Universität Siegen
Fachbereich Wirtschaftswissenschaften
Institut für Wirtschaftsinformatik

Benutzer-Matching auf Basis automatischer Textanalyse

Ein Ansatz zur Ähnlichkeitsbestimmung von Benutzern durch Dokumentenanalyse für das ExpertFinder Framework

Diplomarbeit

von

Jochen Battenfeld
Matr.-Nr. 561866

Prüfer: Prof. Dr. Volker Wulf
Prof. Dr. sc. techn. Manfred Grauer

Februar 2005

Hiermit versichere ich, dass ich diese Diplomarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Siegen, den 15.02.2005

Jochen Battenfeld

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

„Was wir wissen ist ein Tropfen, was wir nicht wissen ein Ozean" (Sir Isaac Newton). Jeder Mensch wird in seinem beruflichen oder privaten Umfeld mit Problemstellungen konfrontiert, die sein Wissen übersteigen und bei denen er Hilfe benötigt. Ein möglicher Umgang mit diesem Problem ist die Hinzuziehung von Experten, d.h. Personen, die mit der jeweiligen Problemstellung vertraut sind und Empfehlungen über sinnvolle Lösungsstrategien geben können. Bei der Identifikation von Experten können Recommender Systeme eine Person unterstützen. Recommender Systeme haben die Aufgabe, zu einer bestimmten Problematik qualifizierte Personen zu identifizieren und dem Anfrager vorzuschlagen. Um aus einer Menge von Personen die geeigneten Experten zu selektieren ist es notwendig, aussagekräftige Daten über die im Recommender System verwalteten Personen zu identifizieren. Eine Möglichkeit, diese Daten zu erlangen, ist, die von jeder Person verfassten Dokumente, seien es wissenschaftliche Arbeiten, Projektbeschreibungen, Protokolle oder ähnliches auf ihren Inhalt hin zu untersuchen.

Genau diese Problemstellung stellt die Motivation der vorliegenden Arbeit dar. Es wird ein Verfahren entwickelt, das Textdateien eines Benutzers automatisch analysiert und auf Basis der Ergebnisse die verschiedenen Personen hinsichtlich der Ähnlichkeit ihrer Textproduktion vergleicht. Die zu einem Benutzerprofil oder einer Anfrage am besten passenden Benutzer werden daraufhin dem Anfrager vorgeschlagen. Das Verfahren soll in das Recommender System ExpertFinder Framework integriert werden, um dessen Ergebnisse zu optimieren.

Kapitel 2 gibt einen Überblick über den Stand der Forschung im Bereich dieser Diplomarbeit. Dabei wird zum einen auf den Bereich Recommender Systeme eingegangen, der den Rahmen für das entwickelte Verfahren darstellt. Zum anderen sind gängige Verfahren zur automatischen Textanalyse Gegenstand dieses Kapitels, da diese die Informationen extrahieren, welche später die Grundlage für die Generierung von Empfehlungen bilden.

In Kapitel 3 wird ein Konzept für ein Benutzer-Matching auf Basis automatischer Textanalyse vorgestellt. Es wird ein zweistufiges Verfahren konzipiert, welches zunächst einen Text automatisch hinsichtlich seines Inhalts analysiert und im zweiten Schritt die Ergebnisse aus diesem Arbeitsschritt auf ihre Ähnlichkeit hin untersucht.

Gegenstand von Kapitel 4 ist die prototypische Implementierung des entwickelten Verfahrens. Der Prototyp besteht aus drei Teilen: Der erste Teil generiert aus den Texten eines Benutzers ein Benutzerprofil, das dessen Interessen abbilden soll. Der zweite Teil

besteht aus der persistenten Speicherung der Benutzerprofile in einer Datenbank. Im dritten Teil werden die gespeicherten Benutzerprofile gematcht, d.h. es werden Ähnlichkeiten zwischen den Benutzerprofilen ermittelt, und anschließend werden auf Basis der Ergebnisse Empfehlungen generiert.

Kapitel 5 beschreibt die Evaluation des Prototyps. Ziel dieser Evaluation ist es, zum einen, die Ergebnisgüte des entwickelten Verfahrens zu bestimmen und zum anderen die Performance des Algorithmus zu ermitteln.

Schließlich wird in Kapitel 6 das entwickelte Verfahren diskutiert. Hierbei werden die Verwendbarkeit des Verfahrens sowie mögliche Problemquellen dargestellt. Zum Abschluss werden einige Verbesserungsmöglichkeiten für das Verfahren aufgezeigt.

2 Stand der Forschung

Dieses Kapitel gibt einen Überblick über den Stand der Forschung im Kontext dieser Arbeit. Zunächst wird in Abschnitt 2.1 der Begriff „Recommender System“ definiert und eingeordnet. Daraufhin wird anhand ausgesuchter Beispiele ein Überblick über die Funktionsweise solcher Systeme gegeben. Ein Überblick über Verfahren der automatischen Textanalyse ist Gegenstand von Kapitel 2.2.

2.1 Recommender Systeme

Der Begriff „Recommender System“ wird in diesem Kapitel definiert, eingeordnet sowie seine Funktionalität anhand ausgewählter Referenzprojekte beschrieben. Abschnitt 2.1.1 befasst sich mit dem Bereich Wissensmanagement, in welchem Recommender Systeme anzusiedeln sind. In diesem Zusammenhang wird auch der Begriff „Wissen“ näher betrachtet, da er im Kontext dieser Arbeit eine große Bedeutung innehat. In Abschnitt 2.1.2 erfolgt eine Definition des Begriffs Recommender System sowie eine Beschreibung des zugrunde liegenden Empfehlungsprozesses. Abschnitt 2.1.3 zeigt anhand von Beispielen unterschiedliche Typen von Recommender Systemen auf und beschreibt ihre Funktionalität. Schließlich wird in Abschnitt 2.1.4 das ExpertFinder Framework vorgestellt, in welches das entwickelte Verfahren integriert werden soll.

2.1.1 Wissensmanagement

Der Begriff „Wissensmanagement“ deckt ein sehr weites Feld von IT-Systemen und Strategien ab, die Menschen bei der Suche nach Informationen und der Wissensbildung unterstützen sollen. Zu diesen Systemen zählen auch Recommender Systeme.

In jüngerer Vergangenheit erfolgte eine kontinuierliche Entwicklung hin zur Wissensgesellschaft. Die klassischen Produktionsfaktoren Arbeit, Kapitel und Boden gelten als nahezu ausgereizt, der Faktor Wissen hingegen gewinnt immer mehr an Bedeutung und stellt gegenwärtig einen zentralen Parameter dar, um sich im Wettbewerb vorteilhaft zu positionieren. Für das Verständnis des Begriffs Wissen ist es hilfreich, einige grundsätzlich damit in Verbindung stehende Begriffe voneinander abzugrenzen. *Information* bedeutet im umgangssprachlichen Sinne Kenntnisse über Sachverhalte oder Vorgänge, wo hingegen die Betriebswirtschaft unter Information zweckorientiertes bzw. zielgerichtetes Wissen versteht. Informationen werden mittels Zeichen dargestellt. Wenn Informationen durch Zeichen zum Zweck der Verarbeitung erstellt werden, bezeichnet man diese als

Daten. Wenn Informationen von einem Benutzer an einen Anderen weitergegeben werden, nennt man sie *Nachrichten* [StHa02].

Unter *Wissen* versteht man nach [Prob99] die Gesamtheit der Kenntnisse und Fähigkeiten, die Individuen zur Lösung von Problemen einsetzen. Man unterscheidet nach [NoTa97] Wissen grundsätzlich in zwei Kategorien, *explizites* und *implizites* Wissen. *Explizites Wissen* zeichnet sich dadurch aus, dass es formal kommunizierbar ist. Es existiert in Form von Dokumenten oder Spezifikationen, wodurch die Weitergabe dieses Wissens relativ einfach ist. *Implizites Wissen* hingegen ist relativ schwer kommunizierbar, da es persönlich und auf einen bestimmten Kontext spezifiziert ist. Diesem Bereich unterfallenfallen beispielsweise subjektive Eindrücke, die man von einem Thema gewonnen hat oder persönliche Erfahrungen. [Stew98] bezeichnet Wissen als den vierten und mittlerweile zentralen Produktionsfaktor, da er in Form von Patenten, Dokumentationen, Prozess- und Kundenwissen eines Unternehmens zu einem wichtigen Wettbewerbsfaktor avanciert ist und so maßgeblich zur Wertschöpfung des Unternehmens beiträgt. Das Problem, das beim Umgang mit dem Produktionsfaktor Wissen auftritt, ist die mangelnde Transparenz und Unstrukturiertheit dieser Ressource. Oftmals ist in Unternehmen die Zuständigkeit für das Management von Wissen nicht hinreichend geregelt. Das führt dazu, dass die Unternehmensführung nicht in ausreichendem Maße über die Kompetenzen und Fähigkeiten der eigenen Mitarbeiter informiert ist. Dies gilt sowohl für individuelle, als auch für kollektive Fähigkeiten. Diese Problematik bildet den Ansatzpunkt für das Wissensmanagement als Forschungsdisziplin.

Da es sich beim Wissensmanagement um ein relativ junges und interdisziplinäres Gebiet handelt, existieren vielfältige Definitionen dieses Begriffs. An dieser Stelle soll die erste relativ umfassende Definition des Begriffs „Wissensmanagement“ aus dem Jahr 1986 von Wiig vorgestellt werden.

> „Knowledge Management is the semantic, explicit, and deliberate building, renewal, and application of knowledge to maximize an enterprise’s knowledge-related effectiveness and returns from its knowledge assets.“[KaTe01]

Diese Definition beschreibt Wissensmanagement ganz allgemein aus Sicht eines Unternehmens.[1] In Abbildung 1 sind die Kernprozesse des Wissensmanagements schematisch dargestellt. Sie reichen von der Erarbeitung der Wissensziele, die durch Erkennung von Wissenslücken sowie vorhandenem Wissen die Strategie des Wissensmanagements vorgeben, über Identifikation und Erwerb bis hin zur Nutzung und Bewahrung von Wissen.

[1] Einen Überblick über weitere Definitionen im Bereich des organisationalen Wissensmanagement bietet [AlLa03].

Recommender Systeme haben die Aufgabe, in einer Organisation bestehendes Wissen zu verteilen und gleichzeitig dieses Wissen zu bewahren. Die automatische Textanalyse verfolgt die Zielsetzung, Wissen zu identifizieren. Sie soll dabei helfen, die riesige Flut an Informationen, die im Zeitalter elektronischer Dokumente herrscht, zu kanalisieren und die Transparenz des Wissens zu erhöhen. Im Kontext dieser Arbeit sollen diese beiden Disziplinen kombiniert werden. Das Wissen von Personen soll auf der einen Seite durch die Textanalyse identifiziert werden, auf der anderen Seite soll durch die Funktionalität des Recommender Systems dieses Wissen transparenter gemacht und nach Bedarf verteilt werden.

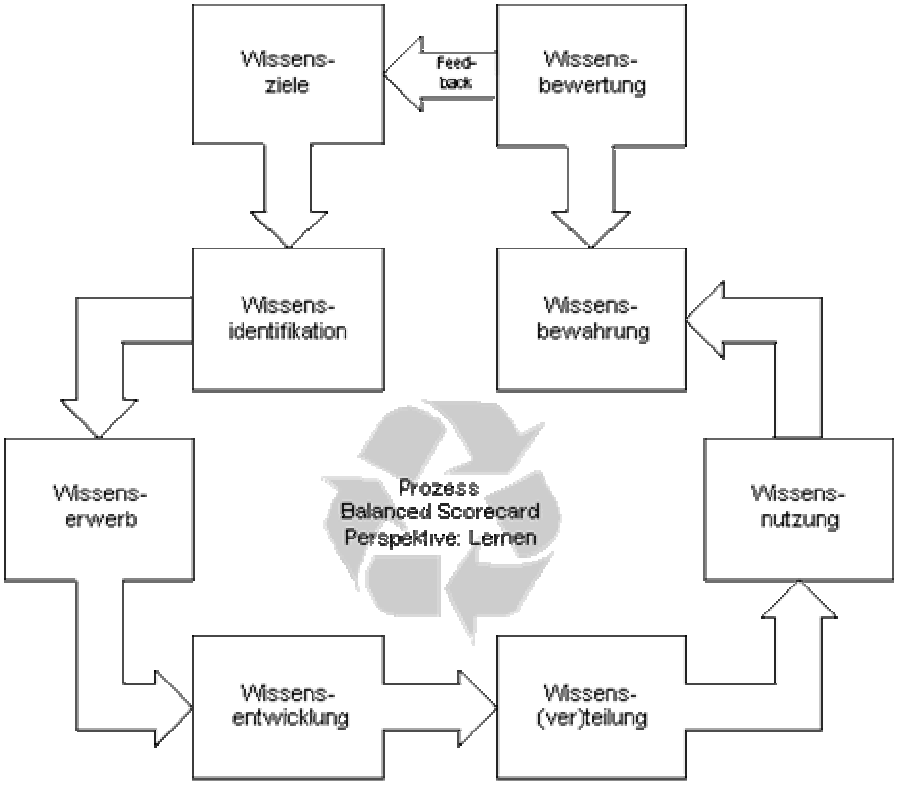


Abbildung 1: Kernprozesse des Wissensmanagements [Prob99]

2.1.2 Definition von Recommender Systemen und zugrunde liegendem Empfehlungsprozess

Recommender Systeme oder Empfehlungssysteme haben die Zielsetzung, Empfehlungen für einen Benutzer automatisch zu generieren. [Runt00] unterscheidet grundsätzlich zwei Arten von Recommender Systemen, nicht personalisierte und personalisierte Systeme. Nicht personalisierte Systeme zeichnen sich dadurch aus, dass die erzeugten Empfehlungen für alle Benutzer gelten. Jeder Benutzer erhält also dieselbe Empfehlung. Beispiele für solche Systeme sind Bestsellerlisten oder Charts. Personalisierte Recommender Sys-

teme hingegen richten ihre Empfehlungen speziell an einzelne Benutzer und versuchen die Empfehlungen nach seinen Bedürfnissen zu erstellen. Im Kontext dieser Arbeit ist der letztgenannte Typ von Systemen von Interesse und mit dem Begriff Recommender System ist im Folgenden dieser personalisierte Typ gemeint.

Im Rahmen der Entwicklung des ersten Recommender Systems Tapestry aus dem Jahr 1992 wurden Recommender Systeme als Collaborative Filtering bezeichnet. Die Bezeichnung Recommender System wurde von [ReVa97] eingeführt und wird auch in dieser Arbeit weiterhin verwendet. Nach [ReVa97] assistieren Recommender Systeme den Menschen bei dem natürlichen Prozess des Heranziehens von Empfehlungen von Anderen. Zusätzlich können Recommender Systeme diesen Prozess durch zusätzliche Funktionalität ergänzen.

Die Motivation, die hinter der Entwicklung von Recommender Systemen steht, ist die Bewältigung der immer größer werdenden Menge an verfügbarem Wissen und die damit einhergehende Menge zu treffender Entscheidungen. Eine mögliche Lösung besteht in Empfehlungen von Akteuren, die mit der entsprechenden Thematik vertraut sind. Bei einer Empfehlung eines Experten ist die Wahrscheinlichkeit relativ hoch, dass sie sich schon bewährt hat und bei der Suche nach einer Problemlösung hilfreich ist.

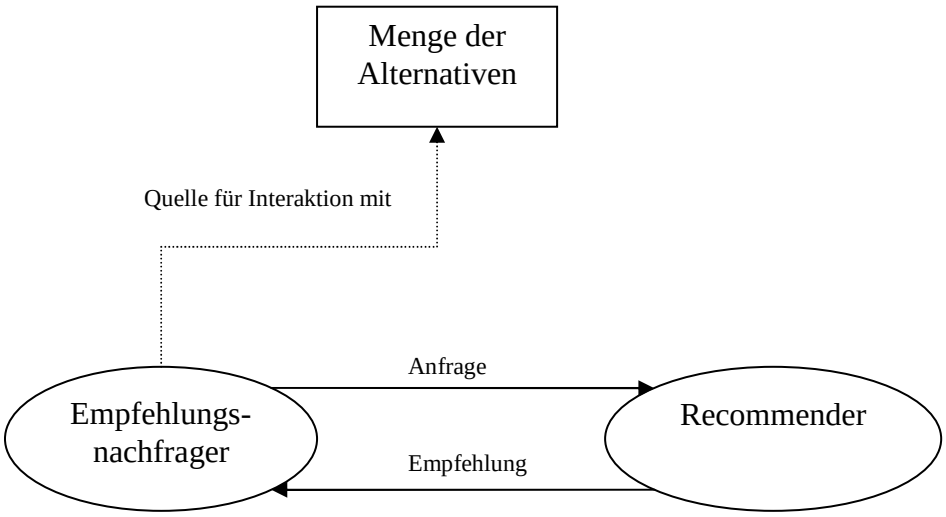


Abbildung 2: Recommendation Prozess [TeHi01]

Der Empfehlungsprozess, der von Recommender Systemen geleistet werden soll, läuft folgendermaßen ab: Der Benutzer hat ein Problem und fragt nach einer Empfehlung. Der Recommender, in diesem Fall also das System, sucht anhand der Problemstellung nach einer Empfehlung, welche dem Benutzer weiterhelfen kann. Diese Empfehlung kann beispielsweise in Form von Dokumenten oder auch durch Verweise auf Personen mit Kom-

petenzen im Problembereich erfolgen. Der Empfehlungsprozess ist in Abbildung 2 dargestellt[2].

2.1.3 Typen von Recommender Systemen

Dieses Kapitel beschreibt die Funktionsweise von Recommender Systemen. Im Kontext dieser Arbeit ist von besonderem Interesse, auf welcher Informationsbasis die Empfehlungen generiert werden. Die verschiedenen Systeme beziehen ihre Informationen aus unterschiedlichen Quellen. Es zeichnen sich drei Kategorien von Informationsquellen ab, die von vielen Systemen verwendet werden[3]:

1) Benutzerhistorische Daten wie Bookmarks können Hinweise auf die Interessen und Kompetenzen des Benutzers geben. Das System Siteseer [RuPo97] arbeitet nach diesem Prinzip.
2) Dokumente, die ein Benutzer geschrieben oder gelesen bzw. kommentiert hat, sind ebenfalls hinsichtlich der Qualifikationen eines Benutzers aussagekräftig. Die Art der Dokumente ist in diesem Zusammenhang durchaus variabel. Systeme wie PHOAKS [Terv97] und GroupLens [Resn94] arbeiten mit Newsgroups-Artikeln. Andere Systeme werten den Inhalt von E-Mails aus (siehe [McBr03]). Eine weitere Möglichkeit, die auch diese Arbeit motiviert, liegt in der Analyse von beliebigen Textdokumenten, die ein Benutzer verfasst hat.
3) Eine dritte Kategorie der Gewinnung von Quelldaten liegt in der Analyse des sozialen Umfelds eines Benutzers. Die grundlegende Idee dahinter ist, dass Menschen, die schon miteinander kooperiert haben, dies auch zukünftig wahrscheinlich tun werden. Der Expertise Recommender von [McAc00] funktioniert nach diesem Prinzip.

Da das Spektrum an Recommender Systemen mittlerweile sehr breit gefächert ist, soll die Beschreibung einiger repräsentativer Beispiele einen Querschnitt der Funktionsweisen von Recommender Systemen geben.

Mit Hilfe des Recommender Systems *PHOAKS* (People Helping One Another Know Stuff) werden Informationen aus Newsgroups gefiltert sowie dynamische Web-Seiten generiert und organisiert. Gegenwärtig werden mit PHOAKS Online-Empfehlungen und FAQs behandelt. Das PHOAKS Konzept besteht aus folgenden Prozessen:

[2] Bei [TeHi01] ist im Recommendation Process zusätzlich ein Präferenzverwalter involviert, der persönliche Vorlieben des Benutzers berücksichtigt. Dieser wird hier nicht dargestellt, da es sich um eine Extrafunktion handelt.

[3] Es gibt auch Systeme, die mehreren Kategorien zuzuordnen sind, wie beispielsweise Fab von [BaSh97]. Man spricht in diesem Fall von hybriden Systemen.

- **Suchen**: Durchsucht das Usenet anhand von Mustern und liefert die kontextbezogenen Informationen, die sich um das Muster herum befinden.
- **Kategorisieren**: Auf das Suchergebnis wird ein Algorithmus angewendet, der alle Instanzen nach vorher bestimmten Regeln klassifiziert.
- **Verwenden**: Die im zweiten Schritt gewonnenen Informationen werden beispielsweise in einer Datenbank gespeichert oder in eine dynamische Webseite eingebaut.

Im Gegensatz zu vielen anderen Systemen, die von konformen Benutzern ausgehen, bietet PHOAKS zwei verschiedenen Benutzerrollen an, den Anbieter und den Empfänger von Empfehlungen. Weiterhin zeichnet sich das System durch die Wiederverwendung von Empfehlungen aus bereits bestehenden Konversationen aus [Terv97].

Ein System, das soziale Netzwerke sucht, visualisiert und strukturiert heißt *Referral Web*. Dieses System legt besonderes Gewicht darauf, soziale Gemeinschaften bzw. Communities zu erkennen oder zu schaffen. Referral Web gibt keine anonymen Empfehlungen, sondern personalisierte Ratschläge über eine Kette von bekannten Personen. Das System baut Benutzerprofile anhand sozialer Kontakte des Benutzers auf und ist darüber hinaus nicht auf einzelne Domänen spezialisiert. Die Informationen für die Identifizierung der sozialen Netzwerke bezieht Referral Web aus Untersuchung von Links, Ermittlung von Co-Autoren und anderen Hinweisen auf persönliche Zusammenarbeit [Kaut97].

Das Recommender System *Fab* wurde von der Stanford University als Teil des Stanford University Library Projekts entwickelt, um dem Internet-Benutzer beim Umgang mit der immensen Informationsmenge zu helfen. Im Kontext von Fab unterscheidet [BaSh97] im Wesentlichen zwei Arten von Empfehlungen, rein *inhaltsbasierte* und rein *gemeinschaftsbasierte*. Beim inhaltsbasierten Ansatz beruhen die Empfehlungen auf einem Benutzerprofil, das sich durch die Analyse von Texten ergibt, die der Benutzer in der Vergangenheit bewertet hat. Dieser Ansatz hat seinen Ursprung im Information Retrieval (IR). IR bezeichnet zusammenfassend alle Verfahren, die mit der Aufbereitung, Speicherung und Wiedergewinnung von gespeicherten Informationen zu tun haben und wurde ursprünglich zum Widerauffinden wissenschaftlicher Literatur entwickelt [Ferb03]. Problematisch an diesem Ansatz ist zum einen, dass in manchen Domänen dieses Verfahren aufgrund der Beschaffenheit der Medien zu nicht-befriedigenden Ergebnissen führt. Dies betrifft zum Beispiel die Bereiche Musik und Film, die inhaltlich nur schwer erschlossen werden können. Andererseits berücksichtigt dieses Verfahren andere Faktoren wie zum Beispiel Performance oder Design nicht, die aber für einen Benutzer durchaus zur Empfehlungsentscheidung beitragen könnten. Ein weiteres Problem liegt möglicherweise in

Variationen der Terminologie zwischen verschiedenen Benutzern. Das Matching zwischen zwei Benutzern ist nur erfolgreich, wenn diese die gleiche Fachsprache benutzen. Unter Matching versteht man in diesem Zusammenhang eine Ähnlichkeitsbestimmung zwischen mehreren Benutzern. Der gemeinschaftsbasierte Ansatz generiert seine Empfehlungen auf Basis der Ähnlichkeit von Benutzern. Hier werden nicht Dokumente analysiert, sondern nur die Profile von Benutzern verglichen. Die Probleme, die beim inhaltsbasierten Ansatz aufgetreten sind, sind hier nicht von Bedeutung, allerdings bringt dieses Verfahren eigene Schwierigkeiten mit sich. Wenn die Datenbank beispielsweise durch ein Dokument ergänzt wird, so kann das System dieses Dokument solange nicht in die Empfehlung mit einbeziehen, bis Referenzen durch Bewertungen anderer Benutzer über das Dokument eingegangen sind. Ein anderes Problem kann entstehen, wenn ein Benutzer ein eher außergewöhnliches Profil hat, wodurch der Vergleich mit den anderen Benutzern schwierig wird. Fab ist ein hybrides System, das beide Ansätze, den inhaltsbasierten und den gemeinschaftsbasierten Ansatz, kombiniert.

Ein auf Webseiten basierendes Recommender System, das die Bookmarks eines Benutzers untersucht, um seine Präferenzen in Erfahrung zu bringen, ist *Siteseer*. Dieses System basiert auf der Annahme, dass man die Bookmarks eines Benutzers als Interessenindikator ansehen kann. Zudem versucht das System Schlüsse aus dem Gruppierungsverhalten des Benutzers zu ziehen, um relevante Themen zu identifizieren. Siteseer erlernt einerseits die Benutzerpräferenzen des einzelnen und andererseits für jede Webseite die verschiedenen Communities von Nutzern. Das Ergebnis besteht aus personalisierten Empfehlungen von Online-Informationen und Webseiten, organisiert nach der Verzeichnisstruktur des Benutzers [RuPo97].

