nicht eine exakte geographische Angabe zu tätigen, sondern lediglich aus einer Autovervollständigungsliste den richtigen Ort auszuwählen. Die korrekte eindeutige strukturierte Ortsangabe landet aber letztendlich korrekt im System. Dieses ist ein simples Beispiel und so werden von Gruber (2008) weitere “snap to grid” Anwendungsfälle genannt, welche dem Benutzer helfen sollen korrekte semantische Eingaben zu tätigen ohne es zu wissen.

4 Schluss

In Brauer und Schmidt (2012) wurden die ersten Ansätze zu einem eLearning unterstützendem OSN gelegt, wobei ein Verfahren enstand User in zufällig, jedoch stets in Betracht derer Fähigkeiten, in Gruppen aufeinander treffen zu lassen. Diese Ausarbeitung legt den Grundsazt zu einem Verfahren, welches einer solchen entstanden Gruppe eine Contentempfehlung gemäß der Interessen ihrer Teilnehmer tätigen kann.

Zunächst wurden sog. Collaboartive Filtering Verfahren untersucht, wobei gezeigt wurde, dass traditionelle Empfehlungssysteme keine Profile von Benuztern oder Items benötigen, sondern lediglich nach dem Leitbild “Bentuzer, welche gleiche Items ähnlich bewerten, werden auch weitere Items ähnlich bewerten” vorgehen. Allerdings gibts ebenfalls hybride Ansätze, welche nicht nur solche Bewertungsbeziehungen, sondern auch Item und Benutzerprofile zusätzlich in Betracht ziehen. Doch auch reine collaborative Filtering Techniken ohne Benutzer und Itemprofile können davon profitieren ein Metamodell der Bewertungen zu bilden und so die eigentlichen Profileigenschaften implizit durch sog. *latent factors* aufdecken. Um ein solches Metamodell zu erstellen können sogar Verfahren aus dem Bereich des Maschine Learning Feldes herangezogen werden, was am Beispiel von Restricted Boltzman Machines gezeigt wurde.

Schließlich wurden Arbeiten gezeigt, dessen Gegenstansthema dem Problem der passenden Contenempfehlung in einem eLearning OSN nahe lag. Dabei zeigte sich, dass meistens sich ein Graph als Datenstruktur für die Empfehlungsanalysen eignet und dass in einem OSN Umfeld Empfehlungen an Qualität gewinnen können, wenn semantische Informationen, wie Freundschaftsbeziehungen, Tags und weiter Annotationen aus dem social Graph zusätzlich dem Collaboartive Filtering Verfahren zur Verfügung stehen.

Das Mindstone Projekt will ein Open Source OSN Namens Diaspora erweitern, so dass dieses Lerninhalte aus einem LCSMS beziehen kann. Weiterhin sollen Benutzerprofile durch semantische Annotation eweitert und in der graphenbasierten Datenbank Neo4j abgelegt werden, so dass später

ein Collaboartive Filtering Verfahren mit zusätzlicher Inbetrachtnahme dieser Information darauf Contentempfehlungen trifft.

Literatur

[Awerbuch u. a. 2005] AWERBUCH, Baruch ; PATT-SHAMIR, Boaz ; PELEG, David ; TUTTLE, Mark R.: Improved recommendation systems. In: *SODA*, SIAM, 2005, S. 1174–1183. – URL http://dblp.uni-trier.de/db/conf/soda/soda2005.html#AwerbuchPPT05. – ISBN 0-89871-585-7

[Bell und Koren 2007] BELL, Robert M. ; KOREN, Yehuda: Scalable Collaborative Filtering with Jointly Derived Neighborhood Interpolation Weights. In: *IEEE International Conference on Data Mining (ICDM*, 2007

[Bell u. a.] BELL, Robert M. ; KOREN, Yehuda ; VOLINSKY, Chris: *The BellKor solution to the Netflix Prize*

[Brauer und Schmidt 2012] BRAUER, Steffen ; SCHMIDT, Thomas C.: Group Formation in eLearning-enabled Online Social Networks. In: AUER, Michael E. (Hrsg.): *Proc. of the International Conference Interactive Computer aided Learning (ICL'12)*. Piscataway, NJ, USA : IEEEPress, Sep. 2012

[Cartell 2011] CARTELL, Rick: *Scaleable SQL and NoSQL Datastores*. 2011. – URL http://www.cattell.net/datastores/Datastores.pdf. – Letzter Aufruf am 27. Feb 2011

[Dalia Sulieman 2012] DALIA SULIEMAN, Hubert Kadima Dominique L.: Semantic Social Breadth-first search and Depth- first search Recommendation Algorithms / ETIS-ENSEA Universite de Cergy-Pontoise CNRS France. 2012. – Forschungsbericht