GroupLens ist ein Recommender System, das Artikel aus dem Usenet als Quelle für Empfehlungen benutzt. Das Usenet schafft die Illusion eines Aushangbrettes, das weltweit verfügbar ist. GroupLens ist ein verteiltes System, das Einschätzungen von Benutzern sammelt, benutzt und verbreitet, um Interessen anderer Benutzer bezüglich bestimmter Artikel vorauszusagen. Die Architektur von GroupLens ist in Abbildung 3 dargestellt. Hierbei wird teilweise die Usenet Architektur benutzt, welche durch die fünf oberen Server dargestellt ist. Artikel werden über ein Netz von Servern, Nutzern auf der ganzen Welt zugänglich gemacht. GroupLens ergänzt diese Architektur durch eine Art von Brokern, die die Aufgabe haben, Benutzerdaten und Bewertungen weiterzuleiten. Solche Broker werden als Better Bit Bureau (BBB) bezeichnet. BBBs sammeln Beurteilungen von

Benutzern über Artikel, die sie gelesen haben. Diese Daten werden dazu benutzt, um Empfehlungen für andere Benutzer zu generieren [Kons97, Resn94].

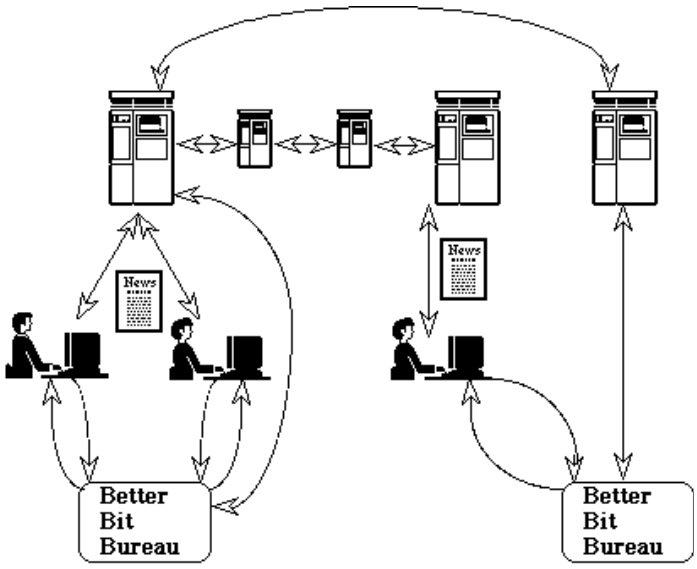


Abbildung 3: GroupLens Architektur [Resn94]

Der *Expertise Recommender* (ER) [McAc00] wurde auf Basis einer Studie in einem mittelständischen Software-Unternehmen zur Untersuchung der Vorgehensweisen bei der Suche nach Informationen entwickelt, um das Verhalten und die Ergebnisse von Benutzern bei der Informationssuche zu verbessern. ER zeichnet sich durch eine offene und flexible Architektur aus, wodurch die Anwendbarkeit auch in heterogenen organisationalen Umgebungen gewährleistet sein soll. Das System setzt dort an, wo eine Schlüsselperson oder ein Experte nicht verfügbar ist. ER sucht in diesem Fall alternative Experten, die beispielsweise aus einem anderen Umfeld des Unternehmens kommen, aber bezogen auf die jeweilige Problemstellung über Kompetenzen verfügen. ER stellt dem Benutzer durch die Verwendung bestimmter Filter auch in sozialer Hinsicht, d.h. unter Berücksichtigung seiner bisherigen Ansprechpartner, auf ihn zugeschnittene Empfehlungen aus [McAc00, McDo03]. Der Prozessablauf beginnt mit einer Anfrage eines Benutzers zu einer bestimmten Problemstellung. Darüber hinaus hat der Benutzer die Möglichkeit, bestimmte Filter zu setzen, welche im Ergebnis berücksichtigt werden. Das System generiert daraufhin auf Basis der Anfragedaten eine Empfehlung, die mögliche Experten zum Problembereich vorschlägt [McAc00].

Die schon angesprochene Verwendung von Filtern resultiert aus der Erkenntnis, dass soziale Beziehungen ein wichtiger Faktor bei der Zusammenarbeit in Unternehmen sind. Nach [McDo03] besteht das Ziel hierbei in der Steigerung der Sensitivität der Systeme für soziale Beziehungssysteme. Diese sozialen Netzwerke repräsentieren Gruppen von Men-

schen sowie die zwischen ihnen bestehenden Beziehungen. Es werden nach [McDo03] zwei verschiedene Filterverfahren unterschieden: *Departmental Matchin*g und *Social Network Matching*. Es besteht auch die Möglichkeit, auf die Filter zu verzichten. In diesem Fall wird für die Berechnung des Ergebnisses nur die Anfrage des Benutzers berücksichtigt. Dem Departmental Matching liegt der Work Group Graph (WGG) zugrunde, der ein Netz von logischen Arbeitsgruppen und Arbeitsgemeinschaften unabhängig von der organisationalen Struktur des Unternehmens abbildet. Der WGG wurde auf Basis von ethnographischen Methoden wie Interviews, Beobachtung von Teilnehmern und Sammeln von Artefakten erstellt, was mit hohem Aufwand verbunden ist. Social Network Matching arbeitet auf Basis der Annahme, dass man besser mit Menschen arbeitet, mit denen man ohnehin in sozialem Kontakt innerhalb des Unternehmens steht. Das zugrunde liegende Netzwerk, das Successive Pile Sort (SPS) Social Network, ordnet die Benutzer anhand ihres sozialen Umgangs, der auch unabhängig von fachlicher Zusammenarbeit sein kann. Ein SPS-Netzwerk entsteht durch die Sammlung und Analyse von persönlich erstellten Rangfolgen der Mitarbeiter in Bezug auf die Kooperation mit Kollegen. Diese Rangfolgen werden durch Aggregation und multidimensionale Skalierung zum SPS-Netzwerk zusammengefasst.

2.1.4 ExpertFinder Framework

Das ExpertFinder Framework ist ein Expertise Recommendation System [Reic04, Beck04], das der zunehmenden Bedeutung von sozialen Netzwerken menschlicher Akteure in Arbeitsprozessen Rechnung tragen soll. Dieses Framework wurde ursprünglich am Fraunhofer Institut für Angewandte Informationstechnologie (FhG-FIT), Sankt Augustin und am Institut für Wirtschaftsinformatik der Universität Siegen entwickelt, um eine E-Learning-Plattform zur Weiterbildung Berufstätiger zu ergänzen. Die Architektur des ExpertFinder Frameworks zielt darauf ab, an möglichst beliebige Plattformen angebunden zu werden, in denen Benutzer verwaltet werden. Ziel ist es dabei, eine bessere Vernetzung der Benutzer zu erreichen, die im Allgemeinen durch Plattformen wie Lernportale nicht unterstützt wird. Hierdurch sollen kooperative Prozesse wie Arbeiten oder Lernen angestoßen und gefördert werden.

Das Framework bildet eine Möglichkeit zur Vernetzung von Akteuren aufgrund ähnlicher Interessen oder Qualifikationen. Auf Anfragen werden einem Akteur mittels Empfehlungen des ExpertFinders andere Benutzer vermittelt, die im Problembereich der Anfrage Expertenwissen besitzen und somit hilfreich sein könnten. Um zu den Empfehlungen zu gelangen verwendet der ExpertFinder verschiedene Matching-Verfahren, um geeignete

Benutzer zusammenzuführen. Zwei bei [Reic04] schon implementierte Verfahren sollen hier kurz erwähnt werden. *Profile Matching* ist ein Verfahren, das die Benutzer anhand ihres Ausbildungsstandes sowie ihrer Berufserfahrung vergleicht. Ein anderes Verfahren namens *History Matching* führt ein Matching anhand dynamischer Benutzerhistorien aus. Es wird vom System hierzu aus den über die Anwendungsplattform abgerufenen Inhalten (Historien) ein Benutzerprofil erstellt, dem die wahrscheinlichen Interessen und Kenntnisse der Benutzer zu entnehmen sind.

Diese Matching-Strategien sollen optimale Zusammenarbeit und Hilfe unter den Benutzern gewährleisten. Die einzelnen Matching-Verfahren sind in Form von Modulen implementiert und können beliebig durch neue Verfahren ergänzt werden. Die Architektur des ExpertFinder Frameworks ist in Abbildung 4 dargestellt und besteht aus drei wesentlichen Teilbereichen, dem eigentlichen ExpertFinder (1), einem Java-Servlet als Anbindung zum Client (2) sowie der Anbindung zur Anwendungsplattform.

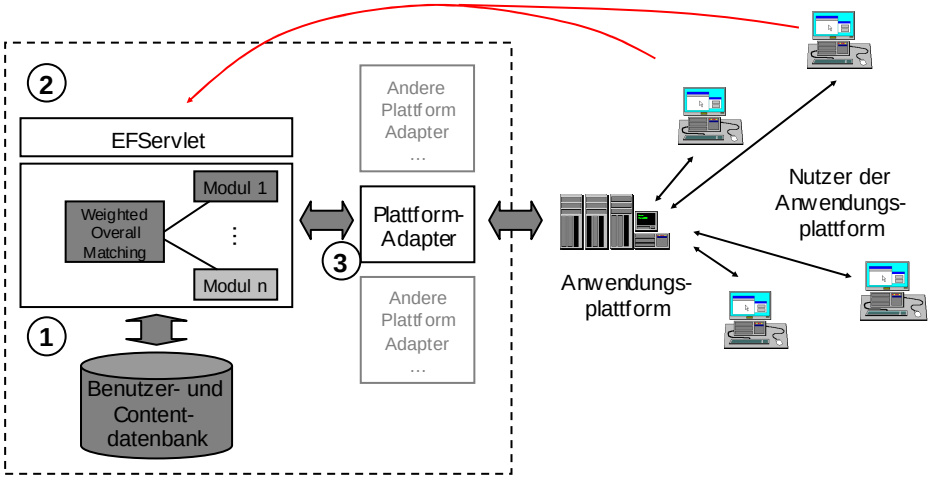


Abbildung 4: Architektur des ExpertFinder Frameworks [Reic04]

Der ExpertFinder selbst besteht aus einer Reihe interner Datenbanken, die Benutzerinformationen sowie Inhalte zwischenspeichern. Der ExpertFinding-Prozess, d.h. das Finden von geeigneten Benutzern auf Anfragen, wird von den Matching-Modulen geleistet. Die Ergebnisse der einzelnen Module werden gemäß den Präferenzen des Benutzers gewichtet und auf Basis dieser Daten wird im WeightedOverallMatching-Modul eine Empfehlung generiert. Dieses Modul soll die Ergebnisse der einzelnen Module sammeln und anhand der vom Benutzer bestimmten Gewichtungen einem Gesamtergebnis zu aggregieren (siehe Abbildung 5).

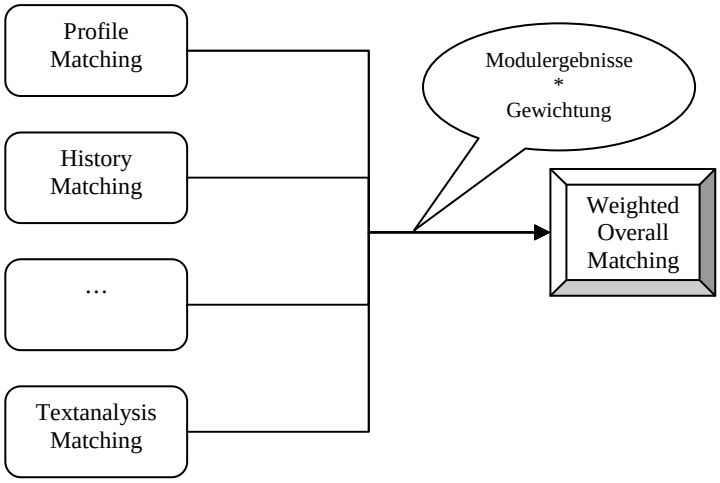


Abbildung 5: Matching-Module des ExpertFinders

Die Verbindung zum Client erfolgt über ein Java Servlet, welches die Benutzerschnittstelle erzeugt und mit dem ExpertFinder interagiert. Um die Anwendbarkeit bezüglich der Anwendungsplattform flexibel zu gestalten, werden Plattformadapter zwischengeschaltet, deren Aufgabe darin besteht, plattformspezifische Informationen in ein einheitliches und vom ExpertFinder verwendbares Format umzuwandeln.

Das ExpertFinder Framework wird zurzeit mit der Absicht, es in einem allgemeineren organisationalen Kontext einzusetzen, weiterentwickelt. Auf diese Weise ist es nicht mehr so stark an die Anwendungsplattform gebunden, sondern bezieht seine Informationen auch aus darüber hinaus gehenden Quellen. Die Zielsetzung besteht im Einsatz des ExpertFinder Frameworks in verteilten sowie mitarbeiterstarken Organisationen.

2.2 Methoden zur automatischen Textanalyse

Dieses Kapitel gibt einen Überblick über die grundlegenden Verfahren automatischer Textanalyse. Viele der Verfahren stammen aus dem Bereich Text Mining, einer relativ jungen wissenschaftliche Disziplin, die aus dem Bereich des Data Mining[4] entstanden ist. [Sull01] definiert Text Mining als die Wissenschaft, Informationen und Wissen aus Texten zu extrahieren. Es geht im Wesentlichen darum, große Bestände von unstrukturierten Textdokumenten auf ihren Inhalt hin zu untersuchen und die gefundenen Informationen in automatischer Form zu extrahieren. Die zentrale Aufgabe von Text Mining Applikationen ist es, wichtige Textfragmente oder Schlüsselwörter in Texten zu identifizieren, und diese dann in logischer Weise zu strukturieren und miteinander zu verknüpfen. Text Mining ist als interdisziplinäre Domäne aus einer Reihe von anderen wissenschaftlichen Gebieten

[4] Mit Data Mining bezeichnet man das Lösen von Problemen durch die Analyse von großen und komplexen Datenmengen. Ziel ist es, Beziehungen zwischen oder Muster in den einzelnen Datensätzen zu finden.

wie dem Information Retrieval [SaMc87] und der Computerlinguistik [Cars01] hervorgegangen. Text Mining bildet jedoch nur einen kleinen Teil dieser beiden wissenschaftlichen Bereiche.

In den meisten Fällen wird zur automatischen inhaltlichen Erschließung von Texten nach besonderen Begrifflichkeiten gesucht, welche man als Fachterminologie bezeichnet. Fachterminologien entstehen, wenn Fachwissen strukturiert werden soll. Dies geschieht dadurch, dass verschiedene Terminologien der gleichen Fachrichtung vereinheitlicht werden. In der Regel bildet sich eine Terminologie durch Sprachnormierung heraus, was als Language Engineering bezeichnet wird. Triebkräfte auf diesem Gebiet sind die Deutsche Industrie Norm (DIN) und die International Standards Organisation (ISO).

In Abschnitt 2.2.1 werden linguistische Verfahren, die sich durch einen sprachwissenschaftlichen Hintergrund auszeichnen, beschrieben. Danach wird in Abschnitt 2.2.2 auf die wohl intuitivsten Methoden eingegangen, welche aus dem Bereich Statistik stammen und mit relativen und absoluten Häufigkeiten von Wörtern oder Wortformen arbeiten. Künstliche neuronale Netze, die Gegenstand von Abschnitt 2.2.3 sind, bilden eine weitere Methode, um Texte zu erschließen, die sich sehr stark an der Funktionalität des menschlichen Gehirns orientiert. Abschnitt 2.2.4 befasst sich mit Verfahren, die auf Basis von schon bestehenden Vektoren von Begriffen arbeiten. Als letzte Kategorie von Verfahren werden in Abschnitt 2.2.5 Heuristiken behandelt, welche nicht so stark wissenschaftlich geprägt sind, sondern die sich eher aus praktischen Gesichtspunkten heraus entwickelt haben. Zum Abschluss werden die einzelnen Verfahren in Kapitel 2.2.6 auf ihre Vor- und Nachteile hin untersucht.

2.2.1 Linguistische Methoden

Die Linguistischen Methoden haben ihren Ursprung in der Sprachwissenschaft, wobei der Bereich Terminologielehre im Kontext dieser Arbeit von Interesse ist [Scha01,Wüst91]. Terminologielehre ist die Lehre von Begriffen und ihren Benennungen im Bereich der Fachsprachen (DIN 2342). [Wüst91] definiert allgemeine Terminologielehre als „eine linguistisch-pragmatische Disziplin, die auf internationaler Ebene durch Angleichung der Begriffe, Sinnformen und Schreibungen die nationalen Fachsprachen einander angleicht.“ Ziel dieser Disziplin ist es, Fachsprachen zu schaffen, um eine eindeutige und widerspruchsfreie Kommunikation innerhalb eines Fachbereiches zu ermöglichen. Die Terminologielehre soll das Funktionieren genau dieser Kommunikation unterstützen und sichern.

Ein Indiz für die Bedeutung, welche die Sprachwissenschaft für die Informatik gewonnen hat, ist in der immer stärkeren Verbreitung der Computerlinguistik zu sehen. Computerlinguistik ist das Fachgebiet, welches sich mit der maschinellen Verarbeitung von natürlicher Sprache beschäftigt [Cars01, siehe auch Haus00, Schm92]. In diesem Bereich der Informatik manifestiert sich die Verwandtschaft zwischen den beiden Fachgebieten Sprachwissenschaft und Informatik.

In diesem Kapitel werden exemplarisch zwei zentrale Verfahren der Computerlinguistik kurz vorgestellt, die im Kontext dieser Arbeit am wichtigsten sind, Morphologie und syntaktische Analyse.

Morphologie

Die Morphologie beschäftigt sich mit dem Aufbau von Wortformen. Für die Segmentierung von Texten, d.h. die Zerstückelung in Wörter, bildet die Morphologie die Grundlage. Unter einem Morphem versteht man die kleinste selbständige Einheit der Sprache, die noch eine eigenständige Bedeutung besitzt. Segmentierte Texte dienen als Basis für weitere Methoden der Textanalyse. Morphologische Fragestellungen im engeren Sinne sind betroffen, sobald die Analyse über den reinen Vergleich von Zeichenketten hinausgeht. Die Bildung von Stammformen, Lemmatisierung genannt, und die Zerlegung von zusammengesetzten Wörtern sind Probleme, die in diesem Kontext auftauchen.

Morphologische Methoden zur Textanalyse gehen von der Annahme aus, dass Fachtermini typische morphologische Komponenten wie zum Beispiel signifikante Prä- oder Suffixe enthalten [Heid99]. Mit Hilfe regulärer Ausdrücke können nun Terme nach bestimmten Suchmustern aus dem Text herausgefiltert werden [Heye02]. In Abbildung 6 sieht man eine aus verschiedenen Affixen bestehende Morphem-Liste. Unter Affixen versteht man eine Wortsilbe, die nur zur Bildung von Worten beiträgt, aber nicht selbständig als Wort dient. Das +-Zeichen dient als Platzhalter für den Wortstamm.

Die morphologischen Verfahren stützen sich in der Regel auf umfangreiche Morphemlisten oder Wörterbücher und die Qualität der Ergebnisse hängt auch in entscheidendem Maße von der Beschaffenheit dieser Wörterbücher ab.

ab+, auf+, ent+, anti+, bi+, mega+, mikro+, multi+, radial+, semi+, ad+, ex+, in+, ko+, pro+, [...]
+grad, +heit, +nis, +schaft, +tum, +ial, +gramm, +graph, +id, +ik, +tion, +tät, +um, […], +ator

Abbildung 6: Beispiel für eine Morphem-Liste [Heid99]

Syntaxanalyse

In der Sprachlehre ist die Syntax die Lehre vom Satzbau, die Regelmenge zur Beschreibung der Syntax wird als Grammatik bezeichnet. Die syntaktische Analyse zielt auf die formale Beschreibung von Sätzen ab, wobei das Ergebnis durch eine baumartige Struktur repräsentiert wird. Die syntaktische Analyse stellt den Versuch dar, natürliche Sprachen in einer zur Backus-Naur-Form analogen Art für Programmiersprachen zu beschreiben. Die Analyse basiert hauptsächlich auf dem Scannen und Parsen der Texte.
Problematisch wird dieses Verfahren, sobald die im Text verwendete Sprache nicht mehr die Wohlgeformtheit einer Programmiersprache hat. Eher umgangssprachlich geschriebene Texte, wie zum Beispiel E-Mails oder Notizen, sind mit dieser Methode nicht in angemessener Form zu analysieren [Kamp02].

2.2.2 Statistische Methoden

Die ältesten textanalytischen Methoden kommen aus dem Bereich der Statistik. Bei diesen Verfahren werden Wörter oder Wortformen auf ihre Häufigkeit hin untersucht. Die resultierenden absoluten oder relativen Häufigkeiten werden in Ergebnislisten gespeichert und dann verglichen. Als Grundlage dienen mehrere Fachtexte auf der einen und ein allgemeinsprachlicher Referenztext[5] auf der anderen Seite. Auf beiden Seiten werden nun die Häufigkeiten der Wortformen berechnet und zueinander in Beziehung gesetzt. [Heye02] unterscheidet vier Klassen von Wortformen:

- Wortformen, die im allgemeinen Textkorpus nicht vorkommen. Diese Wortformen sind mit hoher Wahrscheinlichkeit Fachtermini.
- Wortformen, die im Fachtext relativ häufiger vorkommen als im allgemeinsprachlichen Teil. Mit gewisser Wahrscheinlichkeit handelt es sich hier um Fachbegriffe.
- Wortformen, die in beiden Texten etwa gleich häufig vorkommen. Es handelt sich hierbei wahrscheinlich um Stoppwörter wie Artikel und Präpositionen, also keine Fachbegriffe.
- Wortformen, die im allgemeinen Referenztext häufiger vorkommen als im Fachtext. Das sind im Allgemeinen keine Fachbegriffe.

Im Folgenden werden einige Verfahren aus dem statistischen Bereich näher beschrieben. Es handelt sich dabei um Konkordanzen, Markov-Modelle und Kollokationen.

Konkordanzen und Indizes

[5] Die Verwendung eines allgemeinen Referenztextes stellt nicht das gängige Verfahren dar, sondern wurde von [Heye02] eingeführt.

Unter einer Konkordanz versteht man in der Literaturwissenschaft und Philologie eine alphabetische Zusammenstellung der Begriffe eines literarischen oder religiösen Werkes mit kompletter Quellenangabe. Man kann sie also als ein Verzeichnis oder eine Statistik von Wortformen verstehen, die als Basis für das gezielte Auffinden von Textstellen verwendet werden kann. Eine andere Bezeichnung für eine Konkordanz ist Index. Man unterscheidet zwischen Wortformenkonkordanzen und lemmatisierten Konkordanzen. Erstere verzeichnet die Wortformen, wie sie im Text vorgefunden werden. Lemmatisierte Konkordanzen hingegen registrieren die Stammform des jeweiligen Wortes, was zusätzlichen Aufwand bedeutet, da die Wortformen mit Hilfe von linguistischen Verfahren auf die Stammform reduziert werden müssen. Hierfür existieren einige Verfahren unterschiedlicher Qualität. Eine Methode ist Stemming, ein Algorithmus, der nach der Art der morphologischen Methoden Wortendungen abschneidet, so dass nur der Wortstamm übrig bleibt. Auf die Verfahren zur Wortstammreduktion wird in Kapitel 3.2.1 genauer eingegangen.

Konkordanzen stellen ein relativ einfaches Mittel zur Analyse von Texten dar, um beispielsweise Indizes zu erstellen oder Suchfunktionen zu unterstützen. Unter Indizes versteht man meist alphabetische Aufstellungen von Begriffen eines Dokuments. Lemmatisierte Konkordanzen können Texte auch inhaltlich erschließen, hierzu werden die Häufigkeiten der Stammwortformen in Relation zu allgemeinsprachlichen Referenzkorpora gesetzt. Diese Referenzen bilden die allgemeine statistische Verteilung der Wörter in einer Sprache ab und sind dadurch zur Identifikation von Eigenheiten eines Fachtextes geeignet. Das Verfahren, welches man zur Schaffung von Konkordanzen verwendet, wird als Indexierung bezeichnet. Wenn das Verfahren von einem Computer ausgeführt wird, nennt man es automatische Indexierung.

Markov-Modelle

Markov-Modelle oder Markov-Ketten sind eine andere statistische Methode zur Gewinnung von Informationen über Texte, wobei hier nicht die Auftrittshäufigkeit von Wörtern, sondern die Übergangswahrscheinlichkeit zweier Morpheme für die Informationsgewinnung genutzt wird. Die zentrale Frage ist also, wie wahrscheinlich es ist, dass ein bestimmtes Wort auf ein anderes folgt.

Unter einem Markov-Modell versteht man grundsätzlich einen probabilistischen Prozess über einer endlichen Menge von Zuständen. Markov-Ketten entstehen dadurch, dass sprachliche Einheiten in Relation zu vorangegangenen sprachlichen Einheiten gesetzt

werden. Unter sprachlichen Einheiten versteht man einzelne Worte oder Morpheme. Entscheidend sind die dabei notierten Übergangswahrscheinlichkeiten.

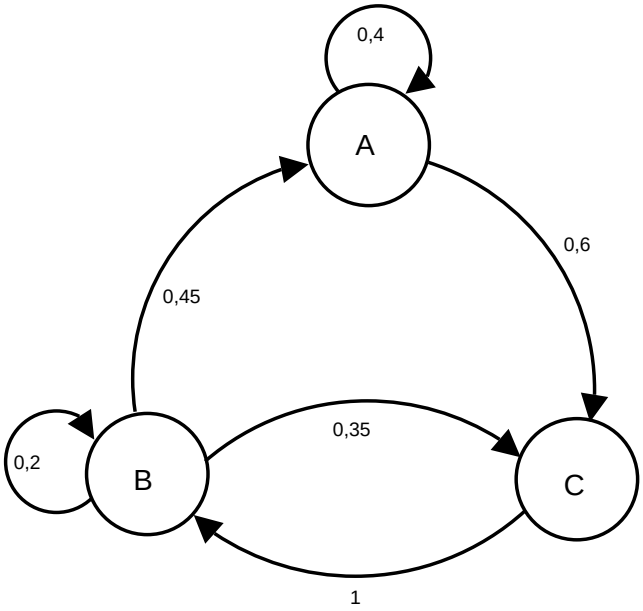


Abbildung 7: Beispiel für eine Markov-Kette

Abbildung 7 zeigt ein Beispiel für die Darstellung einer Markov-Kette, wobei die mit Buchstaben bezeichneten Kreise die Zustände sind und die gerichteten Kanten die Zustandsübergänge mit ihren Wahrscheinlichkeiten. In Markov-Modellen wird Sprache als zufallsgesteuerter Prozess verstanden, wobei aber die Zufälle nicht gleichverteilt sind. Markov-Modelle liefern eine präzise Beschreibung von Sprache und finden in Feldern wie Syntaxüberprüfung oder Spracherkennung ihre Anwendung. Eine genauere und formale Beschreibung von Markov-Modellen findet man bei [Herm02] und [Hotz97].

Kollokationen

Kollokationen beschreiben häufig gemeinsam auftretende Wörter oder Wortformen innerhalb eines bestimmten sprachlichen Kontextes. Dieser Kontext ist relativ frei wählbar, es handelt sich aber meist um direkt benachbarte Wörter oder Sätze bzw. Abschnitte.

Die grundlegende Annahme im Zusammenhang mit Kollokationen ist, dass Wörter, die überdurchschnittlich oft in einem begrenzten Kontext vorkommen, eine gemeinsame Bedeutung haben. Der Wahrheitsgehalt dieser Annahme ist gleichzeitig das Gütekriterium für diese Methode. Das Verfahren ist gut, wenn möglichst viele Ergebnisse im semantischen Umfeld des gegebenen Begriffs liegen. Eine Visualisierung von Satz-Kollokationen ist in Abbildung 8 dargestellt. Eine Schwierigkeit bei Kollokationen besteht ähnlich zu den Konkordanzen in der Frage der Basis: Wortformen oder Stammformen?

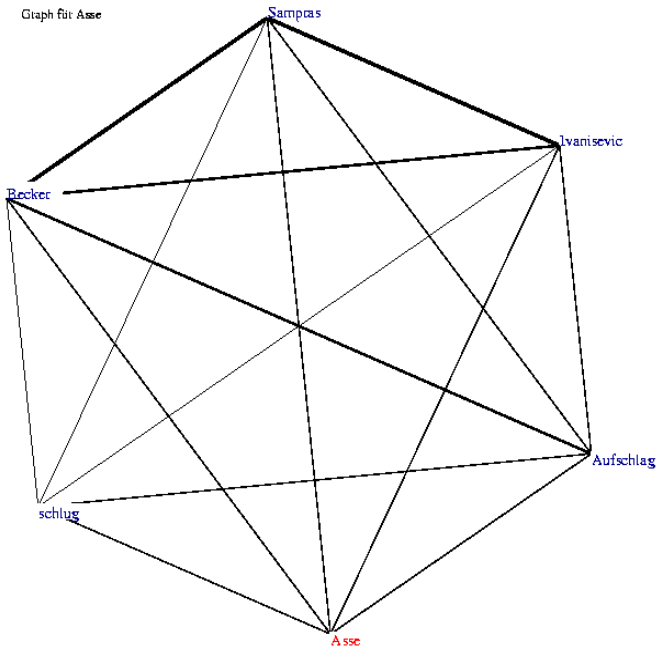


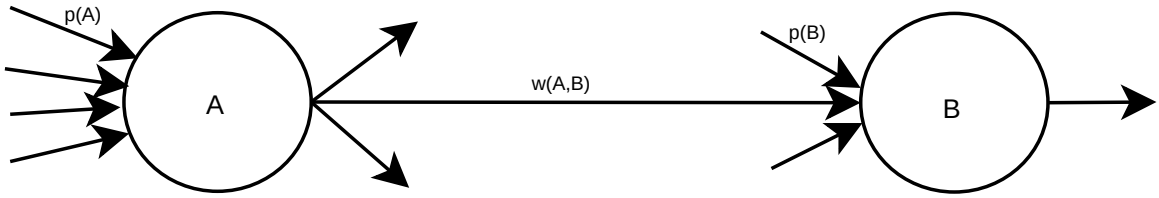
Abbildung 8: Visualisierung der Kollokationen für das Wort Asse [Quas98]

2.2.3 Künstliche neuronale Netze

Der Funktionsweise künstlicher neuronaler Netze liegt die Arbeitsweise des menschlichen Gehirns zugrunde. Künstliche neuronale Netze (KNN) sind eng mit dem Begriff Konnektivismus verbunden, welcher Informationsverarbeitung als Interaktion einer großen Anzahl einfacher Einheiten ansieht [Zell94]. KNNs sind „Netzwerke intensiv verbundener neuronaler Prozessorelemente, deren Fähigkeit darin besteht, auf Eingabereize zu reagieren, zu lernen und sich der Umgebung entsprechend anzupassen“[Patt96]. Die Komponenten eines künstlichen neuronalen Netzes sind Neuronen, welche die Informationen speichern, das Verbindungsnetzwerk der Neuronen, das als ein gerichteter Graph zu verstehen ist, die Propagierungsfunktion, die aus den Ausgaben eines oder mehrerer Neuronen eine Eingabe für ein anderes Neuron errechnet und schließlich ein Lernalgorithmus, nachdem das KNN arbeitet [Zell94]. In Abbildung 9 ist ein KNN in sehr vereinfachter Form dargestellt. Bei A und B handelt es sich um Neuronen, w(A, B) stellt einen Teil des Verbindungsnetzwerkes in Form einer gewichteten Kante dar, also einfacher gesagt eine Verbindung zwischen zwei Zellen (in diesem Fall zwischen A und B). P(A) und p(B) sind die Propagierungsfunktionen, die angeben, wie aus den verschiedenen eingehenden Informationen die Ausgabe des Neurons berechnet wird.

Man unterscheidet bei KNNs zwischen Netztopologien mit bzw. ohne Rückkopplungen. Bei Netzen ohne Rückkopplung gibt es keinen Pfad, der von einem Neuron ausgehend zu

demselben Neuron zurückführt, es handelt sich also um Bäume. In den Topologien mit Rückkopplungen existieren solche Zyklen.



p=Propagierungsfunktion

w(A,B)=Gewicht der Verbindung zwischen Neuronen A und B

Abbildung 9: Darstellung eines Künstlichen Neuronalen Netzes

Lernen wird im Zusammenhang mit KNNs als Prozess verstanden, der die freien Parameter eines neuralen Netzes durch einen fortfahrenden Prozess der Anregung durch die Umgebung anpasst, in dem das Netz eingebettet ist [Hayk94]. Es gibt bei KNNs grundsätzlich drei verschiedene Lernverfahren:

- **Überwachtes Lernen**: Jedes Beispielmuster für das Training beinhaltet Eingabemuster und Ausgabemuster, also auch eine „richtige“ Antwort.
- **Bestärkendes Lernen**: Es wird ebenfalls die Anwesenheit eines Lehrers vorausgesetzt, die korrekte Antwort wird jedoch nicht präsentiert. Stattdessen werden Hinweise gegeben.
- **Nicht-überwachtes Lernen**: Das Netzwerk erhält keine Rückmeldung über Korrektheit der Ausgabe. Es gibt keinen Lehrer.

KNNs finden vor allem im Bereich der Mustererkennung ihre Anwendung.

2.2.4 Vektorbasierte Methoden

In diesem Abschnitt werden zwei Verfahren, die auf der Basis von Begriffsvektoren arbeiten, beschrieben: Das Vector Space Model und Latent Semantic Indexing.

Vector Space Model

Das Vector Space Model (VSM) befasst sich hauptsächlich mit drei Problemstellungen. Die erste ist die Repräsentation von Dokumenten, also wie ein Dokument dargestellt werden kann, um in angemessener Weise mit anderen Dokumenten vergleichbar zu sein. Die zweite Problemstellung des VSM ist die Darstellung von Anfragen, so dass sie mit den Dokumenten vergleichbar sind. Die dritte Aufgabe ist das Finden von Ergebnissen auf eine Anfrage.

Als Basis für den Vergleich von Dokumenten oder Anfragen dient ein mehrdimensionaler Vektor, wobei die Anzahl der Dimensionen nicht festgelegt ist. Die Umwandlung eines

Textes in einen Vektor muss von anderen textanalytischen Verfahren im Voraus geleistet werden. Das Ähnlichkeitsmaß zwischen zwei verschiedenen Dokumenten wird in diesem Modell als geometrisches Problem angesehen. Mehrere Dokumente können auf Basis ihrer Vektoren in einer Dokumentenlandkarte (siehe Abbildung 10) dargestellt werden. Dokumentenlandkarten veranschaulichen inhaltliche Ähnlichkeiten von Dokumenten durch eine Visualisierungsstruktur, die an topologische Karten erinnert und in welcher inhaltlich ähnliche Dokumente nahe zusammen liegen.

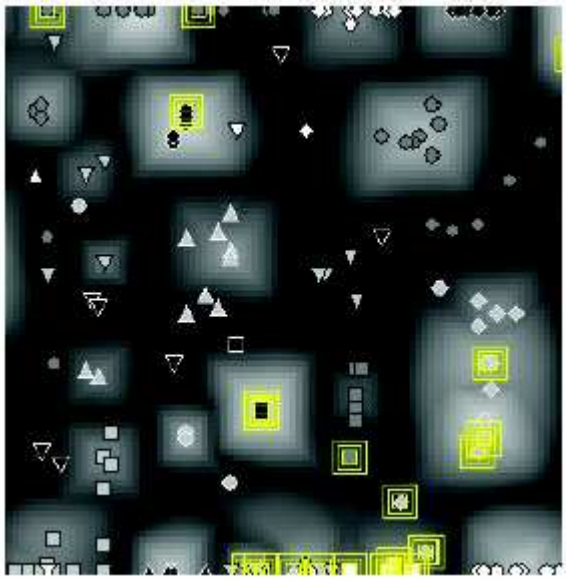


Abbildung 10: Beispiel für eine Dokumentenlandkarte [JaBe03]

Interessant ist nun die Fragestellung, wie sich die Ähnlichkeit zweier Dokumente in der Dokumentenlandkarte ersehen oder errechnen lässt. Relativ nahe liegend erscheint die aus der linearen Algebra stammende euklidische Distanz zwischen zwei Punkten [Sull01]. Ein anderer Ansatz stammt aus der Trigonometrie und versteht die Dokumente nicht mehr nur als Punkte im Raum, sondern als Ursprungsvektoren(siehe Abbildung 11). Bei diesem Vorgehen wird das Ähnlichkeitsmaß durch den Winkel zwischen zwei Dokumentenvektoren bestimmt. An seine Grenzen stößt das VSM bei einer sehr großen Zahl von Dokumenten sowie bei häufigem Auftreten von Synonymen oder Worten mit Mehrfachbedeutungen.

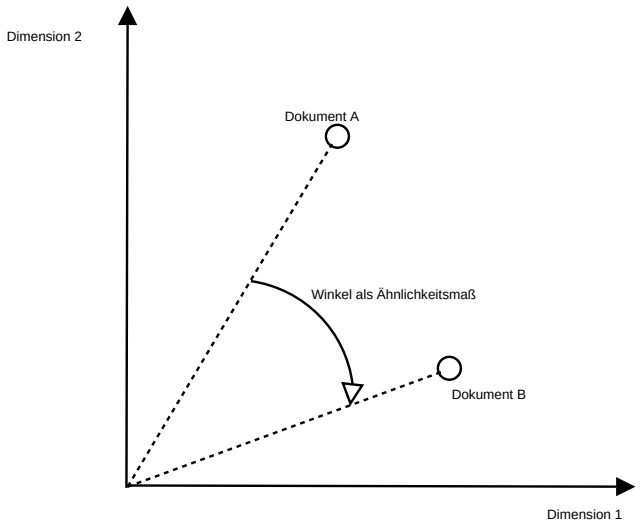


Abbildung 11: Visualisierung des Vector Space Models

Latent Semantic Indexing

Das VSM ist mit dem Problem behaftet, mit Polysemie und Synonymie nicht umgehen zu können. Unter Polysemie versteht man dabei Wörter mit mehrfacher verschiedener Bedeutung, welche als Folge die Auswahl von irrelevanten Dokumenten mit sich bringen kann. Synonymie bezeichnet genau das Gegenteil, nämlich die Bezeichnung eines Objekts durch verschiedene Begrifflichkeiten. Die Gefahr hierbei ist, dass relevante Dokumente nicht berücksichtigt werden.

In den meisten Fällen sind die Personen, welche die Indexierung vornehmen nicht dieselben, die Abfragen ausführen. Dies kann dazu führen, dass verschiedene Begriffssysteme aufeinanderprallen. Ein weiteres Problem kann dadurch entstehen, dass nicht alle im Dokument verwendeten relevanten Terme im Vektor berücksichtigt werden, da sonst die Matrix, welche die Basis für die folgenden Berechnungen darstellt, zu groß und damit zu unhandlich würde. Diese Matrix wird als Term-Dokument-Matrix bezeichnet und setzt sich aus den verschiedenen Dokumenten, von denen jedes in einer Spalte der Matrix repräsentiert ist, sowie aus den in den Dokumenten enthaltenen Termen, welche die Zeilenköpfe bilden, zusammen.

An dieser Stelle setzt das Latent Semantic Indexing (LSI) an [Deer90, Sull01]. Durch ein Verfahren aus der linearen Algebra, der Singular Value Decomosition (SVD), ist es möglich, eine Term-Dokument-Matrix, wie sie sich bei der Betrachtung mehrerer VSM-Vektoren aufspannen würde, in eine Matrix mit unabhängigen, künstlichen Dimensionen zu übersetzen, welche den Vorteil hat, dass sie in ihrer Dimension verkleinert werden kann.

Vorzüge dieser Methode sind der wesentlich bessere Umgang mit Synonymen und das Finden von Dokumenten mit semantischer Verwandtschaft, die aus dem Suchvektor nicht ersichtlich ist. Die Funktionsweise von LSI ist Gegenstand des Kapitels 3.2.2 und soll deshalb an dieser Stelle nicht weiter vertieft werden.

2.2.5 Heuristiken

Neben den bisher aufgeführten Verfahren gibt es noch eine Vielzahl von weiteren Methoden, die einen weniger wissenschaftlichen Hintergrund haben. Diese werden als Heuristiken bezeichnet. Mit einer Heuristik wird grundsätzlich eine Lehre zur methodischen Gewinnung neuer Erkenntnisse mit Hilfe der Erfahrung bezeichnet. Heuristiken beruhen in der Künstlichen Intelligenz meist auf Faustregeln bzw. Algorithmen. Sie liefern in der Informatik häufig keine optimalen, aber meist gute Lösungen [Wiki05].
An dieser Stelle sollen beispielhaft zwei Verfahren aus der Vielzahl von Heuristiken angesprochen werden, die im Kontext dieser Arbeit interessant erscheinen, das Auswerten von Strukturinformationen sowie die gezielte Inhaltsanalyse.

Ontologien

Der Begriff Ontologie stammt ursprünglich aus der Philosophie und bezeichnet eine Disziplin, die sich primär mit dem Sein, dem Seienden als solchem und mit den fundamentalen Typen von Entitäten beschäftigt. In der Informatik versteht man unter einer Ontologie nach [Grub93] „die explizite formale Spezifikation einer gemeinsamen Konzeptualisierung“. Durch Ontologien werden einzelne Wissensbereiche durch eine standardisierte Terminologie, Beziehungen innerhalb dieser Terminologie sowie gegebenenfalls zugehörige Ableitungsregeln vereinheitlicht. Zweck von Ontologien ist es, den Austausch und das Teilen von Wissen zu erleichtern und die Verbesserung der Kommunikation zwischen menschlichen und maschinellen Akteuren [Staa02].
In der jüngeren Vergangenheit haben sich Ontologien vor allem für das Semantic Web als interessant erwiesen [Bern01]. Die Idee des Semantic Web ist es, Web-Dokumente mit Hilfe von Metadaten eine Semantik zu verleihen, um in effizienterer Weise inhaltlich im Internet nach Dokumenten suchen zu können. Ontologien haben in diesem Zusammenhang die Aufgabe, die Metadaten sowie eventuell Verknüpfungsregeln für die Metadaten zu liefern.

Auswertung von Strukturinformationen

Viele auf den ersten Blick unstrukturierte Texte tragen Strukturinformationen oder Metadaten in sich, die zunächst auf irgendeine Weise extrahiert werden müssen, um dann im

zweiten Schritt inhaltlich ausgewertet zu werden. Beispielsweise sind in einem MS Word oder Rich Text Dokument verschiedene Textpassagen unterschiedlich formatiert. Wenn ein Satz im Text als Überschrift formatiert ist, so kann man davon ausgehen, dass dieser Satz aussagekräftiger bezüglich des Inhalts des gesamten Textes ist, als ein beliebiger Satz im Standardformat. Ein anderes Beispiel sind E-Mails, bei denen die Betreffzeile in der Regel einen extrem hohen Informationsgehalt besitzt.

2.2.6 Bewertung der Methoden

In diesem Abschnitt sollen die aufgeführten Verfahren zur automatischen Textanalyse auf ihre Stärken und mögliche Schwierigkeiten hin untersucht werden.

Linguistische Verfahren

Die Verfahren der Linguistik haben ihre Vorzüge ganz klar in ihrer Nutzbarkeit auch bei relativ kleinen Textmengen, da andere Verfahren erst bei großen Textmengen zuverlässige Ergebnisse liefern können. Generell ist zu den linguistischen Verfahren zu sagen, dass ihre Qualität sehr stark von der zugrunde liegenden Datenbasis, wie zum Beispiel der Morphem-Liste, abhängig ist. Ist diese Basis schlecht gewählt, so sind auch die zu erwartenden Ergebnisse von minderer Qualität. Die Datenbasis bringt noch eine weitere Schwierigkeit mit sich, nämlich den zusätzlichen Aufwand, der bei der Erstellung von Referenz-Daten entsteht. Gegebenenfalls müssen für jedes Fachgebiet neue Referenzdaten erstellt werden, da beispielsweise Morphem-Listen einzelner Fachgebiete sehr stark variieren können und eine einzige Datenbasis wie schon erwähnt zu einer qualitativen Verschlechterung der Daten führen kann. Es bleibt also festzuhalten, dass linguistische Methoden, wenn sie nicht auf eine fachliche Domäne beschränkt sind, große Probleme bei der Realisierung mit sich bringen können. Bei der syntaktischen Analyse ist die Problematik anders gelagert. Hier ist die Fachrichtung prinzipiell beliebig, das Entscheidende ist die Strukturiertheit der Texte. Quellcodes oder ähnlich stark strukturierte Texte sind mit dieser Methode sehr gut zu bearbeiten. Wenn aber ein Text nicht so wohlgeformt ist oder er eher umgangssprachlich formuliert wurde, so ist die Verwendung der syntaktischen Analyse sehr problematisch.

Statistische Verfahren

Der Vorteil der statistischen Verfahren liegt in ihrer relativ problemlosen Umsetzbarkeit, da man prinzipiell nur die Häufigkeiten der verschiedenen Wortformen ermittelt. Die Qualität der Ergebnisse hängt sehr stark von den allgemeinsprachlichen Referenzkorpora[6]

[6] Sofern man diese verwendet und nicht nur die reine Häufigkeit betrachtet

und den damit verbundenen Schwellenwerten ab, wann ein Ergebnis als Fachbegriff bezeichnet wird. Weiterhin können die Referenztexte auch sehr domänenspezifisch sein, so dass sie je nach Fachrichtung unter Umständen ausgetauscht werden müssen. Die geeignete Auswahl der Referenzen setzt zudem ein hohes Domänenwissen voraus. Bei Konkordanzen und Kollokationen stellt sich noch die Frage nach der Datenbasis. Werden die Wörter einfach so behandelt, wie sie im Text vorkommen oder wird die Stammform gebildet. Der zweite Fall bedeutet einen zusätzlichen Aufwand, da die Stammformen der Wörter zunächst gebildet werden müssen. Zu den Markov-Modellen lässt sich sagen, dass sie eine sehr präzise Beschreibung von Sprache liefern, bei zunehmender Größe aber sehr komplex und kaum mehr nachzuvollziehbar werden. Ein Problem, das alle statistischen Methoden haben ist, dass sie erst ab einer bestimmten Textmenge, die in der Regel sehr groß ist, zufrieden stellende Ergebnisse liefern.

Vektorbasierte Verfahren

Die auf Vektoren basierenden Methoden schaffen eine gute Möglichkeit, verschiedene Texte vergleichbar zu machen und Dokumentenlandkarten zu erstellen. Das Vector Space Model hat jedoch in zwei Bereichen seine Schwierigkeiten. Die erste Schwierigkeit besteht in der schlechten Performance dieses Verfahrens bei großen Mengen von Daten. Das zweite Problem ist der Umgang mit Synonymen und Polysemen, die das Endergebnis verfälschen und dadurch die Qualität des Verfahrens mindern können. So können inhaltlich verwandte Wörter bei Anfragen nicht berücksichtigt werden. Latent Semantic Indexing ist als Reaktion auf diese Problematiken entstanden und kann den problematischen Umgang des Vector Space Models mit Synonymen weitgehend neutralisieren, während Polyseme allerdings problematisch bleiben.

Künstliche Neuronale Netze

KNNs zeichnen sich vor allem durch ihre hohe Geschwindigkeit aus, die aus der massiven Parallelität dieser Netze resultiert. Außerdem laufen die Systeme relativ robust, da der Ausfall eines einzelnen Neurons nicht so sehr ins Gewicht fällt [Patt97]. Negativ anzumerken ist, dass KNNs erst mit großem Aufwand „trainiert“ werden müssen, bevor sie ihre volle Leistungsfähigkeit erreichen. KNNs scheinen insbesondere zur paarweisen Mustererkennung geeignet, nicht aber zur Analyse von Textdokumenten.

Heuristiken

Das Auswerten von Strukturinformationen kann sicherlich eine große Hilfe bei der Erschließung von Texten sein. Eine Bewertung ist jedoch schwierig, da dieses Verfahren nicht ausreichend standardisiert ist und man im Einzelfall über die Eignung der Verfahren

urteilen müsste. Ontologien sind für die Kommunikation innerhalb eines Fachbereichs ein gutes Werkzeug, allerdings hängt der Wert der Ontologien in hohem Maße von der Akzeptanz innerhalb des jeweiligen Fachgebietes ab. Sobald die Thematik aber über ein Fachgebiet hinausgeht, ist der Umgang mit Ontologien problematisch, da man in diesem Fall für jede mögliche Fachrichtung eine eigene Ontologie anbieten müsste.

Fazit

Für die Problemstellung der vorliegenden Diplomarbeit eignet sich eine Kombination verschiedener, in diesem Kapitel vorgestellter, Textanalyse-Verfahren, um Dokumente automatisch zu analysieren und die so erzeugten Benutzerprofile zu matchen. Da das in dieser Arbeit vorgestellte Verfahren erwartungsgemäß mit größeren Textmengen arbeitet, bietet sich ein statistisches Verfahren zur Analyse der Texte an. Weiterhin empfiehlt sich Latent Semantic Indexing für die das Matching der schlüsseltermbasierten Benutzerprofile. Diese beiden Bereiche der automatischen Textanalyse bilden den Kern entwickelten Algorithmus zum Matching von Benutzern auf Basis ihrer Textproduktion.

3 Konzept für ein Benutzer-Matching mittels Dokumentenanlyse

In diesem Kapitel wird ein Konzept vorgestellt, das auf der Basis von Dokumenten eines Benutzers ein Profil erstellt und die Funktionalität bietet, verschiedene Benutzerprofile auf ihre Ähnlichkeit hin zu vergleichen. Zunächst wird unter der Annahme, dass die Dokumente eines Benutzers Rückschlüsse über seine Qualifikationen und Interessen zulassen, aus den geschriebenen Texten des Benutzers ein Profil erstellt. Das Konzept zielt darauf ab, jedem Benutzer die Möglichkeit zu geben, sich Empfehlungen über zu seinem eigenen Profil oder zu einer von ihm gestellten Anfrage ähnlichen Benutzer geben zu lassen.

Abschnitt 3.1 erläutert die Problemstellung, welche die Motivation dieser Arbeit darstellt sowie die Zielsetzung für das Konzept. In Abschnitt 3.2 wird die Konzeption für ein Verfahren entwickelt, mit dem zunächst Texte eines Benutzers analysiert werden und danach ein Matching mit anderen Benutzern oder mit Anfragen durchgeführt werden kann.

3.1 Motivation und Zielsetzung des Konzepts

In den letzten Jahren nimmt die Ressource Wissen in unserer Gesellschaft einen immer größeren Stellenwert ein. Das Wissensmanagement übernimmt die immer mehr in den Fokus des allgemeinen Interesses gerückte Aufgabe, Wissen zu identifizieren und verfügbar zu machen. Diese Aufgabe ist nicht zuletzt mit der steigenden Bedeutung des Internets als vielseitige Bibliothek im Arbeitsprozess zunehmend wichtiger geworden, da dort kaum zu bewältigende Informationsmengen zur Verfügung stehen. Dem Wissensmanagement kommt in diesem Zusammenhang die Aufgabe zu, die Informationen herauszufiltern, die für das jeweilige Problem von Interesse sind.