[Deshpande und Karypis 2004] DESHPANDE, M. ; KARYPIS, G.: Item-based top-n recommendation algorithms. In: *ACM Transactions on Information Systems (TOIS)* 22 (2004), Nr. 1, S. 143–177. – URL http://scholar.google.de/scholar.bib?q=info:HcsE95oEHDEJ:scholar.google.com/&output=citation&hl=de&as_sdt=0,5&ct=citation&cd=0

[Diaspora 2013] DIASPORA: *What's Diaspora?* Webseite. 2013. – URL http://diasporial.com/whats-diaspora

[Drineas u. a. 2002] DRINEAS, Petros ; KERENIDIS, Iordanis ; RAGHAVAN, Prabhakar: Competitive recommendation systems. In: REIF, John H. (Hrsg.): *STOC*, ACM, 2002, S. 82–90. – URL http://dblp.uni-trier.de/db/conf/stoc/stoc2002.html#DrineasKR02. – ISBN 1-58113-495-9

[Gruber 2008] GRUBER, Tom: Collective knowledge systems: Where the Social Web meets the Semantic Web. In: *Web Semant.* 6 (2008), Nr. 1, S. 4–13. – URL http://tomgruber.org/writing/collective-knowledge-systems.htm. – ISSN 1570-8268

[Kleinberg und Sandler 2004] KLEINBERG, J. ; SANDLER, M.: Using Mixture Models for Collaborative Filtering. In: *Proceedings of the 36th ACM Symposium on Theory of Computing*, 2004

[Kleinberg und Sandler 2003] KLEINBERG, Jon M. ; SANDLER, Mark: Convergent algorithms for collaborative filtering. In: *ACM Conference on Electronic Commerce*, ACM, 2003, S. 1–10. – URL http://dblp.uni-trier.de/db/conf/sigecom/sigecom2003.html#KleinbergS03. – ISBN 1-58113-679-X

[Konstas u. a. 2009] KONSTAS, Ioannis ; 0002, Vassilios S. ; JOSE, Joemon M.: On social networks and collaborative recommendation. In: ALLAN, James (Hrsg.) ; ASLAM, Javed A. (Hrsg.) ; SANDERSON, Mark (Hrsg.) ; ZHAI, ChengXiang (Hrsg.) ; ZOBEL, Justin (Hrsg.): *SIGIR*, ACM, 2009, S. 195–202. – URL http://dblp.uni-trier.de/db/conf/sigir/sigir2009.html#KonstasSJ09. – ISBN 978-1-60558-483-6

[Koren 2009] KOREN, Yehuda: *1 The BellKor Solution to the Netflix Grand Prize*. 2009

[Linden u. a. 2003] LINDEN, Greg ; SMITH, Brent ; YORK, Jeremy: Item-to-Item Collaborative Filtering / IEEE Computer Society. JANUARY 2003. – Forschungsbericht

[LOHR 2009] LOHR, STEVE: Netflix Awards 1 Million Dollar Prize and Starts a New Contest. In: *The New York Times* (2009). – URL http://bits.blogs.nytimes.com/2009/09/21/netflix-awards-1-million-prize-and-starts-a-new-contest/

[LOUPPE 2010] LOUPPE, Gilles: *Collaborative filtering: Scalable approaches using restricted Boltzmann machines*. 2010

[Michael Hunger 2012] MICHAEL HUNGER, Andreas K.: *The Spring Data Neo4j Guide Book*. 2012. – URL http://static.springsource.org/spring-data/data-neo4j/docs/2.0.0.RELEASE/reference/multi/index.html

[Rails 2013a] RAILS: *Active Record Query Interface*. Webseite. 2013. – URL http://guides.rubyonrails.org/active_record_querying.html

[Rails 2013b] RAILS: *Getting Started with Engines*. Webseite. 2013. – URL http://guides.rubyonrails.org/engines.html

[Rails 2013c] RAILS: *Ruby on Rails Guides*. Webseite. 2013. – URL http://guides.rubyonrails.org/

[Technology 2012] TECHNOLOGY, Neo: *The Neo4j Manual*. 2012. – URL http://docs.neo4j.org/chunked/stable/index.html

[Yildirim und Krishnamoorthy 2008] YILDIRIM, Hilmi ; KRISHNAMOORTHY, Mukkai S.: A random walk method for alleviating the sparsity problem in collaborative filtering. In: PU, Pearl (Hrsg.) ; BRIDGE, Derek G. (Hrsg.) ; MOBASHER, Bamshad (Hrsg.) ; RICCI, Francesco (Hrsg.): *RecSys*, ACM, 2008, S. 131–138. – URL http://dblp.uni-trier.de/db/conf/recsys/recsys2008.html#YildirimK08. – ISBN 978-1-60558-093-7

Literaturverzeichnis

Abbildungsverzeichnis