Die Filterung sinnvoller Informations- oder Wissensträgern wird unter Anderem von Recommender Systemen geleistet. Um dem Benutzer sinnvolle Empfehlungen geben zu können, braucht ein Recommender System möglichst viele Informationen über mögliche Alternativen. Im Fall des ExpertFinder Frameworks heißt das, möglichst viel über die in der entsprechenden Organisation befindlichen Benutzer und ihre fachlichen Kompetenzen und Interessen zu erfahren. Diese Informationen werden bisher durch Analyse von Benutzerhistorien sowie Hinzuziehung der Ausbildungssituation der Benutzer beschafft. Für eine vollständige Berücksichtigung aller Informationsquellen ist es sinnvoll, eine weitere Option zu nutzen: Die Dokumente jedes Benutzers. Ausgehend von der Annahme, dass

sich in den Texten eines Benutzers seine Fähigkeiten bzw. Interessen widerspiegeln, sollten solche Dokumente sehr gezielt Aufschluss über die Qualifikationen eines Benutzers geben können und so ein sehr guter Indikator für qualifizierte Empfehlungen sein. Diese Arbeit versucht die beiden in Kapitel 2 beschriebenen wissenschaftlichen Disziplinen, Recommender Systeme und automatische Textanalyse zu integrieren, um so das explizite Wissen von Benutzern transparent zu machen. Durch den mittels der Empfehlung entstehenden persönlichen Kontakt des Benutzers mit dem Experten kann daraufhin unter Umständen sogar das implizite Wissen des Experten kommuniziert werden.

Die Zielsetzung dieser Diplomarbeit ist, ein Verfahren zu entwickeln, das die Erstellung von Empfehlungen auf Basis von Textdokumenten ermöglicht. Das Verfahren soll zunächst die Dokumente einer Person auf ihren Inhalt hin analysieren. Die aus der Analyse der Texte gewonnenen Informationen sollen in einer Weise verdichtet werden, so dass das erstellte Format möglichst kompakt ist, damit ein Vergleich mehrerer Personen einfach durchgeführt werden kann. Gleichzeitig sollen möglichst wenig aussagekräftige Informationen verloren gehen. Das Ergebnis aus diesem Arbeitsschritt ist ein Format, das man als „Interessen- und Kenntnis-Profil“ des Benutzers bezeichnen könnte. Eine weitere Eigenschaft dieser Benutzerprofile soll die Möglichkeit eines Matchings solcher Profile sein. Matching bezeichnet in diesem Zusammenhang die Berechnung eines Ähnlichkeitswertes zweier Benutzerprofile bezüglich ihres Inhalts. Es gibt zwei Arten des Matchings, die in diesem Kontext sinnvoll erscheinen:

- **Matchen von verschiedenen Benutzerprofilen**

  Bei dieser Variante des Matchings werden verschiedene Benutzerprofile auf ihre Ähnlichkeit hin untersucht. Diese Methode findet ihre Anwendung, wenn eine Person andere Personen identifizieren will, die grundsätzlich ähnliche fachliche Qualifikationen besitzen. In diesem Fall ist kein konkretes Problem Anlass für die Anfrage sondern die Suche nach Experten aus dem eigenen fachlichen Umfeld.

- **Matchen von Benutzerprofilen auf eine Anfrage**

  Diese Methode bietet die Möglichkeit, die Benutzerprofile mit einer Anfrage zu matchen. Sie findet ihre Anwendung, wenn eine Person ein konkretes Problem hat und zu diesem Anliegen einen Experten sucht.

Auf den Ergebnissen dieser beiden kurz skizzierten Matching-Varianten basierend sollen anhand der ermittelten Ähnlichkeiten die Benutzer in eine Rangfolge gebracht werden. Diese Ähnlichkeiten können entweder in die Ergebnisse der anderen Matching-Module

des ExpertFinder Frameworks integriert werden oder es kann direkt eine Empfehlung generiert werden, die allein auf den Daten der Textanalyse basiert.
Weiterhin soll eine Evaluation des konzipierten Verfahrens stattfinden, um einerseits Aufschlüsse über seine Qualität zu erlangen und andererseits eine sinnvolle Konfiguration des Verfahrens zu erreichen. Hierbei ist insbesondere die Wahl der LSI-Dimensionen von Interesse, da diese ein bekanntes Problem darstellt, zu dessen Lösung die Evaluation beitragen soll.

3.2 Konzeption eines Prototyps

In diesem Abschnitt wird ein prototypisches Verfahren zur automatischen Textanalyse konzipiert. Das Verfahren soll die schon bestehenden Matching-Module des ExpertFinder Frameworks [Reic04] ergänzen.
Das Ziel des in diesem Kapitel beschriebenen Algorithmus ist es, anhand einer Auswahl von Dokumenten des Benutzers, Schlüsse über seine Kompetenzen zu ziehen. Das Verfahren wird benutzt, um für das Recommender System „ExpertFinder“ Informationsgrundlagen in Form von Benutzerprofilen zu schaffen und auf Basis dieser Benutzerprofile geeignete Experten für ein bestimmtes Problem zu ermitteln.
Das Verfahren besteht grundsätzlich aus zwei Teilbereichen: Der erste Bereich umfasst das Erstellen des Benutzerprofils aus einem unformatierten Text[7], welches die Datenbasis für die weiteren Berechnungen bieten soll. Das Benutzerprofil soll in Form einer Liste von Begriffen dargestellt werden, die bezüglich der Kompetenzen des Autors aussagekräftig ist. Im zweiten Teilbereich geht es um die Durchführung von Matching-Operation zur Findung geeigneter Experten zu einem Problem. Unter Matching versteht man in diesem Kontext den Vergleich verschiedener Benutzerprofile bzw. Anfragen und Benutzerprofile auf ihre Ähnlichkeit. Das Ergebnis des Matchings ist eine Liste von Empfehlungen in Form von Benutzern. Die einzelnen Arbeitsschritte des Verfahrens und ihr Zusammenhang sind in Abbildung 12 dargestellt.

[7] Die Filterung von unformatiertem Text aus den gängigen Text-Formaten wie doc, pdf, ppt, usw. war Thema einer vorgelagerten Projektarbeit und wird an dieser Stelle vorausgesetzt.

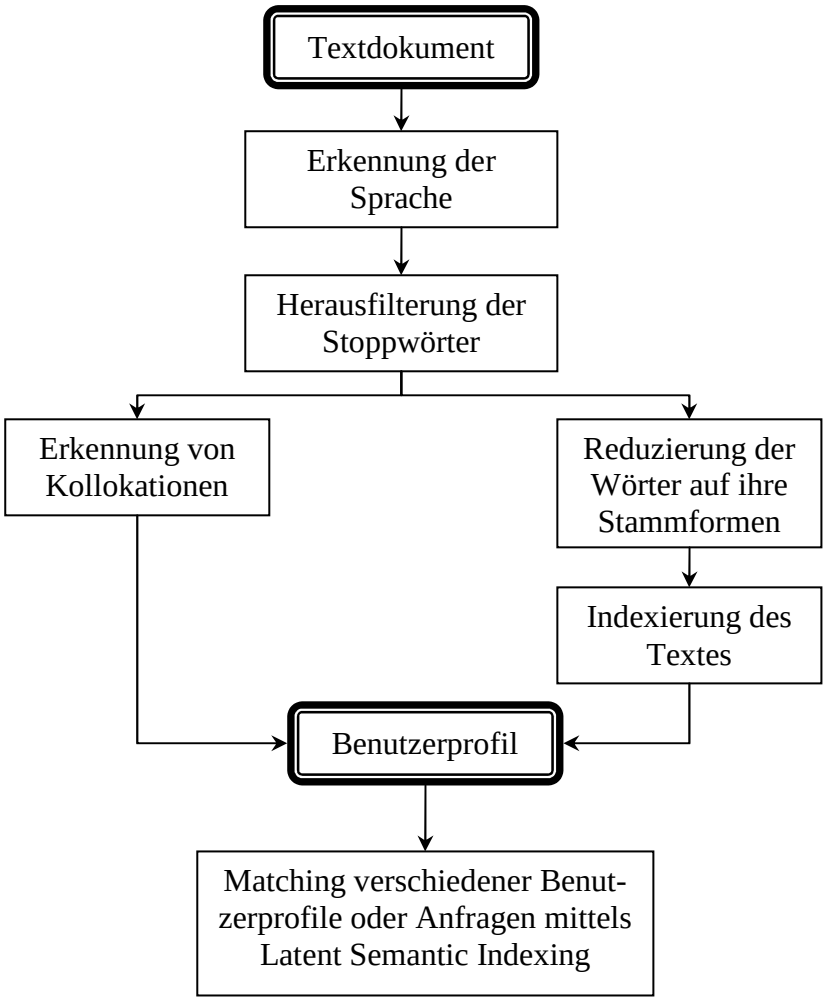


Abbildung 12: Konzept für einen Algorithmus zur automatischen Textanalyse

Die ersten beiden Schritte des Verfahrens sind eine Spracherkennung und eine anschließende Stoppwortfilterung. Danach teilt es sich in zwei Verfahrensstränge auf. Auf der einen Seite steht die Reduktion der Wörter auf ihre Stammformen sowie anschließende Indexierung des Textes mit Bestimmung der Worthäufigkeiten, auf der anderen Seite das Finden von Kollokationen. Das Benutzerprofil setzt sich aus den Ergebnissen dieser beiden Verfahrensstränge zusammen. Es bildet die Datenbasis für die Bestimmung der Ähnlichkeit zwischen verschiedenen Benutzern sowie die Berechnung von Empfehlungen auf Benutzeranfragen. Weiterhin können auf diese Weise Empfehlungen zu bestimmten Anfragen erzeugt werden. Dieses Benutzer- bzw. Anfrage-Matching wird mittels LSI durchgeführt. In Abschnitt 3.2.1 wird die Erstellung des Benutzerprofils genauer beschrieben. Der darauf folgende Matching-Vorgang wird in Abschnitt 3.2.2 erläutert.

3.2.1 Erstellung eines Benutzerprofils

Dieser Abschnitt beschreibt den ersten Teilbereich des erarbeiteten Verfahrens der automatischen Textanalyse, die Erstellung eines Benutzerprofils, welches aus einer Reihe von bedeutungstragenden Begriffen bestehen soll. Dieser Teilbereich soll durch ein statistisches Verfahren realisiert werden, das durch integrierte linguistische Verfahrensschritte noch präzisere Ergebnisse liefert. Bei der Generierung dieser Beschreibung der Kompetenzen eines Benutzers wirken mehrere Einzelschritte zusammen, die in diesem Kapitel genauer beschrieben werden. Der erste Abschnitt befasst sich mit der Identifizierung der hauptsächlich im Text vorkommenden Sprache. Daran schließt sich die Stoppwortfilterung an, die den Text von Elementen bereinigt, die keine Informationen beinhalten. Im dritten Abschnitt wird die darauf folgende Reduktion der einzelnen Wörter auf ihre Stammformen beschrieben, woran sich die Indexierung des Textes unter Berücksichtigung der Worthäufigkeiten anschließt. Zusätzlich zu den Ergebnissen aus diesem Teil des Verfahrens fließen in das Ergebnis Kollokationen, die im fünften Abschnitt erläutert werden, mit ein. Schließlich wird im letzten Abschnitt auf die Zusammensetzung des Benutzerprofils eingegangen.

Erkennung der Textsprache

Der erste Schritt des Verfahrens besteht in der Identifizierung der Sprache des Textes. Es ist natürlich möglich, dass im Text mehrere Sprachen enthalten sind. In diesem Fall wird die am häufigsten auftretende Sprache sowie die Anteile aller identifizierten Sprachen im Text bestimmt. Das Herausfinden der Sprache des Textes ist notwendig, da die später durchgeführte Reduktion der Wörter auf ihre Stammformen von der Sprache abhängig ist. Die Textsprache wird durch gezielte Suche nach bestimmten, in einer Sprache häufig auftretenden, Wörtern bestimmt. Wenn nun die identifizierten Wörter alle zur gleichen Sprache gehören, kann man davon ausgehen, dass der gesamte Text in dieser Sprache verfasst ist.

Herausfiltern der Stoppwörter

Unter Stoppwörtern versteht man Wörter, die sehr häufig in Texten enthalten sind, aber keine Relevanz bezüglich des Inhalts haben. Diese Stoppwörter müssen vor der weiteren Verarbeitung des Textes entfernt werden, da sie das Gesamtergebnis in negativer Weise beeinflussen würden. Dieser Einfluss resultiert aus der Tatsache, dass Stoppwörter relativ häufig in Texten auftreten, also bei einer Schlüsselwortanalyse weit vorn im Ergebnis auftauchen würden, aber keine Informationen über den Text beinhalten. Stoppwörter stammen aus verschiedenen Wortarten, die in Abbildung 13 zusammengestellt sind.

Zusätzlich zum Herausfiltern von nicht informationsträchtigen Wörtern wird durch die Stoppwortfilterung die Gesamtzahl an Wörtern reduziert, was im weiteren Verlauf eine bessere Performance zur Folge hat.

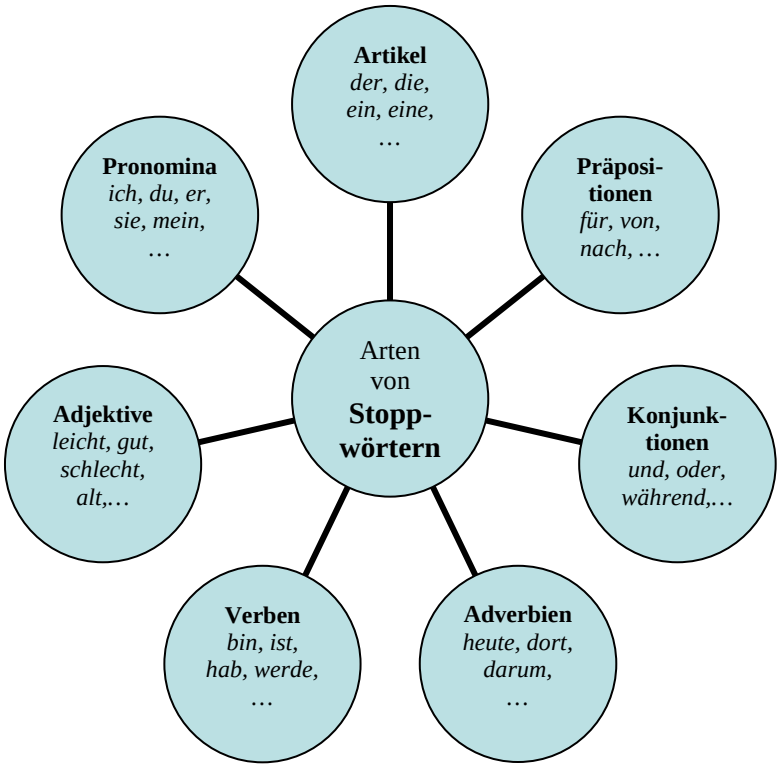


Abbildung 13: Arten von Stoppwörtern

Reduktion der Wörter auf Stammformen

Der Zweck der Reduzierung aller Wörter auf ihre Wortstämme liegt in der Identifizierung verwandter Wörter oder Wörter einer Wortfamilie und ihrer gemeinsamen Gewichtung bei der späteren Suche nach bedeutungsträchtigen Stichworten. Beispielsweise kann der gleiche Begriff in Kasi im Text auftreten, beispielsweise einmal als Nominativ und einmal Genitiv, wie System und Systems. Der Begriff soll in diesem Fall natürlich nur einmal als Schlüsselwort auftauchen, gleichzeitig sollen aber zur Gewichtung beide Formen herangezogen werden. In Abbildung 14 ist ein solches Szenario dargestellt. Auf der linken Seite steht die Liste mit allen im Text auftretenden Wörtern. Die in Klammern stehende Zahl hinter jedem Wort gibt die absolute Häufigkeit des Vorkommens des jeweiligen Wortes an. Die vier Worte in der linken Liste stammen aus einer Wortfamilie, werden jedoch nicht als solche erkannt. Durch die Reduktion der Wörter auf ihre Stammformen wird erreicht, dass die vier Wörter als zu einer Wortfamilie gehörig identifiziert werden und gemeinsam zu einem Schlüsselwort aggregiert und entsprechend mit der Summe aller Auftrittshäufigkeiten gewichtet werden.

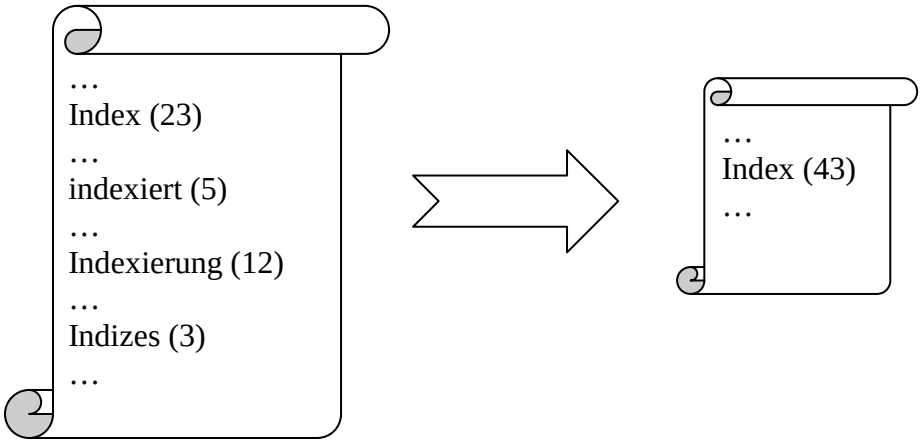


Abbildung 14: Ziel der Reduktion von Wörtern auf ihre Stammform

Es gibt verschiedene Verfahren um eine Wortstammreduktion durchzuführen:

- Ein sehr einfaches Verfahren, bei dem von jedem Wort eine festgelegte Anzahl von Buchstaben abgeschnitten wird heißt *Trunkierung*. Problematisch an diesem Verfahren ist die Tatsache, dass es ebenso fehleranfällig wie einfach ist. Man kann sich sehr leicht am obigen Beispiel klarmachen, dass Trunkierung für die deutsche Sprache nicht zu zufrieden stellenden Ergebnissen führt. Wenn man vom Wort Indexierung ausgeht, müsste man sechs Buchstaben „abschneiden“, um zum gemeinsamen Wortstamm zu kommen. Diese Vorgehensweise würde bei den anderen Wörtern zu absolut unsinnigen Ergebnissen führen. Trunkierung ist also im Kontext dieser Arbeit nicht zu gebrauchen.
- *Stemming* ist ein Verfahren, das auf Basis eines Algorithmus automatisch jedes Wort auf seinen Wortstamm reduziert. Diese Reduktion passiert jedoch nicht auf Basis eines Wörterbuches, sondern anhand einer Reihe von Regeln. Der Stemming-Algorithmus von Porter [Port80], der ursprünglich für die englische Sprache entwickelt wurde, wurde von [Snow04a] so modifiziert, dass er auf die deutsche Sprache anwendbar ist.

  In Abbildung 15 ist die deutsche Version des Porter-Stemming-Algorithmus dargestellt. Grundsätzlich besteht das Verfahren aus fünf Schritten: einem Vorbereitungsschritt, drei Schritten, die die eigentliche Wortstammreduktion ausführen, sowie einem Nachbereitungsschritt. Da die deutsche Sprache aufgrund ihrer relativ unregelmäßigen Morphologie für das Stemming nicht optimal geeignet ist, muss der deutsche Stemmer unter Umständen etwas angepasst werden.

Porter Stemming Algorithmus für die deutsche Sprache

Vorbereitung:

- ersetze u und y in Großbuchstaben, wenn sie zwischen Vokalen stehen
- ersetze ß durch ss
- ersetze ae durch ä
- ersetze oe durch ö
- ersetze ue durch ü, wenn kein q vorausgeht

Schritt 1:

- Suche das längste der folgenden Suffixe
  - e em en ern er es
  - s, wenn b, d ,f ,g ,h ,k ,l ,m ,n ,r oder t vorausgeht

  und lösche, wenn in R1

Schritt 2:

- Suche das längste der folgenden Suffixe
  - en er est
  - st, wenn b, d, f, g, h, k, l, m, n oder t und mindestens drei Buchstaben vorausgehen

  und lösche, wenn in R1

Schritt 3:

- Suche das längste der folgenden Suffixe und führe die entsprechende Anweisung aus
  - end ung — lösche, wenn in R2 und ig oder e vorausgeht
  - ig ik isch — lösche, wenn in R2 und kein e vorausgeht
  - lich heit — lösche, wenn in R2 und er oder en voausgeht oder wenn in R1
  - keit — lösche, wenn in R2 und lich oder ig vorausgeht oder wenn in R1

Nachbereitung:

- ersetze U und Y wieder durch Kleinbuchstaben

Definition von R1 und R2

R1: R1 ist die Region nach dem ersten Konsonanten, der auf einen Vokal folgt im Wort oder null, wenn diese Region nicht existiert.
Beispiel: W i r t s c h a f t s i n f o r m a t i k
|< R1 >|

R2: R2 ist die Region nach dem ersten Konsonanten, der auf einen Vokal folgt in R1.
Beispiel: S c h a u f e n s t e r
|< R1 >|
|<R2>|

Abbildung 15: Porter Stemming Algorithmus für die deutsche Sprache [Snow04a]

Der Vorbereitungsschritt hat die Aufgabe, den Text von Uneinheitlichkeiten zu bereinigen. Für die deutsche Sprache liegt diese Aufgabe vor allem im Bereich der Umlaute sowie dem Umgang mit “ß“.

Dieser Textvereinheitlichung folgen die drei eigentlichen Stemming-Schritte. Hier werden anhand bestimmter Regeln die Suffixe, also die Endungen der einzelnen Wörter, reduziert. Für die Reduktion ist auf der einen Seite die Wortendung ausschlaggebend und auf der anderen Seite der Bereich, der dieser Wortendung vorausgeht. Zum Zweck dieser zweiten Prämisse wird jedes Wort in verschiedene Bereiche unterteilt, die R1 und R2 genannt werden. Abhängig von diesen beiden Werten wird überhaupt nur eine Reduktion des Wortes durchgeführt. Diese Vorgehensweise hat den Sinn, Suffixe die zwar von der Wortendung her reduziert werden müssten, aber nicht dem Regelfall entsprechen, zu erkennen und in ihrer ursprünglichen Form zu belassen. Beispielsweise wird in Schritt drei die Endung „isch“ reduziert. Das Wort „italienisch“ würde durch diesen Schritt zu Wortstamm „italien“ reduziert, was durchaus nachvollziehbar ist, das Wort „Tisch“ würde jedoch zu „T“, was keinen Sinn machen würde.

Der letzte Schritt des deutschen Porter-Stemmers hat einzig die Funktion, die im Vorbereitungsschritt vorgenommenen Änderungen teilweise wieder rückgängig zu machen.

- Verfahren die auf Lexika basieren liefern wohl die besten Ergebnisse bei der Reduktion von Wörtern auf ihre Wortstämme. Diese *lexikonbasierten Verfahren* zeichnen sich durch ein umfangreiches Wörterbuch aus, anhand dessen die einzelnen Wörter einem Wortstamm zugeordnet werden. Diese Verfahren liefern recht zuverlässige Ergebnisse, haben jedoch das Problem, dass ein immenser Aufwand zur Erstellung eines Wörterbuches nötig ist und dass sie, hinsichtlich der Performance, wenig effizient sind.

Im Rahmen dieses Konzepts wird zur Reduktion der Wortstämme das Stemming verwendet. Es ist performanter als wörterbuchbasierte Verfahren und es entsteht kein zusätzlicher Aufwand bei der Erstellung bzw. Beschaffung von Wörterbüchern. Da Stemming ein regelbasiertes Verfahren ist, kann es nicht mit Ausnahmeregelungen einer Sprache umgehen. Stemmer haben also eine gewisse Fehlerquote, die aber für den hier verfolgten Zweck in einem akzeptablen Bereich liegt.

Identifizierung der Schlüsselwörter

Auf Basis der Ergebnisse der beiden vorangegangenen Schritte soll eine Reihe von Wörtern identifiziert werden, welche Rückschlüsse auf den Inhalt des Textes geben. Solche informationsträchtigen Wörter bezeichnet man als Schlüsselwörter. In dieser Arbeit sind Schlüsselwörter als bedeutungsträchtige Begriffe definiert, die aus einem Wort bestehen. Die Schlüsselwörter sollen anhand einer automatischen Indexierung ermittelt werden. Das Ergebnis ist ein Index bestehend aus Begriffen, der zusätzlich mit der Frequenz der Begriffe, d.h. der absoluten Auftrittshäufigkeit im Text, versehen ist.

Das Ergebnis der Schlüsselwortanalyse ist eine Liste von Begriffen, die alle mit der Termfrequenz gewichtet sind. Die gewichtete Liste von Schlüsselwörtern fließt in das zu generierende Benutzerprofil ein und bildet dessen Hauptteil.

Identifizierung von Kollokationen

Bei Kollokationen geht man von der Annahme aus, dass zwei aufeinander folgende Wörter eine gemeinsame Bedeutung haben, wenn sie häufig zusammen im Text vorkommen (siehe Kapitel 2.2.2). Die bisherigen Schritte des Verfahrens haben den Nachteil, dass sie nur Schlüsselwörter identifizieren können, die aus nur einem Wort bestehen. Es gibt jedoch viele Fachbegriffe, die aus zwei Wörtern zusammengesetzt sind wie beispielsweise „Information Retrieval“ oder „Recommender Systeme“. Ziel dieses Verfahrensschrittes ist es, genau solche Kollokationen zu erkennen, damit sie später im Benutzerprofil berücksichtigt werden können.

Die Identifizierung der Kollokationen im Text wird auf Basis des stoppwortfreien Textes ausgeführt. Würde man den Quelltext inklusive Stoppwörter als Grundlage verwenden, so würden als beste Ergebnisse wahrscheinlich einerseits Kombinationen von Pronomen und Verben wie „es ist“ oder „ich habe“ und andererseits Zusammensetzungen von Artikeln und Substantiven wie „das System“ oder „ein Ziel“ auftauchen. Solche Kollokationen bringen jedoch keinerlei zusätzlichen Nutzen und werden durch die Filterung der Stoppwörter beseitigt.

Das Ergebnis dieses Moduls ist eine in seiner Größe variable Liste von Begriffen, die aus zwei Wörtern bestehen. Die Gewichtung der einzelnen Kollokationen erfolgt analog zu der Gewichtung der Schlüsselwörter. Die am häufigsten im Text auftretenden Kollokationen sollen in das Benutzerprofil einfließen.

Benutzerprofil

Das Ergebnis der bisherigen Verfahrensschritte ist eine Liste von gewichteten Begriffen, die den Benutzer hinsichtlich seiner fachlichen Qualifikationen beschreiben soll. Diese

Liste, die das Resultat des Analyseteils des Algorithmus bildet, wird im Folgenden als Benutzerprofil bezeichnet.

Das Benutzerprofil setzt sich aus Daten zusammen, die aus zwei verschiedenen Verfahrenszweigen stammen:

1) Den größten Anteil am Benutzerprofil hat die gewichtete Liste von Begriffen, welche sich aus der Schlüsselwortanalyse ergibt. Es werden nicht alle Begriffe in das Benutzerprofil aufgenommen, sondern nur eine vom Benutzer festzulegende Zahl der häufigsten Begriffe.
2) Eine variable Zahl an Kollokationen wird ebenfalls in das Benutzerprofil aufgenommen. Allerdings wird der Anteil der Kollokationen im Vergleich zu den Schlüsselwörtern eher gering sein. Eine sinnvolle Relation zwischen Schlüsselwörtern und Kollokationen soll durch die Evaluation gefunden werden.

3.2.2 Verwendung der Benutzerprofile zum Matchen von Benutzern oder Anfragen

Nachdem für jeden Benutzer aus den von ihm zur Verfügung gestellten Dokumenten ein Benutzerprofil nach dem beschriebenen Verfahren generiert wurde, ist die Aufgabe des zweiten Teils einen Benutzer bzw. eine Anfrage gegen andere Benutzer zu matchen. Dieses Matching hat das Ziel, eine Ähnlichkeitsbeziehung zwischen allen Benutzern sowie evtl. einer Anfrage zu generieren. Auf Basis dieses Verfahrensschritts sollen die Empfehlungen berechnet werden. Für diesen Verfahrensteil bietet sich ein vektorbasiertes Verfahren an, da die Benutzerprofile eine vektorartige Form haben.

In diesem Teil des Verfahrens wird mit einer verfeinerten Gewichtung der Begriffe gearbeitet. Hierzu wird das Profil jedes Benutzers in das term frequency-inversed document frequency- (TF-IDF-)Maß überführt. TF-IDF setzt sich aus zwei Teilen zusammen, der Häufigkeit eines Terms in einem Dokument (term frequency) und der inversen Dokumentenfrequenz (inversed document frequency). Unter inverser Dokumentenfrequenz versteht man die Anzahl der Dokumente im Datenbestand, in denen ein bestimmter Term auftritt. Diesem Maß liegt die Annahme zugrunde, dass Terme eine höhere Signifikanz besitzen, wenn sie innerhalb eines Dokumentenbestandes nur in wenigen Dokumenten vorkommen. TF-IDF soll eine Verbesserung der Ergebnisse mit sich bringen, stellt aber keine grundlegende Änderung des Verfahrens dar.

Die Termfrequenz tf_{ij} aller Terme T_j in allen Dokumenten D_i gibt die Häufigkeit von T_j in D_i an. Die Dokumentenfrequenz df_j ist als die Anzahl der Dokumente in einem Dokumen-

tenbestand von N Dokumenten definiert, in denen der Term T_j enthalten ist. Das TF-IDF-Maß w_{ij} ergibt sich aus der Kombination von Termfrequenz und inverser Dokumentenfrequenz [Salt89].

$$w_{ij} = tf_{ij} * \log\left(\frac{N}{df_j}\right)$$

Als Verfahren für die Berechnung der Ähnlichkeitsbeziehungen soll Latent Semantic Indexing (siehe Kapitel 2.2.4) verwendet werden. Ausgangsbasis für LSI ist eine Term-Dokument-Matrix, die schematisch in Abbildung 16 dargestellt ist.

Terme	Dokumente						
	D_1	D_2	…	D_i	…	D_{n-1}	D_n
T_1							
T_2							
…							
T_j				h_{ij}			
…							
T_m							

Abbildung 16: Schematische Darstellung der Term-Dokument-Matrix

Die Term-Dokument-Matrix besteht aus n Dokumenten, die mit D_1 bis D_n bezeichnet werden sowie m Termen T_1 bis T_m. Der Matrix-Eintrag h_{ij} ist die Häufigkeit bzw. in unserem Falle die TF-IDF-Häufigkeit des Terms T_j im Dokument D_i.

Im nächsten Schritt wird diese Matrix mit dem Singular Value Decomposition-Verfahren (SVD), welches den Kern von LSI bildet, in drei Matrizen umgewandelt, die in Abbildung 17 zu sehen sind.

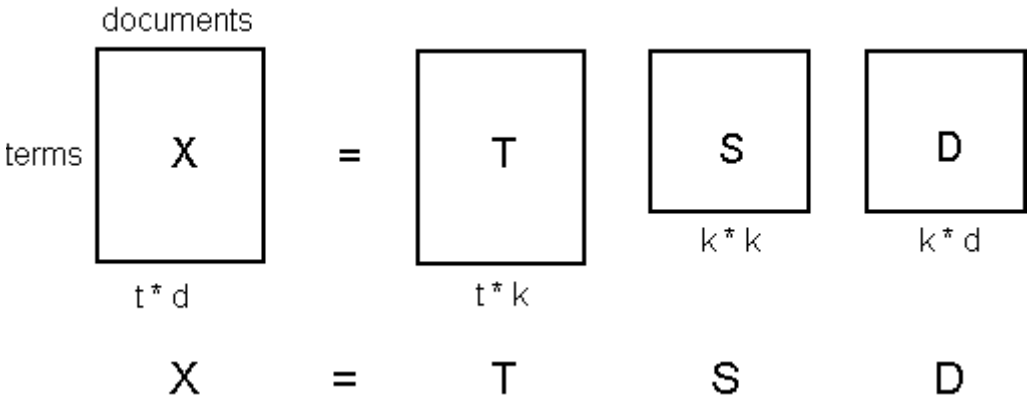


Abbildung 17: Singular Value Decomposition [Deer90]

Auf der linken Seite ist die schon beschriebene Term-Dokument-Matrix dargestellt. Diese Matrix kann, durch die drei Matrizen auf der rechten Seite beschrieben werden. Zweck der SVD-Berechnung ist es, die natürlichen Dimensionen der Term-Dokument-Matrix in

neue künstliche Dimensionen umzuwandeln, die den Vorteil der besseren Verarbeitung von Synonymen haben. Entscheidend ist die mittlere Matrix S, die eine Diagonalmatrix ist, d.h. alle Werte außerhalb der Diagonalen sind 0. Die Größe dieser Matrix richtet sich nach dem Wert von k, welcher die Anzahl der neu generierten künstlichen Dimensionen beschreibt und welcher frei wählbar ist. Wird k sehr klein gewählt, werden die Ergebnisse ungenau. Wird der Wert von k zu groß gewählt, werden viele unwichtige Informationen im Ergebnis berücksichtigt.

SVD liefert ein Maß für die Ähnlichkeit zwischen den einzelnen Dokumenten, indem die Dokumente in einem Raum angeordnet werden. Als Ergebnis von SVD erhält man also einen Raum, in dem alle Dokumente in Form von Punkten angeordnet sind. Man kann jeden Punkt im Raum auch als einen Ursprungsvektor verstehen. Das Skalarprodukt zweier Vektoren a und b ist nach [JaWa87] definiert als:

$$\vec{a} * \vec{b} = \begin{pmatrix} a_1 \\ a_2 \\ \mathrm{M} \\ a_m \end{pmatrix} * \begin{pmatrix} b_1 \\ b_2 \\ \mathrm{M} \\ b_m \end{pmatrix} = (a_1,a_2,\ldots,a_m)*(b_1,b_2,\ldots,b_m) := a_1b_1+a_2b_2+\ldots+a_mb_m$$

Der Kosinus des Winkels φ zwischen den Vektoren ist definiert als:

$$\cos(\varphi) = \frac{\vec{a} * \vec{b}}{|\vec{a}| * |\vec{b}|}$$

Das Skalarprodukt zweier Vektoren ist nun proportional zum Kosinus des Winkels zwischen den Vektoren. Wegen dieser Proportionalität kann der Kosinus des Winkels zwischen zwei Vektoren als Kriterium für die Ähnlichkeit der beiden Vektoren herangezogen werden. Im Falle dieser Diplomarbeit setzt sich die Term-Dokument-Matrix aus den verschiedenen generierten Benutzerprofilen zusammen, die die Stelle der Dokumente einnehmen, in diesem Fall muss man also richtigerweise von einer Term-Benutzer-Matrix sprechen. Für das Verfahren macht dies jedoch keinen Unterschied, da es sich in beiden Fällen um Vektoren mit einer Reihe von Begriffen handelt.

Zum besseren Verständnis sollen die Schritte des LSI-Verfahrens anhand eines einfachen Beispiels noch einmal erläutert werden. Hierzu sollen fünf Benutzer anhand ihrer Benutzerprofile auf ihre Ähnlichkeit hin verglichen werden. Die Benutzer sind mit ihren Benutzerprofilen in Tabelle 1 dargestellt. Der Einfachheit halber sind die Benutzerprofile in diesem Beispiel sehr klein. Die angegebenen Fachgebiete dienen nur der Illustration und fließen nicht in das Verfahren ein.

Name	Fachgebiet	Benutzerprofil
Jens	Wirtschaftsmathematik	*Mathematik* 35 *Wirtschaft* 13 *Statistik* 21 *Java* 10
Thomas	BWL	*BWL* 40 *Wirtschaft* 29 *Steuerlehre* 15 *Wirtschaftsrecht* 12 *Statistik* 20
Jürgen	Informatik	*Informatik* 40 *Java* 35 *UML* 30 *Softwareentwicklung* 25 *Datenbanken* 12
Bastian	BWL	*BWL* 34 *Kostenrechnung* 14 *Statistik* 23 *VWL* 30
Daniel	Wirtschaftsinformatik	*Softwaretechnik* 28 *Softwareentwicklung* 26 *CSCW* 20 *Datenbanken* 15 *Java* 23

Tabelle 1: beispielhafte Benutzerprofile

Aus den Benutzerprofilen ergibt sich die Term-Benutzer-Matrix, die den Input für das LSI-Verfahren bildet (siehe Abbildung 18).

Jede Spalte der Matrix steht für einen Benutzer und jede Zeile repräsentiert einen Term. Ist ein Term T_j in dem Benutzerprofil des Benutzers B_i enthalten, so wird die Häufigkeit an der Stelle h_{ij} mit dem entsprechenden Wert gewichtet[8]. Wenn der entsprechende Term nicht im Benutzerprofil enthalten ist, steht an dieser Stelle der Matrix eine 0.

[8] Eine weitere Vereinfachung bei diesem Beispiel ist die fehlende Umrechnung der Häufigkeiten in das TF-IDF-Maß.

	Jens	Thomas	Jürgen	Bastian	Daniel
Mathematik	35	0	0	0	0
Wirtschaft	13	29	0	0	0
Statistik	21	20	0	23	0
Java	10	0	35	0	23
BWL	0	40	0	34	0
Steuerlehre	0	15	0	0	0
Wirtschaftsrecht	0	12	0	0	0
Informatik	0	0	40	0	0
UML	0	0	30	0	0
Softwareentwicklung	0	0	25	0	28
Datenbanken	0	0	12	0	15
Kostenrechnung	0	0	0	14	0
VWL	0	0	0	30	0
Softwaretechnik	0	0	0	0	28
CSCW	0	0	0	0	20

Abbildung 18:Term-Benutzer-Matrix

Diese Matrix wird nun mittels SVD unter Angabe der Anzahl der gewünschten künstlichen Dimensionen umgerechnet, so dass k neue Dimensionen entstehen, die Cluster der alten natürlichen Dimensionen darstellen. In Abbildung 19 ist der Cluster für das Beispiel dargestellt.

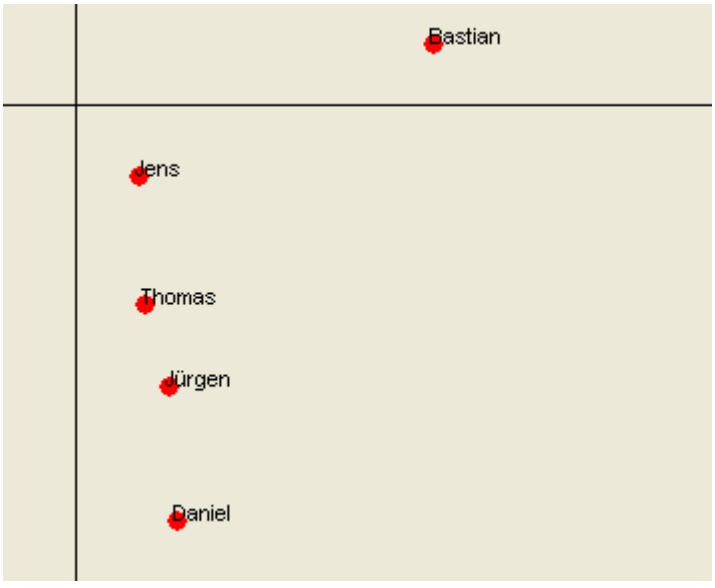


Abbildung 19: LSI-Cluster

Die multidimensionalen Vektoren sind zur Visualisierung auf zwei Dimensionen verdichtet worden. Die Ähnlichkeit zwischen zwei Benutzern lässt sich über den Winkel zwischen den beiden Ursprungsvektoren der Punkte berechnen (vgl. Abbildung 11). Je kleiner dieser Winkel ist, desto größer ist die Ähnlichkeit zwischen den Benutzern. Bezogen auf das Beispiel bedeutet das, dass Bastian und Jens sehr unähnlich sind, Jürgen und Daniel aber eine relativ hohe Ähnlichkeit aufweisen.

Neben der Untersuchung der Ähnlichkeit zwischen zwei Benutzern bietet sich auch die Möglichkeit, Benutzer auf eine Anfrage zu Matchen. Die Anfrage wird in diesem Fall auch als Vektor dargestellt und die Bestimmung der Ähnlichkeit funktioniert analog zum Matchne von zwei Benutzern. In beiden Fällen, Anfrage- und Benutzer-Matching wird eine Liste mit Empfehlungen in Form von Benutzern an den Anfrager zurückgegeben.

4 Prototypische Implementierung des Verfahrens

In diesem Kapitel wird die Implementierung eines Prototyps zum Benutzer-Matching auf Basis automatischer Textanalyse vorgestellt. Der Algorithmus soll in das ExpertFinder Framework (siehe Abschnitt 2.1.4) integriert werden und dessen Funktionalität ergänzen. Durch dieses zusätzliche Matching-Modul wird die Funktionsbreite des ExpertFinders erweitert sowie dessen Ergebnisse präzisiert.

Für die Entwicklung des Prototyps ist die Wahl der Programmiersprache zugunsten von Java [Java04] ausgefallen, da auch das ExpertFinder Framework größtenteils in dieser Sprache entwickelt wurde.

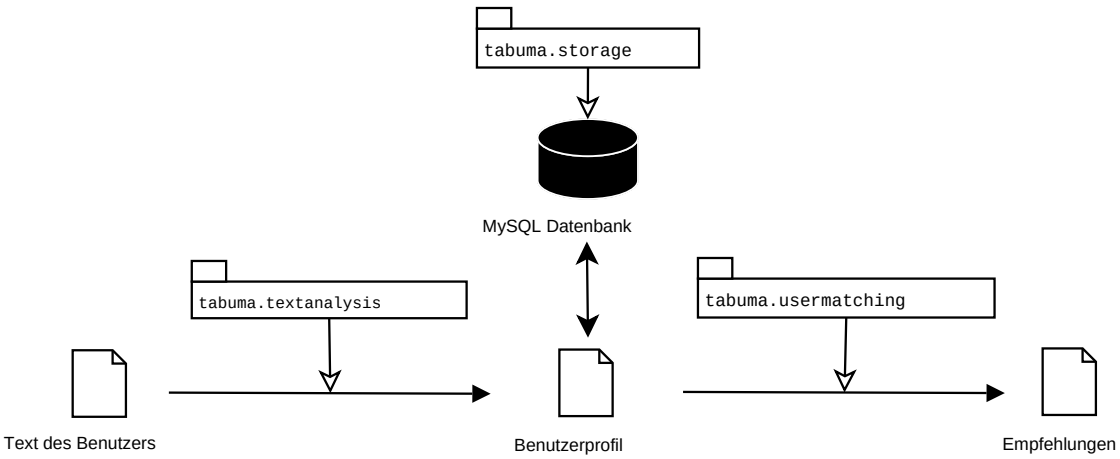


Abbildung 20: Übersicht über die Architektur des Prototyps

Der Prototyp besteht grundsätzlich aus drei Bestandteilen, einem Package, das die automatische Textanalyse realisiert, einer MySQL-Datenbank, welche die Ergebnisse der Analyse speichert und so das Bindeglied zwischen den beiden Verfahrensteilen bildet und einem Package, welches das Matching von mehreren Benutzern sowie auf Anfragen implementiert. Die Aufteilung des Verfahrens in zwei Java-Packages hat den Grund, dass die Analyse der Benutzertexte und das Matching von Benutzern oder Anfragen unabhängig voneinander sind. In Abbildung 20 ist eine schematische Übersicht über den Prototyp mit seinen drei Hauptbestandteilen dargestellt. Alle innerhalb dieses Prototyps realisierten Verfahrensteile sind in dem Package *tabuma* (**t**ext **a**nalysis **b**ased **u**ser **m**atching **a**lgorithm) gespeichert. Am Anfang des Prozesses steht der zu analysierende Text eines Benutzers. Dieser wird durch die Funktionalität des Packages *tabuma.textanalysis* analysiert und in ein Benutzerprofil in Form einer Liste von Termen überführt. Die einzelnen Benutzerprofile werden in einer MySQL-Datenbank [MySQ04] gespeichert. Die Interaktion mit der Datenbank sowie einige andere Klassen zur Speicherung von Zwischenergebnis-

sen sind im Package *tabuma.storage* implementiert. Die Funktionalität für das Benutzer-Matching wird von dem Package *tabuma.usermatching* bereitgestellt. Das Ergebnis aus diesem Prozess sind eine oder mehrere Empfehlungen für den Benutzer.
In den folgenden drei Abschnitten werden die Funktionalität des Packages *tabuma.textanalysis* (Abschnitt 4.1), die persistente Speicherung der Benutzerdaten in einer Datenbank sowie deren Interaktion mit den anderen Verfahrensteilen (Abschnitt 4.2) und die Funktionalität des Packages *tabuma.usermatching* (Abschnitt 4.3) näher erläutert.

4.1 Textanalyse

Der erste Teil des Prototyps führt die automatische Textanalyse durch. Die Implementierung dieser Funktionalität befindet sich im Package *tabuma.textanalysis*. Die automatische Textanalyse erfolgt auf Basis der in Abschnitt 2.2.2 beschriebenen statistischen Methoden, die im Bereich Information Retrieval sehr verbreitet sind. Es existieren mehrere Software-Bibliotheken wie IGLU [Iglu04] oder Lucene [Jaka04], die Funktionen zur automatischen Textanalyse zur Verfügung stellen und im Fall von IGLU auch für den Prototyp verwendet werden. In Abbildung 21 ist der Textanalyseteil in Form eines UML-Klassendiagramms dargestellt. Die einzelnen Klassen implementieren vom ExpertFinder Framework vorgegebene Interfaces und werden in den nächsten Teilabschnitten näher beschrieben. Die Klasse *AnalyzingProcess* implementiert den kompletten Textanalyseteil und stellt eine Kapselung der im Folgenden beschriebenen Stufen der automatischen Textanalyse dar.
In Abschnitt 4.1.1 wird die Realisierung der Erkennung der Textsprache beschrieben. Danach folgt die Herausfilterung der Stoppwörter aus dem Text (4.1.2). In Abschnitt 4.1.3 wird die Umsetzung der Wortstammreduzierung mittels Stemming aufgezeigt. Die beiden daran anschließenden Abschnitte befassen sich mit der Identifizierung von Schlüsselwörtern (4.1.4) und Kollokationen (4.1.5). Schließlich wird in Abschnitt 4.1.6 die Erstellung des Benutzerprofils beschrieben.

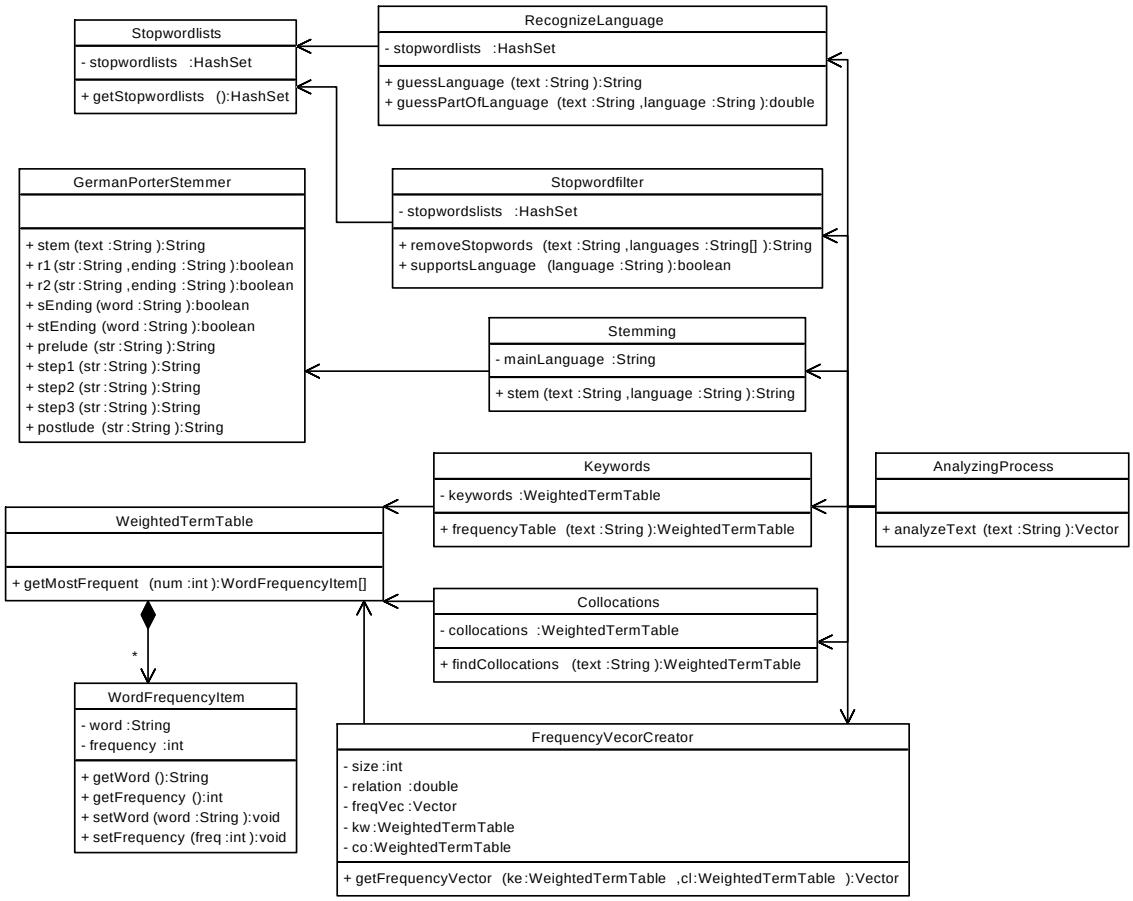


Abbildung 21: Klassendiagramm der Textanalyse

4.1.1 Spracherkennung

Der erste Schritt bei der Erstellung eines Benutzerprofils aus einer Sammlung von Texten eines Benutzers ist die Feststellung der Sprache des Textes bzw. der Sprache mit dem größten Anteil am Text. Die Bestimmung der Sprache ist notwendig, da die folgenden Schritte Stoppwortfilterung und Stemming sprachabhängig sind.

Die Bestimmung der Sprache ist in der Klasse *RecognizeLanguage* implementiert. Diese Klasse implementiert das Interface *EFLanguageDeterminer* und verwendet zur Sprachbestimmung die Klasse *Stopwordlists.*

Die Sprache des Textes wird durch den Vergleich mit den vorhandenen Stoppwortlisten ermittelt, es können also nur Sprachen erkannt werden, von denen Stoppwortlisten existieren[9]. Die Stoppwortlisten werden durch die Verwendung der Klasse *Stopwordlists* eingelesen und mit dem Anfangsteil des Textes verglichen. Wenn nur Stoppwörter aus einer Sprache erkannt werden, wird diese Sprache als Ergebnis der Methode *guessLanguage()* an den Benutzer zurückgegeben. Sollten Stoppwörter aus mehreren Sprachen in etwa

[9] Für den Prototyp sind Stoppwortlisten für die Sprachen Deutsch, Englisch und Französisch vorhanden.

gleichen Anteilen erkannt werden, keine Sprache zurückgegeben. Die Methode *guessPartOfLanguage()* gibt dem Benutzer bei mehrsprachigen Texten den ungefähren Anteil der angefragten Sprache zurück.

4.1.2 Stoppwortfilterung

Die Herausfilterung der Stoppwörter aus dem Text wird von der Klasse *Stopwordfilter* geleistet. Die Klasse implementiert das Interface *EFStopWordFilter* und benutzt die Klasse *Stopwordlists*.

Die Filterung der Stoppwörter passiert anhand eines Vergleichs des Textes mit einer oder mehreren Stoppwortlisten, je nach Anzahl der Sprachen im Text. Die Stoppwortlisten werden von der Klasse *Stopwordlists* eingelesen und so dem Stoppwortfilter zugänglich gemacht. Die Methode *supportsLanguage()* gibt dem Benutzer die Möglichkeit zu prüfen, ob eine Sprache unterstützt wird, d.h. ob eine Stoppwortliste vorhanden ist. Die Methode *removeStopwords()* leistet die eigentliche Filterung der Stoppwörter anhand einer frei wählbaren Menge von Stoppwortlisten. Für den Prototyp sind derzeit Stoppwortlisten für die Sprachen Deutsch, Englisch und Französisch implementiert (siehe Anhang A Verwendete Stoppwortlisten). Sie wurden durch Mischung und Ergänzung von schon bestehenden und im Internet verfügbaren Stoppwortlisten wie beispielsweise [Snow04b], [Snow04c] und [StUH04] erstellt.

4.1.3 Stemming

Der nächste Schritt hin zum Benutzerprofil ist die Vereinheitlichung der Wortstämme im gesamten Text. Diese Wortstammreduktion wird mittels Stemming durchgeführt und wird durch die Klasse *Stemming* implementiert. Das Interface *EFStemmer* wird durch diese Klasse implementiert. Für den Prototyp sind zwei Stemmer verfügbar, ein englischer und ein deutscher. Beide orientieren sich sehr stark am Porter Stemming Algorithmus [Port80]. Für die englische Variante wird auf die Bibliothek IGLU [Iglu04] zurückgegriffen, die eine Implementierung für einen PorterStemmer zur Verfügung stellt. Für die deutsche Sprache wurde wegen der schlechten Qualität der verfügbaren Produkte im Rahmen dieser Arbeit, auf Basis des in Abbildung 15 dargestellten Algorithmus, ein Stemmer in der Klasse *GermanPorterStemmer* implementiert[10].

[10] Bei der Implementierung der deutschen Version des Stemmers handelt es sich um eine abgeschwächte Version des Porter-Stemmers. Der Grund hiefür liegt in der Struktur der deutschen Sprache, die für das Stemming eher schlecht geeignet ist.

Da die Stemming-Funktion sprachspezifisch ist, kann ein Stemmer nur auf einen einsprachigen Text angewendet werden, da sonst unsinnige Ergebnisse produziert würden. Wenn aber ein einsprachiger Text vorliegt, kann man mittels der Methode *stem()* einen Text durch einen Stemmer in der angegebenen Sprache auf seine Wortstämme reduzieren. Erkennt der Algorithmus, dass der Text mehrsprachig ist, wird der Stemmer nicht auf den Text angewendet.

4.1.4 Schlüsselwortanalyse

Nachdem der Text von Stoppwörtern bereinigt sowie auf Wortstämme reduziert worden ist, wird nun die Häufigkeit des Vorkommens der einzelnen Wörter bestimmt. Diese Funktion wird durch die Klasse *Keywords* implementiert. Sie implementiert das Interface *EFKeywordDeterminer* und greift auf die Klasse *WeightedTermTable* zu, die eine Container-Klasse für die Speicherung von Begriffen inkl. Der jeweiligen Gewichtungen bildet. Die Analyse der Schlüsselwörter wird durch die Methode *frequencyTable()* implementiert, die im gestemmten Text die Häufigkeit jedes auftretenden Wortes bestimmt und das Ergebnis in einer *WeightedTermTable* speichert.

4.1.5 Kollokationsanalyse

Die Kollokationsanalyse ist analog zur Schlüsselwortanalyse realisiert. Für jede Kollokation wird die Häufigkeit des Autratens im Text ermittelt. Zur Speicherung der einzelnen Kollokationen wird wieder die Hilfsklassen *WeightedTermTable* verwendet. Der Unterschied besteht im Quelltext für diese Verfahrenstufe, der von Stoppwörtern befreit, aber nicht gestemmt ist. Der Grund hierfür liegt in der Tatsache, dass bei Kollokationen nur das zweite Wort verschiedene Wortendungen annimmt, wo hingegen das erste Wort immer die gleiche Form hat.[11] Wenn man einen Stemmer auf die Kollokationen anwenden würde, würden jedoch beide Wörter gestemmt, was unter Umständen zu unverständlichen Ergebnissen führen könnte. Das Finden der Kollokationen ist in der Methode *findCollokations()* implementiert, die alle Kollokationen im Text zählt und diese in einer *WeightedTermTable* speichert.

[11] Ein Beispiel für verschiedene Formen einer Kollokation ist: Recommender Systeme, Recommender System, Recommender Systemen. Das erste Wort Recommender hat immer die gleiche Form, die auch bei der Reduktion der Kollokation auf ihre Stammform beibehalten werden sollte. Ein Stemmer würde das erste Wort jedoch genauso reduzieren wie das zweite Wort, wodurch die Kollokation unverständlich werden könnten.

4.1.6 Benutzerprofilerstellung

Der letzte Schritt der automatischen Textanalyse besteht in der Generierung eines Benutzerprofils. Dieses Profil setzt sich aus Schlüsselwörtern und Kollokationen zusammen. Die Größe des Benutzerprofils, d.h. die absolute Zahl der Terme (Schlüsselwörter + Kollokationen) ist vom Benutzer frei wählbar. Weiterhin ist das zahlenmäßige Verhältnis zwischen Schlüsselwörtern und Kollokationen frei wählbar, da der optimale Wert erst während der Evaluation bestimmt wird.

Die Erstellung des Benutzerprofils wird durch die Klasse *FrequencyVectorCreator* implementiert. Die Methode *getFrequencyVector()* erzeugt aus der Menge von Schlüsselwörtern und Kollokationen einen Vector, der das Benutzerprofil in Form einer Reihe von Termen generiert.

4.2 Persistente Speicherung der Daten

Die persistente Speicherung der Benutzerprofile in einer Datenbank ist notwendig, da die Ergebnisdaten der automatischen Textanalyse nicht direkt als Quelldaten für das Benutzer-Matching verwendet werden können. Die Textanalyse liefert ein einziges Benutzerprofil, die Matching-Komponente braucht als Informationsquelle aber mehrere Benutzerprofile. Damit nicht bei jedem Matching Vorgang alle Benutzer neu analysiert werden müssen, werden die Benutzerprofile in einer MySQL-Datenbank gespeichert. Die Datenbank stellt somit die Schnittstelle zwischen den beiden Verfahrensteilen automatische Textanalyse und Benutzer-Matching dar.

Für die Speicherung der Benutzerprofile ergibt sich das in Abbildung 22 dargestellte Entity-Relationship (ER) Diagramm. Die Entität *UserProfile* repräsentiert ein Benutzerprofil und hat die Attribute *userID* als eindeutigen Bezeichner sowie Name des Benutzers (*name*) und Datum der Benutzerprofilerstellung (*date*). Ein Benutzerprofil beinhaltet mehrere gewichtete Terme, die durch das Attribut *weightedTermID* eindeutig bezeichnet sind. Weiterhin enthält ein gewichteter Term einen Begriff (*term*) sowie eine Gewichtung (*frequency*)

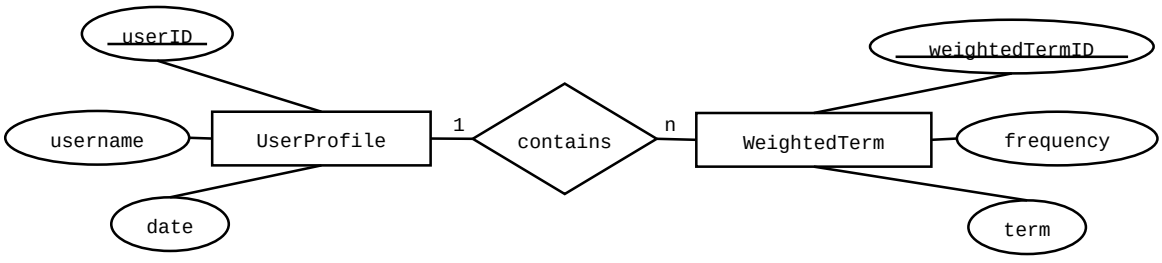


Abbildung 22: ER-Diagramm der Benutzerdaten

Aus dem ER-Diagramm ergeben sich folgende Tabellen für die Datenbank:

userprofile: {userID, username, date}

weightedterm: {weightedTermID, term, frequency, userID}

Der Zugriff auf die Datenbank erfolgt über eine Datenbank-Schnittstelle, die durch die Klasse *DBHandler* realisiert ist (siehe Abbildung 23). Als Container für die Benutzerdaten werden die beiden Hilfsklassen *UserProfile* und *WordFrequencyItem* verwendet. Für die initiale Erstellung der Datenbank-Tables wird zudem die Klasse *InitTables* angeboten.

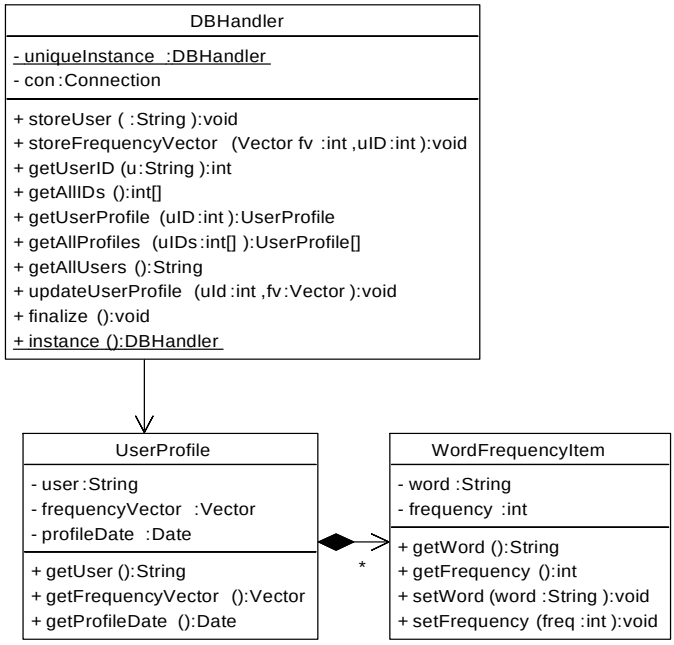


Abbildung 23: Klassendiagramm der Speicherung

Der DBHandler benutzt das Singleton-Pattern [Gamm95], welches sicherstellt, dass immer nur ein Objekt des Typs DBHandler existiert. Diese Maßnahme soll Inkonsistenzen vermeiden. Das Singleton-Pattern wird durch das Klassen-Attribut *uniqueInstance* sowie die Klassen-Methode *instance()* realisiert.

Zur Speicherung eines Benutzerprofils in der Datenbank werden die Methoden *storeUser()* und *storeFrequencyVector()* bereitgestellt. Mit der ersten Methode wird ein Benutzer in der Datenbank gespeichert, in der zweiten Methode wird unter Angabe der Benutzer-ID, die bei der Speicherung des Benutzers vergeben wurde, ein Vektor mit gewichteten Schlüsselwörtern gespeichert. Für den Fall, dass für einen Benutzer schon ein Profil in der Datenbank vorhanden ist, bietet die Methode *updateUserProfile()* die Möglichkeit, dessen Profil durch neue Daten zu ergänzen.

Um die Benutzerprofile zur weiteren Verarbeitung wieder aus der Datenbank zu extrahieren, wird vom DBHandler die Methode *getAllProfiles()* angeboten, die ein Array mit al-

len gespeicherten Benutzerprofilen zurück gibt. Diese Methode wird für die Erstellung der Term-Benutzer-Matrix benötigt.
Die Methode *finalize()* hat lediglich – entsprechend der Java-Spezifikation - die Aufgabe, die Verbindung mit der Datenbank zu beenden.

4.3 Benutzer-Matching

Der letzte Teil des Verfahrens hat die Aufgabe, verschiedene Benutzer auf ihre Ähnlichkeit zu vergleichen sowie auf eine Anfrage geeignete Benutzer zu finden. Die Implementierung dieses Prozesses wird durch das Package *tabuma.usermatching* realisiert. Die UML-Darstellung dieses Packages ist in Abbildung 24 dargestellt. Die Klasse *MatchingProcess* implementiert den gesamten Prozess des Matchings, vom Erstellen der Term-Benutzer-Matrix bis hin zum eigentlichen Matching-Vorgang.
Die Methode *executeUserMatching()* erstellt zu einem übergebenen Benutzer eine Liste anderer Benutzer, deren Profil ähnlich ist. Die Methode *executeQueryMatching()* generiert auf Basis einer Anfrage eine Liste von Benutzern, deren Profil Ähnlichkeiten zur Anfrage aufweist. Die Anfrage wird zu diesem Zweck in einen Vektor umgewandelt und bei der weiteren Verarbeitung wie ein Benutzerprofil behandelt.
Zur Speicherung der Ergebnisse steht die Hilfsklasse *DoubleWeightedTermTable* zur Verfügung, die eine Liste von Benutzern mit ihren Ähnlichkeitswerten zur Anfrage speichert. Diese Klasse ist zur *WeightedTermTable* nahezu identisch, einzig die Ähnlichkeiteswerte sind rationale und keine natürlichen Zahlen. Die Methode *getAllItems()* liefert alle Einträge zurück, die Methode *getMoreFrequent()* gibt alle Einträge ab einem gewissen Schwellenwert zurück und *getMostFrequent()* stellt dem Benutzer eine anzugebende Zahl der höchstwertigen Benutzer zur Verfügung.
Das Matching der Benutzer basiert auf deren Benutzerprofilen, welche in Form von Vektoren vorliegen. Als Verfahren für das Benutzer-Matching wurde Latent Semantic Indexing (LSI) ausgewählt, da es insbesondere im Umgang mit Synonymen Vorteile verspricht. LSI wird auf Basis von Matrizen ausgeführt. Bei der Behandlung dieser Matrizen in Java wurde auf die Java-Bibliothek JAMA [Jam04] zurückgegriffen, die einige notwendige Funktionen aus diesem Kontext zur Verfügung stellt[12].
Der erste Abschnitt 4.3.1 beschreibt den gesamten Matching-Prozess bis zur Erstellung der Empfehlung. Die Generierung der Term-Benutzer-Matrix aus den verschiedenen Be-

[12] Innerhalb des Prototyps werden die Klassen *Matrix* und *SingularValueDecomposition* verwendet.

nutzerprofilen ist Gegenstand von Abschnitt 4.3.2. In Teil 4.3.3 wird die Umsetzung des zentralen Schrittes von LSI, der Singular Value Decomposition (SVD) dargestellt.

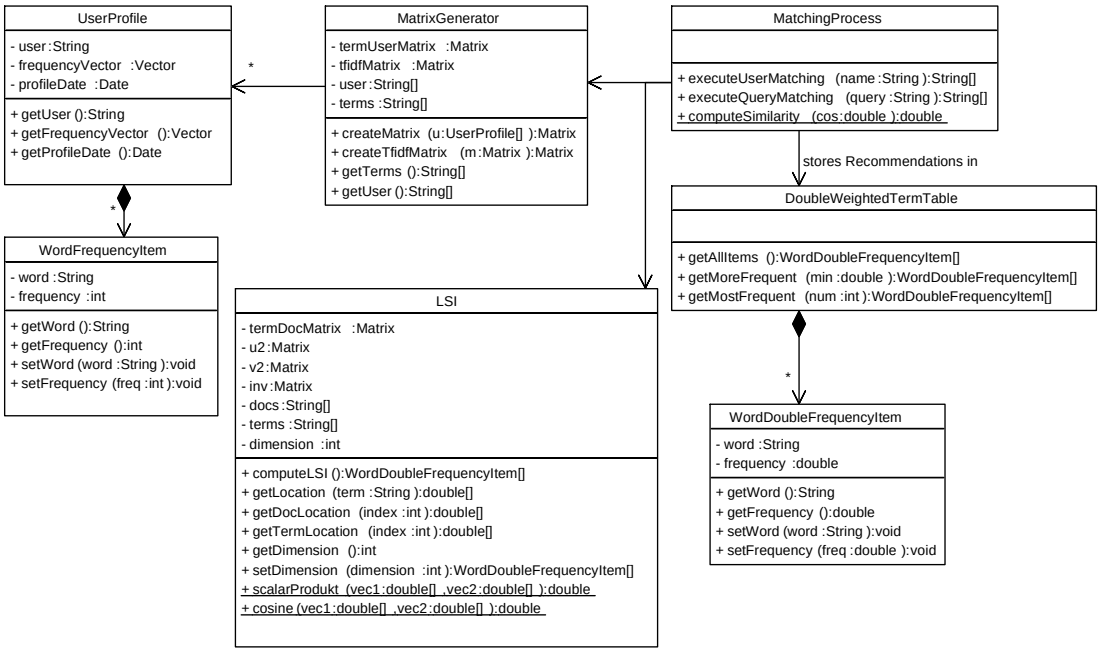


Abbildung 24: Klassendiagramm des Matchings

4.3.1 Matching von Benutzern oder Anfragen

Der gesamte Matching-Prozess wird durch die Klasse *MatchingProcess* implementiert. In dieser Klasse wird zunächst aus den einzelnen Benutzerprofilen eine Matrix generiert. Auf Basis dieser Term-Benutzer-Matrix wird anschließend die LSI-Berechnung durchgeführt.

Die Methode *executeUserMatching()* führt eine Berechnung der Ähnlichkeit zwischen einem ausgewählten Benutzer zu allen anderen Benutzern aus. Als Ergebnis wird eine Liste der ähnlichsten Benutzer zurückgegeben. Die Methode *executeQueryMatching()* gibt zu einer Anfrage eine Empfehlung in Form einer Liste von Benutzern aus. Die Bestimmung der Ähnlichkeit zwischen zwei Benutzern wird durch die Methode *computeSimilarity()* implementiert, welche das Ergebnis von LSI in ein nachvollziehbares Ähnlichkeitsmaß umrechnet. Die Ergebnisse aus beiden Methoden werden in Form einer *DoubleWeigthedTermTable* zurückgegeben.

4.3.2 Erstellung der Term-Benutzer-Matrix aus den Benutzerprofilen

Der erste notwendige Schritt zum Matching der Benutzer ist es, die verschiedenen Benutzerprofile in eine Term-Benutzer-Matrix zu überführen, welche die Eingabe für LSI darstellt.

Diese Funktionalität wird durch die Klasse *MatrixGenerator* implementiert. Die Methode *createMatrix()* erstellt aus einer Reihe von Benutzerprofilen, welche in Form eines Arrays übergeben werden, die Term-Benutzer-Matrix. In dieser Funktion werden zusätzlich zwei Arrays mit den Benutzern und den Termen gespeichert, auf welche über die Methoden *getTerms()* und *getUser()* zugegriffen werden kann. Die erstellte Matrix ist mit der absoluten Termfrequenz gewichtet, d.h. es sind alle Terme mit der Häufigkeit ihres Vorkommens im Benutzertext dargestellt. Die in Kapitel 3.2.2 vorgestellte TF-IDF-Gewichtung wird von der Methode *createTfidfMatrix()* geleistet, welche die Ausgangs-Matrix in diese Form umrechnet.

4.3.3 Latent Semantic Indexing

Die Funktionalität von LSI wird durch die Klasse *LSI* implementiert. Der grundlegende Rechenschritt, die Singular Value Decomposition (SVD), wird hierbei durch das JAMA-Package zur Verfügung gestellt.

Die Methode *computeLSI()* führt SVD aus und stellt die Ergebnismatrizen zur Verfügung. Durch die Methoden *getDocLocation()* und *getTermLocation()* kann jetzt die Position eines Benutzers bzw. einer Anfrage innerhalb des Ergebnisclusters bestimmt werden. Die Dimension des Ergebnisclusters kann vom Benutzer im Konstruktor manipuliert werden. Die statischen Methoden *scalarProduct()* und *cosine()* stellen die Funktionalität zur Verfügung, um für zwei Benutzerprofile einen Ähnlichkeitswert zu ermitteln.

5 Evaluation

Der Inhalt der Evaluation ist auf der einen Seite die Bestimmung der optimalen Konfiguration des Prototyps und auf der anderen Seite die Untersuchung der Qualität des entwickelten Verfahrens. Die Qualität des Verfahrens wird anhand der beiden Merkmale Ergebnisgüte und Performance bestimmt.
In Abschnitt 5.1 werden die mit der Evaluation verfolgten Ziele genauer beschrieben sowie Methoden zur Durchführung dargestellt. Abschnitt 5.2 werden die Evaluationsergebnisse vorgestellt. Schließlich erfolgt in Abschnitt 5.3 eine Diskussion der Ergebnisse.

5.1 Ziele und Methodik der Evaluation

Das Ziel dieser Evaluation ist es einerseits, die bestmögliche Konfiguration der Verfahrensparameter zu bestimmen, und andererseits, Aussagen bezüglich der Brauchbarkeit des entwickelten Algorithmus zu treffen. Zwei Merkmale stehen hierbei im Mittelpunkt:

- Die Güte der Ergebnisse stellt den wichtigsten Indikator für die Qualität und somit auch die Anwendbarkeit des Verfahrens dar.
- Die Performance des Prototyps, also die Laufzeit der einzelnen Funktionen, ist eine zweite wichtige Eigenschaft, die für den Einsatz des Verfahrens überaus wichtig ist.

Zunächst wird in Abschnitt 5.1.1 die generelle Vorgehensweise der Evaluation dargestellt. In Abschnitt 5.1.2 werden daraufhin die kritischen Parameter aufgezeigt, die das Ergebnis maßgeblich beeinflussen. Gegenstand von Abschnitt 5.1.3 sind die innerhalb der Evaluation verwendeten Testdaten.

5.1.1 Vorgehensweise

Die Evaluation soll eine Bewertung des in dieser Arbeit entwickelten Verfahrens zum Benutzer-Matching auf Basis automatischer Textanalyse liefern. Dazu werden Testdaten mittels des entwickelten Prototyps analysiert und gematcht. Die Ergebnisse werden anschließend auf ihre Qualität sowie ihre Performanz hin untersucht.
Als Maßstab für die Ergebnisgüte fungiert auf der einen Seite die Einschätzung eines Experten, der mit den Testkandidaten vertraut und so in der Lage ist, die vom Prototypen generierten Empfehlungen auf ihre Qualität hin zu überprüfen. Auf der anderen Seite sollen die Benutzer selbst eine Empfehlung über die ihnen ähnlichen anderen Benutzer erstellen. Die vom Prototyp generierte Empfehlung wird mit den beiden Referenzen verglichen und mit Hilfe der City-Block-Metrik [Back96] eine Unähnlichkeit errechnet.

Hierbei werden die Ergebnisse als Vektoren der Empfehlungen aufgefasst. Die Vektoren enthalten zu jedem Benutzer den Rang, den er in der Empfehlung einnimmt. Die Distanz zwischen den beiden Punkten, die von den Vektoren beschrieben werden, kann Aufschluss über die Ähnlichkeit zwischen zwei Empfehlungen geben. Je größer diese Distanz ist, desto weniger ähneln sich die Ergebnisse, weshalb man von Unähnlichkeit spricht. Die Distanz d der City-Block-Metrik ist folgendermaßen definiert:

$$d_{kl} = \sum_{r=1}^{R} \left| x_{kr} - x_{lr} \right|$$

mit d_{kl} : Distanz der Punkte k und l

x_{kr}, x_{lr} : Koordinaten der Punkte k und l auf der r-ten Dimension (r=1,2,…,R)

Mit Hilfe dieser Metrik lässt sich die absolute Abweichung zweier Empfehlungen messen. Diese absolute Abweichung repräsentiert die Summe der Rangabweichungen der einzelnen, in der Empfehlung befindlichen, Benutzer. Neben dieser absoluten Abweichung kann mit mittels der City-Block-Metrik auch die durchschnittliche Abweichung aller Personen in zwei verschiedenen Empfehlungen ermitteln.

Für die Evaluation wurden aus der Testgruppe „Publikationen“ 10 Benutzer ausgewählt, die mutmaßlich die besten Selbsteinschätzungen liefern, da sie die meisten anderen Benutzer der Gruppe kennen und fachlich einschätzen können. Die ausgewählten Benutzer haben selbst eine Einschätzung über die fachlichen Ähnlichkeiten zu den anderen Benutzern der Gruppe vorgenommen. Zusätzlich wurde als zweite Referenz die Meinung eines Experten[13] hinzugezogen, der seinerseits Empfehlungen über die Ähnlichkeiten der Benutzer der Gruppe „Publikationen“ vorgenommen hat.

Die Performance des Prototyps wird durch eine mehrmalige Durchführung des Verfahrens bestimmt. Hierbei wurde für die Teilbereiche des Verfahrens die benötigte Zeit gemessen.

5.1.2 Evaluationsparameter

Bei der Benutzung des Prototyps ist eine Konfiguration diverser Parameter anzugeben, die das Ergebnis maßgeblich beeinflussen. Im Rahmen dieser Evaluation wird eine sinnvolle Konfiguration dieser Parameter ermittelt. Die anzugebenden Parameter stammen

[13] Bei dem Experten handelt es sich um Prof. Dr. Volker Wulf, der aufgrund seiner zentralen Stellung in dieser Testgruppe am qualifiziertesten ist, die anderen Gruppenmitglieder einzuschätzen.

sowohl aus dem Bereich der Textanalyse als auch aus dem Matching-Teil. Im Laufe der Evaluation soll eine optimale Konfiguration für diese Einflussgrößen gefunden werden.
Für das Verfahren sind zwei wichtige Parameter vom Benutzer anzugeben:

- Die *Vektorgröße des Benutzerprofils* bestimmt, wie viele Terme im Benutzerprofil gespeichert werden. Ist dieser Wert zu klein gewählt, fehlen möglicherweise aussagekräftige Schlüsselwörter im Benutzerprofil. Bei einem zu großen Wert dieses Parameters bestehen einerseits die Gefahr von Performanz-Problemen im weiteren Verlauf des Verfahrens und andererseits die Wahrscheinlichkeit der Existenz von vielen Termen, die nur in geringem Maße Aufschluss über den Benutzer geben.
- Die *LSI-Dimension* bestimmt die Anzahl der neuen künstlichen Dimensionen, die durch LSI erzeugt werden. Wird die Zahl der Dimensionen zu klein gewählt, werden in der Empfehlung wahrscheinlich große Teile der Benutzerprofile in der Empfehlung gar nicht berücksichtigt. Wird diese Zahl zu groß gewählt, kann die Performance darunter leiden.

Neben diesen beiden entscheidenden Parametern wird im Laufe der Testphase des Verfahrens weiterhin ein sinnvolles Verhältnis zwischen Schlüsselwörtern und Kollokationen im Benutzerprofil sowie das geeignete Maß für die Gewichtung der Begriffe ermittelt werden.

5.1.3 Testdaten

Für die Evaluation wurden zwei Testgruppen gebildet, an denen das entwickelte Verfahren durchgespielt werden soll. Die beiden Gruppen von Testdaten unterscheiden sich durch verschiedene Charakteristiken, die Aufschluss über eine gute Konfiguration sowie über die Brauchbarkeit des Verfahrens geben sollen:

1) **Publikationen**

   In dieser Gruppe werden Dokumente der Fachgruppe „Wirtschaftsinformatik – Kooperations- und Mediensysteme“ sowie einiger Mitarbeiter des Fraunhofer Instituts für angewandte Informationstechnik (FhG FIT) analysiert. Charakteristisch für diese Gruppe sind zum einen die relativ große fachliche Nähe der einzelnen Autoren untereinander sowie die Mehrsprachigkeit der Texte. Anhand dieser Gruppe wird die Güte des Verfahrens bestimmt, da die Benutzer dieser Gruppe größtenteils miteinander bekannt sind und somit durch Selbsteinschätzungen eine Referenz liefern können, die Aufschluss über die Qualität der Verfahrens gibt. Weiterhin wird diese Gruppe zu Untersuchungen bezüglich der Performanz des

Verfahrens herangezogen. Eine Übersicht über die Testgruppe „Publikationen“ ist in Tabelle 2 dargestellt.

Benutzer	Dokumente	Textumfang (in Worten)[14]
Andreas Becks	6	22848
Björn Golombek	2	19667
Joachim Hinrichs	1	8585
Britta Hofmann	1	4602
Helge Kahler	1	7886
Ralf Klamma	17	75568
Bernhard Nett	9	50454
Reinhard Oppermann	7	33954
Andreas Pfeifer	2	22353
Volkmar Pipek	4	39269
Tim Reichling	2	9579
Markus Rohde	18	131512
Barbara Schmidt-Belz	6	20619
Michael Spenke	8	20730
Gunnar Stevens	3	16054
Oliver Stiemerling	5	38680
Markus Won	1	7420
Volker Wulf	26	197852

Tabelle 2: Übersicht über die Testgruppe Publikationen

2) **Various**

Die Testgruppe „Various“ zeichnet sich vor allem durch die heterogenen fachlichen Qualifikationen der einzelnen Personen aus. Wie in Tabelle 3 dargestellt ist, stammen die verschiedenen Personen aus sehr unterschiedlichen Fachgebieten. Die Ähnlichkeit innerhalb dieser Testgruppe wird mutmaßlich relativ gering sein. Eine weitere Besonderheit dieser Gruppe besteht in der Einsprachigkeit der Dokumente, die alle in deutscher Sprache verfasst sind.

Die Untersuchung der Verfahrensergebnisse dieser Gruppe soll Aufschluss über die bestmögliche Parametereinstellung im Bereich der automatischen Textanalyse geben. Da die Personen innerhalb dieser Testgruppe größtenteils nicht miteinander bekannt sind, werden die Ergebnisse zwar zur Untersuchung der Performanz des Prototyps nicht aber zur Bestimmung der Ergebnisgüte verwendet.

Benutzer	Fachgebiet	Textumfang (in Worten)
Jan	Wirtschaftsmathematik	1898
Jörg	Germanistik	47888
Thomas	Betriebswirtschaftslehre	28848
Michael	Wirtschaftsinformatik	41766
Tim	Informatik	24486

[14] Eine Seite unformatierter ASCII-Text entspricht in deutsch etwa 400 Wörtern. In dieser Diplomarbeit stehen auf einer Seite durchschnittlich etwa 240 Wörter.

Benutzer	Fachgebiet	Textumfang (in Worten)
Matthias	Elektrotechnik	23121
Heike	Sozialwesen	17494
Jan Heiko	Biologie	25001
Jürgen	Wirtschaftsinformatik	49467

Tabelle 3: Übersicht über die Testgruppe Various

5.2 Ergebnisse der Evaluation

In diesem Abschnitt werden die Ergebnisse der Evaluation dargestellt. In Abschnitt 5.2.1 wird die innerhalb der Evaluation bestimmte Parameterkonfiguration vorgestellt. Die Ergebnisse der Untersuchungen zur Güte des Verfahrens sind Gegenstand von Kapitel 5.2.2. Abschnitt 5.2.3 beschäftigt sich mit der ermittelten Performance des Prototyps.

5.2.1 Konfiguration der Parameter

Eine sinnvolle Konfiguration der Parameter im Bereich der automatischen Textanalyse wurde in erster Linie durch die Untersuchung der Testgruppe „Various“ ermittelt. Die Testreihen haben ergeben, dass sich bei einer Textmenge von weniger als 100000 Wörtern eine Vektorgröße von etwa 500 Begriffen zur Erfassung der zentralen Begriffe ausreicht. Wenn die Textmenge den Wert von 100000 Wörtern deutlich überschreitet, sollte die Größe des Vektors auch vergrößert werden, da sonst relativ hoch gewichtete Begriffe nicht im Benutzerprofil auftauchen. Während der abschließenden Tests zur Bestimmung der Ergebnisgüte wurden bei Textmengen von weniger als 100000 Wörtern Vektoren von 500 Begriffen benutzt. Bei größeren Textmengen die Vektorgröße auf 1000 Begriffe gesteigert.

Für das Verhältnis zwischen Schlüsselwörtern und Kollokationen, wurde im Laufe der Testphase ein Wert von 0.9 ermittelt. Mit dieser Relation wurden während der Tests alle aussagekräftigen Kollokationen in das Benutzerprofil aufgenommen, gleichzeitig konnte so die Zahl der Kollokationen ohne gemeinsame Bedeutung der beiden Wörter auf einen geringen Anteil beschränkt werden. Bei einer Vektorgröße von 500 Begriffen beläuft sich die Aufteilung von Schlüsselwörtern und Kollokationen auf 450 Schlüsselwörter und 50 Kollokationen.

Im Bereich des Matchings hat sich das TF-IDF-Gewichtungsmaß als sinnvoll erwiesen, da es neben der Frequenz der Terme in einzelnen Dokumenten auch die Zahl der Dokumente berücksichtigt. Dies führt dazu, dass Fachbegriffe höher gewichtet werden als allgemeinsprachliche Begriffe, die in vielen Dokumenten vorkommen. Diese Beobachtung deckt sich mit Empfehlungen aus der Literatur.

Die optimale Zahl der mittels LSI erzeugten künstlichen Dimensionen wurde anhand der Unähnlichkeiten zwischen den Ergebnissen der Prototyps und den Selbst- bzw. Experteneinschätzungen der Gruppe „Publikationen“ ermittelt. Um die optimale Zahl der LSI-Dimensionen zu erreichen, wurden das Verfahren bei verschiedenen Konfigurationen durchgeführt, und die Ergebnisse mit den Referenzempfehlungen verglichen. In Abbildung 25 sind die Unähnlichkeiten der Verfahrensergebnisse zu den Referenzen[15] bei verschiedenen LSI-Dimensionen dargestellt. Bei den dargestellten Unähnlichkeiten handelt es sich um die durchschnittliche Zahl der Abweichungsschritte in kompletten Empfehlungen. Es wird deutlich, dass das Ergebnis des Verfahrens bei geringer Zahl von Dimensionen den beiden Referenzen immer ähnlicher wird. Im Laufe der Evaluation hat sich die Wahl einer Dimension von 3 als sinnvoll für diese – recht überschaubare - Gruppe erwiesen.

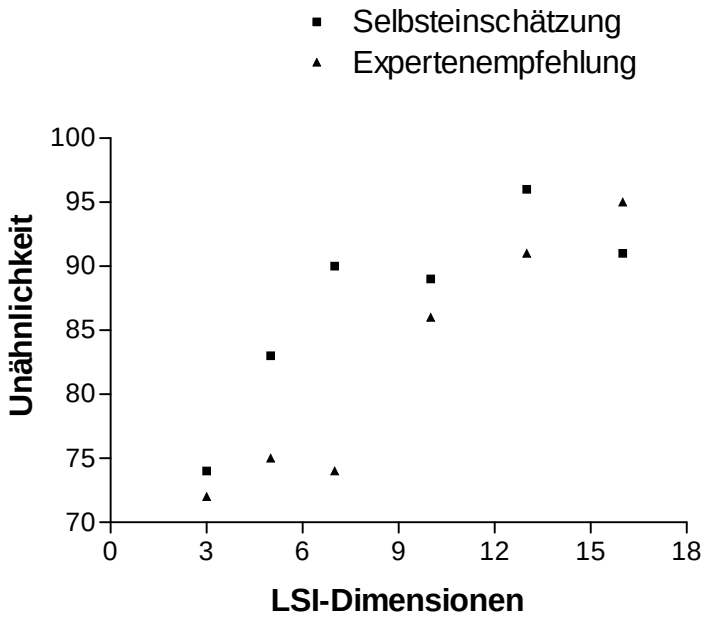


Abbildung 25: Bestimmung der optimalen Zahl von LSI-Dimensionen

5.2.2 Güte der Testergebnisse

Die Güte der Ergebnisse stellt den wichtigsten Indikator für einen erfolgreichen Einsatz des Verfahrens in der Praxis dar. Im vorigen Kapitel wurde eine sinnvolle Konfiguration für die verschiedenen Parameter des Verfahrens aufgezeigt. Auf Basis dieser Konfiguration ergeben sich für die Testgruppe „Publikationen“ die in Abbildung 26 dargestellten durchschnittlichen Abweichungen zwischen den Verfahrensergebnissen und den Selbst- bzw. Expertenempfehlungen.

[15] Die hier dargestellte Empfehlung des Experten erfolgte mit Kenntnis der analysierten Dokumente.

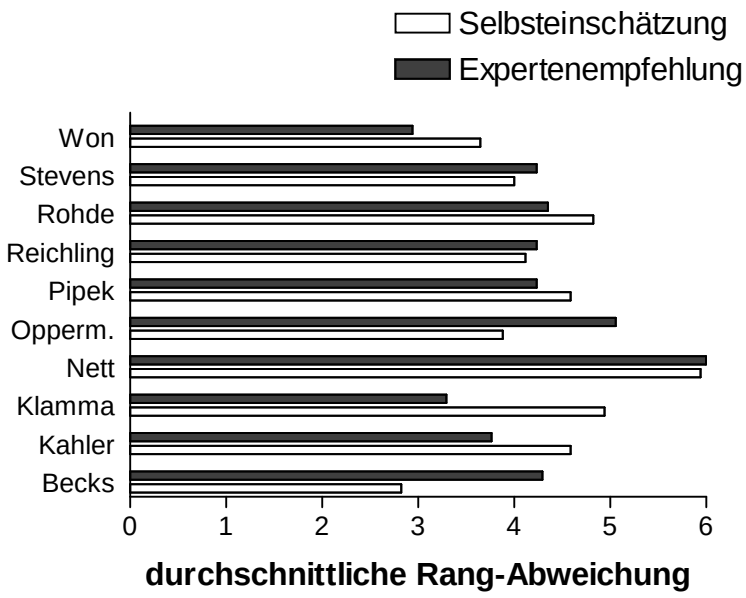


Abbildung 26: Durchschnittliche Abweichung in den Empfehlungen

Um diese Größe zu berechnen, wurde für jede in der Verfahrensempfehlung befindliche Person der Rang innerhalb der Empfehlung ermittelt. Dieser Rang wurde mit den entsprechenden Rängen der Person in der Selbsteinschätzung und der Expertenempfehlung verglichen und die Abweichung berechnet. In der Abbildung ist die durchschnittliche Abweichung von allen Benutzern innerhalb der Empfehlung dargestellt. Um die Werte der Abweichungen richtig einordnen zu können sei die durchschnittliche Abweichung zwischen den Selbsteinschätzungen und der Expertenempfehlung erwähnt, die 4,15 Ränge beträgt.

Die Höhe des zwischen den Empfehlungen des Verfahrens und den Selbst- bzw. Expertenempfehlungen bestehenden Betrags der Abweichung ist teilweise durch folgende Faktoren zu erklären:

- Sowohl bei den Selbsteinschätzungen als auch bei den Expertenempfehlungen handelt es sich um subjektive Eindrücke, die eine Person von anderen Personen hat. Diese Eindrücke können von Benutzer zu Benutzer sehr verschieden sein. Ein Beleg dafür ist die Tatsache, dass die Diskrepanz zwischen den Selbsteinschätzungen und der Meinung des Experten teilweise weitaus größer war als die Abweichung zu den Ergebnissen des Verfahrens.
- Die Selbsteinschätzungen der Testgruppe „Publikationen“ sind teilweise nicht vollständig durchgeführt worden. Das liegt daran, dass einige Personen dieser Gruppe nicht jedes andere Gruppenmitglied kannten. Diese unvollständigen Emp-

fehlungen sind zufällig mit den unbekannten Personen aufgefüllt worden, da für die Bestimmung der Unähnlichkeiten vollständige Empfehlungen notwendig sind.

- Ein weiterer Punkt, der die Ergebnisse in negativer Weise beeinflussen könnte, ist relativ kleine Menge von Dokumenten, die der Gruppe „Publikationen“ zugrunde liegen. Diese wenigen Dokumente spiegeln mit hoher Wahrscheinlichkeit nur einen Teil der fachlichen Qualifikationen eines Benutzers dar. Die Selbsteinschätzungen und die Expertenmeinung basieren jedoch auf dem vollständigen persönlichen Einruck, den man von der jeweiligen Person hat. Um diese Vermutung zu untermauern wurden vom Experten zwei Empfehlungen abgegeben. Die erste Empfehlung wurde analog zu den Selbsteinschätzungen allein auf Basis der Kenntnis der Personen erstellt. Die zweiten Expertenempfehlungen wurden mit der Kenntnis der für die Evaluation bereitgestellten Dokumente durchgeführt. Das Ergebnis dieses Versuchs war die Beobachtung, dass die zweite Expertenmeinung den Verfahrensergebnissen ähnlicher ist als die erste Empfehlung.
- Die Gruppe „Publikationen“ wurde auf Basis von Dokumenten in verschiedenen Sprachen analysiert. Es besteht die Möglichkeit, dass für zwei Personen, die über nahezu identische fachliche Qualifikationen verfügen, eine relativ geringe Ähnlichkeit ermittelt wird, da sie ihre Dokumente in verschiedenen Sprachen verfasst haben.

5.2.3 Performance

Die Performance, d.h. das Laufzeitverhalten ist ein wesentlicher Indikator für die Qualität eines Algorithmus. Um das Laufzeitverhalten zu ermitteln, wurde das entwickelte Verfahren jeweils fünfmal auf die Testdaten angewendet[16]. Mit Hilfe von Java-Systemfunktionen konnte die benötigte Zeit in Millisekunden gemessen werden. Aus den ermittelten Zeiten wurde anschließend der Mittelwert berechnet.

In Abbildung 27 ist das Laufzeitverhalten der automatischen Textanalyse im Verhältnis zur Größe des Eingabetextes dargestellt[17]. Die Geschwindigkeit der Textanalyse sinkt mit zunehmender Textmenge ab. Bei kleineren Texten liegt die Geschwindigkeit bei über 12000 Wörtern pro Sekunde, bei sehr großen Textbeständen von über 150000 Wörtern sinkt sie auf etwa 100 Wörter pro Sekunde. Die zeitkritischen Teile der Textanalyse sind die Stoppwortfilterung und das Stemming, welche etwa 95% der Gesamtzeit ausmachen.

[16] Die Tests wurden auf einem Intel Pentium 4 CPU 1.80 GHz mit 512 MB RAM durchgeführt.
[17] Die einzelnen Ergebnisse der Performanz-Analyse sind in Anhang D Ergebnisse der Performance-Analyse dargestellt.

Die Speicherung sowie das Laden der Benutzerprofile in die Datenbank laufen im Bereich von wenigen Zehntel-Sekunden ab und werden deshalb hier nicht extra aufgeführt.

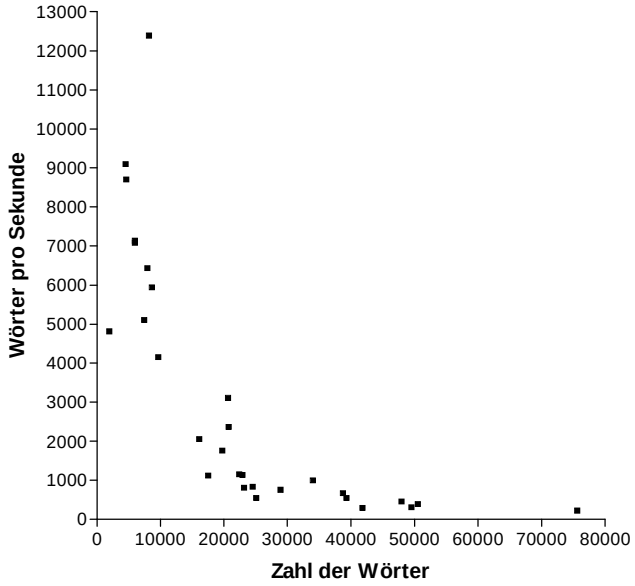


Abbildung 27: Performance der Textanalyse

Die Performance des Matchings ist in Abbildung 28 dargestellt. Es wird deutlich, dass die Durchführung von Benutzer-Matchings wesentlich performanter gemacht werden kann, wenn die Term-Benutzer-Matrix initial beim Programmstart erzeugt wird. Dem Graph ist zu entnehmen, dass die Dauer des Matchings sich durch die nicht mehr notwendige Erzeugung der Term-Benutzer-Matrix um ein Vielfaches reduzieren lässt.

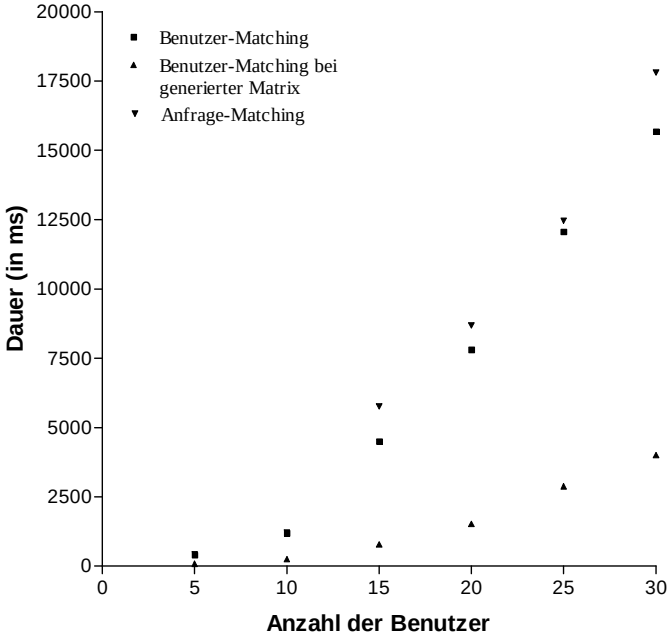


Abbildung 28: Performance des Matchings

5.3 Fazit

Im Verlauf der Evaluation konnte für den entwickelten Algorithmus zum Matchen von Benutzern auf Basis automatischer Textanalyse eine sinnvolle Konfiguration der verschiedenen Verfahrensparameter ermittelt werden. Im Bereich der automatischen Textanalyse hat sich für die Größe des Benutzerprofils eine Größenordnung von 500 Begriffen als sinnvoll erwiesen. Sollte die Textmenge deutlich über 100000 Wörtern liegen, sollte die Anzahl der Begriffe im Benutzerprofil gesteigert werden. Eine alternative Möglichkeit, die Größe des Begriffsvektors festzulegen, wäre die Angabe eines Schwellwertes bezüglich der Frequenz. Nach dieser Methode würden alle Begriffe, die die angegebene minimale Häufigkeit eines Begriffs im Text überschreiten, in den Begriffsvektor aufgenommen. Diese Variante hat jedoch den Nachteil, dass bei großer Textmenge das Benutzerprofil sehr groß werden kann. In diesem Fall würde die Performanz stark darunter leiden. Als sinnvolle Relation zwischen Schlüsselwörtern und Kollokationen hat sich ein Wert von 0,9 erwiesen. Das entspricht einem 90%-igen Anteil an Schlüsselwörtern, während die übrigen 10% aus Kollokationen bestehen.

Bei der Wahl der Matching-Parameter ist die Wahl der Gewichtung zugunsten von TF-IDF ausgefallen. Der Grund hierfür liegt in der stärkeren Gewichtung von Begriffen, die in nur wenigen Dokumenten enthalten sind, was darauf hindeutet, dass es sich bei dem Wort um einen Fachbegriff handelt. Als optimale Zahl der LSI-Dimensionen hat sich im Zuge der Evaluation ein Wert von drei ergeben. Der relativ kleine Wert dieser Größe resultiert mutmaßlich aus dem relativ ähnlichen Fachgebiet aller Benutzer dieser Gruppe Dieser Wert ist jedoch nicht auf andere Benutzergruppen übertragbar und muss im Einzelfall entschieden werden, da er sehr von der Gruppenkonstellation abhängt. Bei größeren Benutzerzahlen schlägt [Berr95] eine Größenordnung zwischen 70 und 100 Dimensionen vor.

Die Evaluation hat weiterhin ergeben, dass das entwickelte Verfahren bezüglich der Ähnlichkeit zu den beiden Referenzen etwa die gleiche Größenordnung hat, wie die Selbsteinschätzung und die Expertenempfehlung untereinander. In Abschnitt 5.2.2 wurden einige Einflussfaktoren genannt, die – unter der Annahme, dass das Verfahren prinzipiell gut arbeitet – Abweichungen zwischen den Verfahrensergebnissen und den Referenz-Einschätzungen erklären.

Die Analyse der Laufzeiten hat ergeben, dass die automatische Analyse der Texte bei großen Textmengen sehr zeitintensiv werden kann. Dieser Bereich des Verfahrens ist jedoch nicht sehr zeitkritisch, da die Dokumente nur einmal analysiert werden müssen.

Wesentlich interessanter ist die Performanz der Matching-Funktionen, die in der Praxis am häufigsten benutzt werden und daher ein gutes Laufzeitverhalten aufweisen sollten. Was das Matchen von verschiedenen Benutzerprofilen angeht, konnte die Laufzeit durch eine einmalige Generierung der Term-Benutzer-Matrix bei Systemstart die Laufzeit um ein vielfaches verringert werden. Das Matching von Anfragen, das nicht vorrangiges Ziel dieser Arbeit war, wird aufbauend auf den Ergebnissen dieser Arbeit, zurzeit optimiert.

6 Zusammenfassung und Diskussion

Resultierend aus dem immer größer werdenden Bedarf an Hilfsmitteln zur Erlangung von Wissen, nimmt die wissenschaftliche Disziplin Wissensmanagement einen zunehmend größeren Stellenwert in der Forschung ein. Eine Möglichkeit verschiedene Personen anhand ihres Wissens zu vernetzen bieten Recommender Systeme, die Personen auf Basis ihrer Fähigkeiten an andere Personen vermitteln. Um die Qualität dieser Systeme zu gewährleisten, ist die Analyse aller verfügbaren Informationen über die in das System involvierten Personen erforderlich. Eine Informationsquelle für Recommender Systeme bietet die Textproduktion jeder Person, da die Vermutung nahe liegt, dass eine Person nur über ein Thema schreibt, wenn sie bezüglich der Thematik über Kompetenzen oder zumindest Interessen verfügt.

In der vorliegenden Arbeit wurde ein Verfahren für ein Benutzer-Matching auf Basis der Textproduktion von Personen entwickelt. Der Arbeit liegt die Zielsetzung zugrunde, das Verfahren bei akzeptabler Qualität als Modul in das Recommender System „ExpertFinder Framework“ zu integrieren und dessen Ergebnisse zu optimieren.

Das Verfahren besteht aus zwei Stufen – Generierung der Benutzerprofile und Benutzer-Matching anhand dieser Profile. In der ersten Stufe werden aus einem Text mittels statistischer und linguistischer Verfahren Benutzerprofile in Form gewichteter Listen von Stichwörtern generiert. Im zweiten Schritt werden die verschiedenen Benutzerprofile mit dem vektorbasierten Verfahren Latent Semantic Indexing (LSI) auf ihre Ähnlichkeit hin verglichen. Daraufhin werden einem Benutzer auf Basis der Ergebnisse von LSI verschiedene andere Benutzer als „Experten“ vorgeschlagen, die zu einer Anfrage bzw. dem eigenen Benutzerprofil passen. Das konzipierte Verfahren wurde in prototypischer Form implementiert, wobei sich der Prototyp in drei Teilbereiche gliedert, die als Java-Packages implementiert sind. Das erste Paket leistet die automatische Analyse der Texte und erzeugt für einen Benutzer ein Profil seiner Interessen bzw. Kompetenzen. Im zweiten Paket ist die persistente Speicherung der Profile in einer MySQL-Datenbank implementiert. Der letzte Teil des Prototyps implementiert das Matching, dessen Ergebnisse als Basis für die Benutzer-Empfehlungen dienen.

Die Evaluation des Prototyps hat einige Aufschlüsse über die Verwendbarkeit des Verfahrens sowie eine sinnvolle Konfiguration der Verfahrens-Parameter geliefert. Die Evaluation hat ergeben, dass das entwickelte Verfahren ein relativ gutes „Interessen- und Kompetenz-Profil“ einer Person bildet und aus diesem Grund für die Integration in das Ex-

pertFinder Framework geeignet ist. Das Verfahren bietet weiterhin ein unvoreingenommenes Bild eines Benutzers, welches nicht von persönlichen Eindrücken geprägt ist. Bezüglich der Performance hat sich gezeigt, dass der Textanalyseteil am zeitintensivsten ist. Da die Textanalyse aber ein einmaliger Vorgang ist, kann man ihn als zeitunkritisch bezeichnen. Im Bereich des Matchings konnte das Benutzer-Matching auf eine zufriedenstellende Laufzeit reduziert werden. Einzig das Anfrage-Matching ist bezüglich der Performance noch verbesserungswürdig.

Im Verlauf der Testphase wurden einige Charakteristiken des Verfahrens deutlich, welche möglicherweise die Ergebnisse des Verfahrens in negativer Weise beeinflussen könnten. Ein wesentliches Kriterium für die Qualität der Ergebnisse ist die Menge des zu analysierenden Textes. Wenn die Textmenge sehr klein ist, werden die Ergebnisse ungenau. Die Beobachtung dieses Phänomens zeichnet nahezu alle statistischen Verfahren aus und wurde in Abschnitt 2.2.6 angesprochen.

Ein weiteres Problem tritt bei der Behandlung von mehrsprachigen Texten auf. Das Verfahren ist nicht in der Lage inhaltlich gleiche Begriffe verschiedener Sprachen als Synonyme zu erkennen. Dieser Zustand stellt insbesondere in Deutschland, wo wissenschaftliche Arbeiten sowohl in deutscher als auch in englischer Sprache verfasst werden ein Problem dar. So könnten im Extremfall zwei inhaltlich nahezu identische Texte, ein deutscher und ein englischer, einen sehr geringen Ähnlichkeitswert annehmen. Ein weiteres Problem, das mit mehrsprachigen Texten zu tun hat, ist die Tatsache, dass Stemmer sprachabhängig sind. Ein Stemmer für die deutsche Sprache würde, auf einen englischen Text angewandt, absolut unbrauchbare und nicht mehr erkennbare Ergebnisse liefern. Wenn ein Text aus verschiedensprachigen Teilen besteht, kann kein Stemmer auf diesen Text angewendet werden. Eine Lösung für dieses Problem ist die separate Analyse verschiedener Sprachen.

Zum Abschluss dieser Arbeit sollen einige Möglichkeiten aufgezeigt werden, das entwickelte Verfahren noch zu verbessern:

Optimierung des Benutzerprofils

Eine Möglichkeit, die Qualität des Verfahrens zu erhöhen, besteht in der Verbesserung der Benutzerprofile. Im Verlauf der Evaluation hat sich gezeigt, dass zum einen gemischtsprachliche Benutzerprofile zu einer Minderung der Ergebnisgüte führen können. Zum anderen hat sich gezeigt, dass bei Benutzerprofilen, die aus sehr großen Textmengen generiert wurden, die Kompetenzen relativ weit gestreut sind. Dieser Zustand kann dazu

führen, dass die einzelnen Kernkompetenzen solcher Personen etwas unscharf werden, was zu einer Minderung der Ergebnisqualität führen kann.

Eine Möglichkeit, um diese Problematik zu umgehen, ist die Erstellung von mehreren Benutzerprofilen für jede Person. Beispielsweise könnte für jede Sprache, in der eine Person Texte verfasst hat, ein eigenes Benutzerprofil erstellt werden. Weiterhin wäre es denkbar, verschiedene fachliche Schwerpunkte einer Person durch jeweils ein eigenes Profil abzubilden. Beide Fälle bringen höheren Aufwand für die Benutzer mit sich, da eine Sortierung der Dokumente im Vorfeld des Verfahrens notwendig ist.

Stemming von Kollokationen

Im Rahmen der Kollokationsanalyse besteht die Möglichkeit, die gefundenen Kollokationen auf ihre Wortstämme zu reduzieren. Die Schwierigkeit hierbei besteht darin, dass bei Kollokationen immer nur das zweite Wort je nach Fall verändert wird. Das erste Wort ist in jedem Fall gleich und sollte also nicht gestemmt werden. Aus diesem Grund ist es nicht möglich, die schon implementierten Stemmer zu verwenden. Es müsste also ein neuer Algorithmus entwickelt werden, der Kollokationen als solche erkennt und entsprechend den Wortstamm des zweiten Wortes stemmt.

Wörterbuchbasierte Wortstammreduktion

Das Ergebnis der Textanalyse könnte qualitativ verbessert werden, wenn man zur Reduktion der Wortstämme statt dem Stemmer ein wörterbuchbasiertes Verfahren benutzt. Diese Verbesserung resultiert aus der Tatsache, dass der Stemmer fehleranfällig ist und bei manchen Wörtern den falschen Wortstamm bildet. Ein auf einem Wörterbuch basierendes Verfahren würde bei guter Qualität des Wörterbuchs wahrscheinlich zu besseren Ergebnissen führen. Problematisch an diesem Verfahren ist der hohe Ressourcenverbrauch. Die Zeit der Analyse würde sich wesentlich verlängern. Außerdem würde zusätzlicher Aufwand entstehen, da Wörterbücher erst besorgt bzw. erzeugt werden müssten.

Temporale Reduktion der Gewichtungen

Um auch den natürlichen Prozess des Vergessens in dem Verfahren zu integrieren, ist es vorstellbar, ältere Texte geringer zu gewichten als gerade erst bearbeitete oder geschriebene. Man könnte so der Tatsache Rechnung tragen, dass jeder Mensch vergesslich ist und über ein Thema, welches er gerade bearbeitet mehr Kenntnisse hat als über eine zeitlich weiter zurückliegende Problematik. Der Prozess des Vergessens könnte in Form einer Reduktion der Gewichtungen in bestimmten zeitlichen Intervallen modelliert werden. Es könnten beispielsweise alle zwei Wochen die Gewichtungen einer Person um 5% reduziert werden. Um dieses Vorgehen nutzbar zu machen ist es jedoch notwendig, dass die

Benutzerprofile auch inhaltlich regelmäßig aktualisiert werden, d.h. jedes neue Dokument eines Benutzers muss sofort in das Benutzerprofil integriert werden. Bisher sind bei der Aktualisierung der Benutzerprofile eher größere zeitliche Intervalle angedacht.

Berücksichtigung von Strukturinformationen

Eine Möglichkeit die Gewichtungen im Benutzerprofil präziser zu machen ist die Berücksichtigung von Strukturinformationen. Unter Strukturinformationen versteht man besondere Markierungen innerhalb von Texten wie beispielsweise Überschriften oder Unterstreichungen. Diese Informationen müssten während der Filterung von ASCII-Text aus doc- oder pdf-Dokumenten gesammelt werden, damit sie später im Benutzerprofil berücksichtigt werden können. Im Einzelnen würde das dann bedeuten, dass ein Wort, wenn es fett geschrieben ist nicht mit eins sondern vielleicht mit drei gewichtet wird. Ein Wort, das als Überschrift formatiert ist, könnte mit 10 gewichtet werden. Da das entwickelte Verfahren mit unformatiertem Text als Eingabe arbeitet, also alle Formatierungen entfernt sind, müsste dieses Verfahren schon vorgelagert arbeiten.

Anhang A Verwendete Stoppwortlisten

Stoppwortliste für die deutsche Sprache:

ab aber ach acht alle allein allem allen aller allerdings alles allgemein als also alt am an andere anderem anderen anderer andererseits anderes anderm andern anders anhand arbeiten auch auf aufweisen aufweisende aufweisenden aus ausser außer ausserdem außerdem

bald beginnen bei beide beidem beiden beides beim beispielsweise bekannt bekennen bereits berichten besonders besser bestehen bestimmt bestimmte bestimmtem bestimmten bestimmter bestimmtes betonen betonte bevor bezüglich bezueglich bin bis bißchen bisher bisschen bist bleiben bringen bzw

da dabei dadurch dafür dafuer dagegen daher dahin dahinter damals damit danach daneben dank dann daran darauf daraus darin darüber darum darunter das daß dass dasselbe davon davor dazu dazwischen dein deine deinem deinen deiner deines dem demselben den denselben denen denn der derart deren derer derselbe derselben des desselben dessen deutlich dich die dies diese dieselbe dieselben diesem diesen dieser dieses dir doch dort drei du durch dürfen

eben ebenfalls ebenso eigen eigenen ein eine einem einen einer einerseits eines einig einige einigen einiges einmal einzeln einzelne einzelnem einzelnen einzelner einzeln es entlang entscheiden entsprechen entsprechend entsprechende entsprechenden entsprechendem entsprechender entsprechendes entweder er erhalten erklären erklärte erst ersten es etwa etwas euch euer eure eurem euren eurer eures

falls fast fest finden fordern fragen frei früh frueh führen fuehren fünf fuenf für fuer fürs fuers

ganz gar gebe geben gegebenenfalls gegen gegenüber gegenueber gehen gehören geht gekennzeichnet gemäss gemaess gemeinsam genau gewesen ggf gibt glauben gleich groß großen gründen gut

habe haben handeln hat hatte hätte hatten hätten heilig heißt her herein herum heute hier hin hinter hintern hoch hören http

ich ihm ihn ihnen ihr ihre ihrem ihren ihrer ihres im immer in indem infolge ins insbesondere insgesamt international ist

ja je jede jedem jeden jeder jedes jedesmal jedoch jene jenem jenen jener jenes jetzt jung

kann kaum kein keine keinem keinen keiner keines kirchlich klein kommen könne können könnte könnten kritisieren

lang laß lass lassen leben letzen letzte letzten

machen man manche manchem manchen mancher manches mehr mehrere mehreren mehrerer mein meine meinem meinen meiner meines meist mich mir mit mitteilen mittels möglich muß muss müsse müssen musste müßten müssten

nach nachdem nacheinander nah nämlich national neben nehmen nein nennen neu neue neuen nicht nichts noch nun nur

ob ober obgleich oder ohne

paar pro

recht reich religiös rund

sagte schaffen schließlich schliesslich schon schreiben schwer sehen sehr sei seien sein seine seinem seinen seiner seines seit seitdem selbst setzen sich sie sind so sobald sodass sodaß sofern sofort sogar solange solch solche solchem solchen solcher solches soll sollen sollte sollten somit sondern sonst soviel soweit sowie sowohl spät sprechen stark statt stehen steht stellen

teilen teils teilte teilweise

über ueber um und uns unser unsere unserem unseren unserer unseres unter unterhalb usw

vergangen vergangenen vergehen veröffentlichen viel viele vier voll vom von vor vorher vorsitzen

währen waehren während waehrend war wär wäre waere waren wären waeren warum was weg wegen weil weit weiter weiterhin weitgehend welche welchem welchen welcher welches wem wen wenig wenige wenigstens wenn wer werde werden weshalb wessen wichtig wie wieder will wir wird wirst wo wobei wodurch wollen womit worden worauf worin wurde wurden würden wuerden www

zehn zeigen zentral zu zueinander zugleich zum zumindest zunächst zunaechst zur zusammen zusätzlich zusaetzlich zwar zwecks zwei zweit zwischen zwischens

Stoppwortliste für die englische Sprache:
a able about above according accordingly across actually after afterwards again against ago ain't all allow allows almost alone along already also although always am among amongst an and another any anybody anyhow anyone anything anyway anyways anywhere apart appear appreciate appropriate are aren't around as a's aside ask asking associated at available away awfully

back be became because become becomes becoming been before beforehand behind being believe below beside besides best better between beyond both brief but by

came can cannot cant can't cause causes certain certainly changes clearly c'mon co com come comes concerning consequently consider considering contain containing contains corresponding could couldn't course c's currently

definitely described despite did didn't different do does doesn't doing done don't down downwards during

e each edu eg eight either else elsewhere enough entirely especially et etc even ever every everybody everyone everything everywhere ex exactly example except

far few fifth first five followed following follows for former formerly forth four from front further furthermore

get gets getting given gives go goes going gone got gotten greetings

had hadn't happens hardly has hasn't have haven't having he hello help hence her here hereafter hereby herein here's hereupon hers herself he's hi him himself his hither hopefully how howbeit however

i i'd ie if ignored i'll i'm immediate in inasmuch inc indeed indicate indicated indicates inner insofar instead into inward is isn't it it'd it'll its it's itself i've

just

keep keeps kept know known knows

last lately later latter latterly least left less lest let let's like liked likely little look looking looks ltd

mainly make many may maybe me mean meanwhile merely might mine more moreover most mostly much must my myself

name namely near nearly necessary need needs neither never nevertheless new next nine no nobody non none noone nor normally not nothing novel now nowhere

obviously of off often oh ok okay old on once one ones only onto or other others otherwise ought our ours ourselves out outside over overall own

particular particularly per perhaps placed please plus possible presumably probably provides put putting

que quite

rather rd re really reasonably regarding regardless regards relatively respectively right

said same saw say saying says second secondly see seeing seem seemed seeming seems seen self selves sensible sent serious seriously seven several shall she should shouldn't since six so some somebody somehow someone something sometime sometimes somewhat somewhere soon sorry specified specify specifying stand still sub such sup sure

take taken tell tends th than thank thanks thanx that thats that's the their where theirs them themselves then thence there thereafter thereby therefore therein theres there's thereupon these they they'd they'll they're they've think third this thorough thoroughly those though three through throughout thru thus till to together too took toward towards tried tries truly try trying twice two

un under unfortunately unless unlikely until unto up upon us use used useful uses using usually uucp

value various very via viz vs

want wants was wasn't way we we'd welcome well we'll went were we're weren't we've what whatever what's when whence whenever where whereafter whereas whereby wherein where's whereupon wherever whether which while whither who whoever whole whom who's whose why will willing wish with within without wonder won't would wouldn't

yes yet you you'd you'll your you're yours yourself yourselves you've

zero

Stoppwortliste für die französische Sprache:
alors au aucuns aussi autre avant avec avoir

bon

ça car ce cela ces ceux chaque ci comme comment

dans début dedans dehors depuis des deux devrait doit donc dos droite du

elle elles en encore essai est et étaient état été étions être eu

fait faites fois font force

haut hors

ici il ils

je juste

la là le les leur

ma maintenant mais même mes mine moins mon mot

ni nommés notre nous nouveaux

ou où

par parce parole pas personnes peu peut pièce plupart pour pourquoi

quand que quel quelle quelles quels qui

sa sans ses seulement si sien son sont sous soyez sujet sur

ta tandis tellement tels tes ton tous tout très trop tu

valeur voie voient vont votre vous vu

Anhang B Benutzerschnittstelle des Prototyps

Für den Prototyp wurde zusätzlich eine graphische Benutzerschnittstelle implementiert, in welcher die drei Verfahrensteile integriert sind. Abbildung 29 zeigt den ersten Schritt des Benutzers, die Auswahl einer zu analysierenden Textdatei.

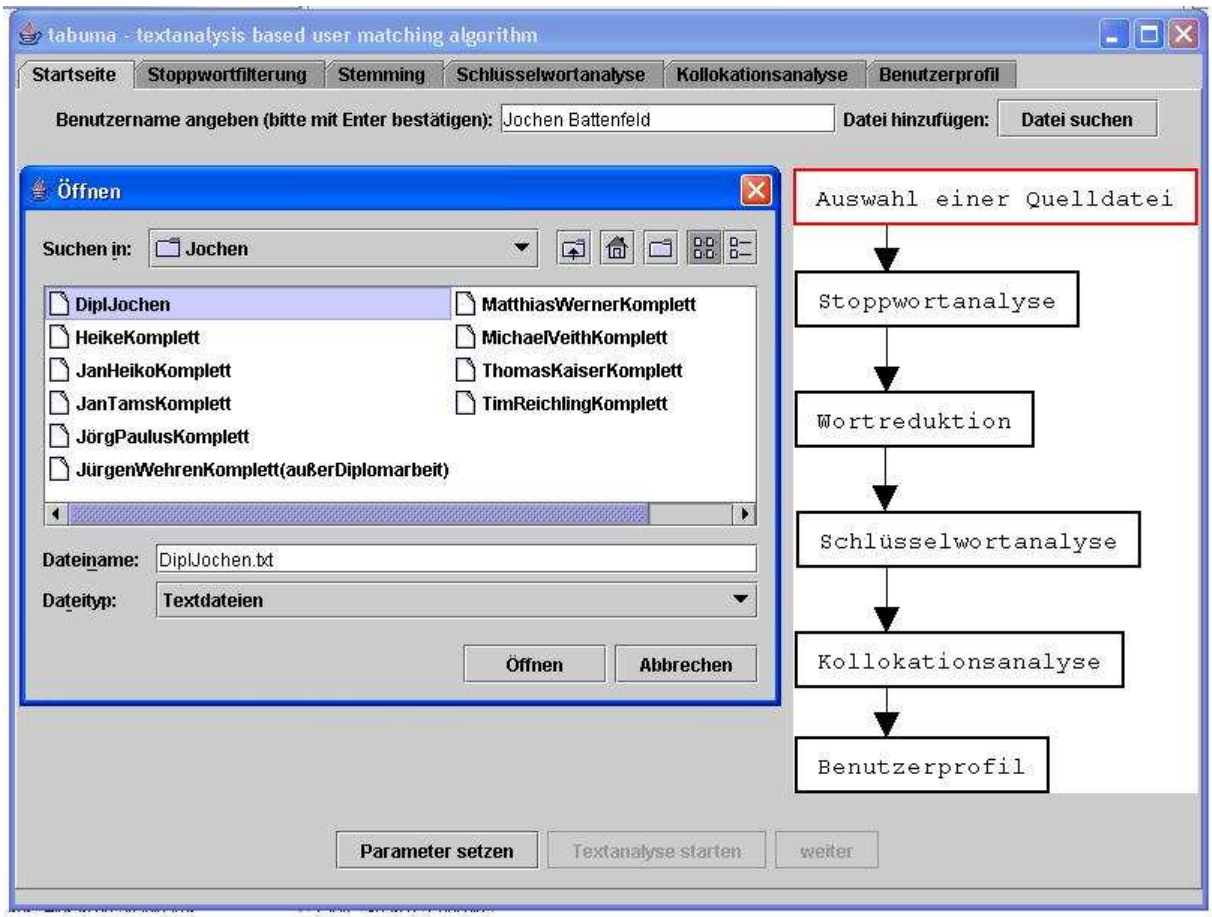


Abbildung 29: Startseite mit „Datei öffnen“-Dialog

Man sieht in der Abbildung, dass die Benutzerschnittstelle in Form von Registern angeordnet ist. Der Benutzer muss anfangs seinen Namen angeben, damit das später generierte Benutzerprofil zugeordnet werden kann. Danach kann man mit dem Knopf „Datei suchen“ den „Datei öffnen“-Dialog starten und eine Textdatei zur Analyse auswählen. Wenn eine Datei ausgewählt ist wird der Knopf Textanalyse starten frei geschaltet und der Benutzer kann eine automatische Textanalyse ausführen. Es besteht für den Benutzer die Möglichkeit, die Größe des Benutzerprofils sowie das Verhältnis zwischen Schlüsselwörtern und Kollokationen zu wählen. Hierzu muss er den Knopf „Parameter setzen“ betätigen. Er erhält daraufhin den in Abbildung 30 dargestellten Dialog. Durch Betätigung des Knopfes „Parameter setzen“ innerhalb des Dialogs werden die gewählten Parameter gespeichert.

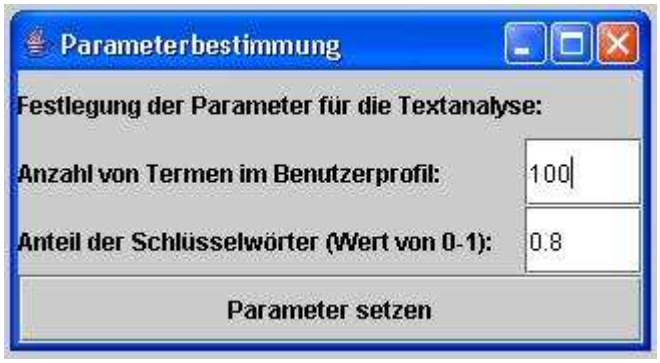


Abbildung 30: "Parameter setzen"-Dialog in der Textanalyse

Nachdem die Textanalyse durchgeführt wurde kann man sich die Zwischenergebnisse auf den Registerseiten „Stoppwortfilterung“, „Stemming“, „Schlüsselwortanalyse“ und „Kollokationsanalyse“ ansehen. Die Navigation erfolgt entweder über die Registerköpfe selbst oder alternativ über „weiter“- und „zurück“-Knöpfe.

Das Benutzerprofil in Form einer Liste von Schlüsselwörtern und Kollokationen wird auf der Registerseite „Benutzerprofil“ ausgegeben (siehe Abbildung 31).

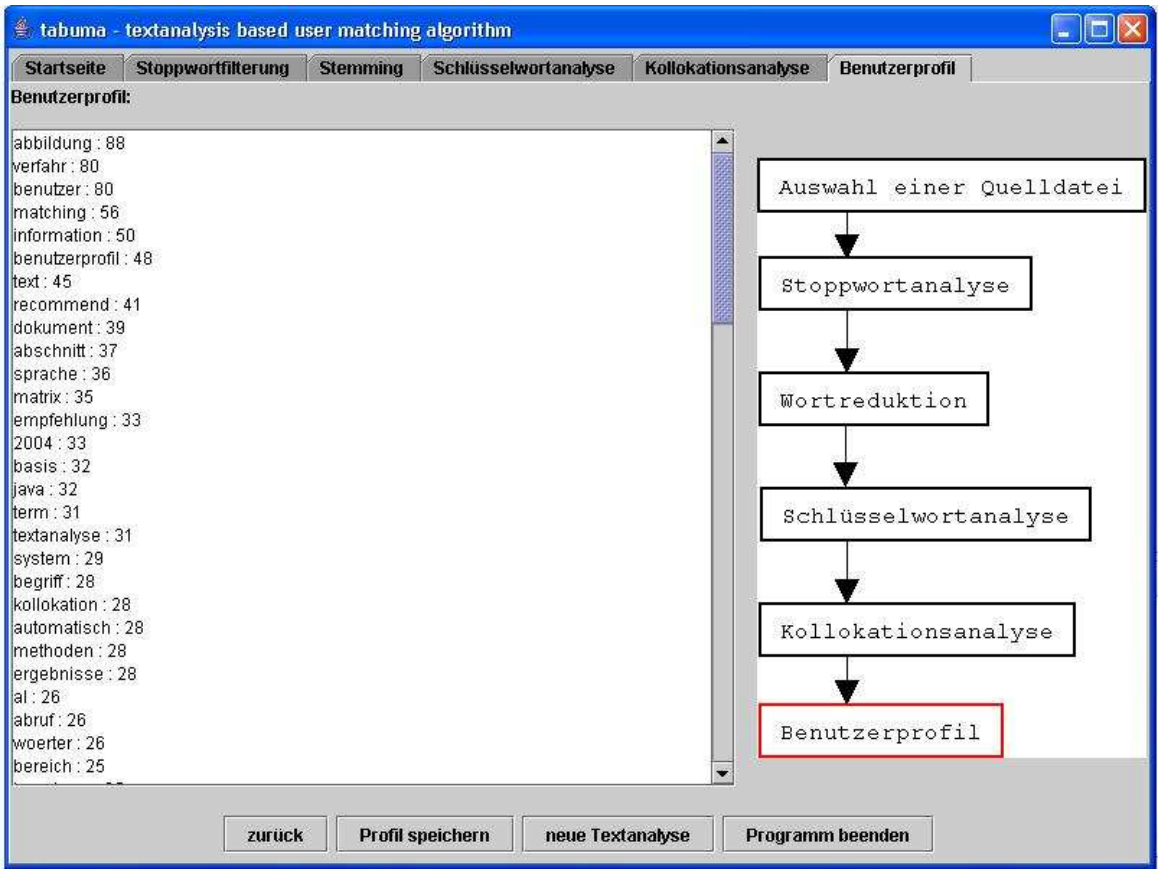


Abbildung 31: Darstellung des Benutzerprofils

Für den Benutzer besteht nun die Möglichkeit, das generierte Profil über den „Profil speichern“-Knopf zu speichern. Nach der Speicherung folgt ein Dialog, der dem Benutzer die

Möglichkeit anbietet, eine Anfrage oder ein Benutzer-Matching durchzuführen. Alternativ kann er entweder eine neue Textanalyse durchführen oder das Programm beenden.
Im zweiten Teil des Verfahrens, dem Matching, hat der Benutzer die Möglichkeit, entweder Empfehlungen zu einer Anfrage zu erfragen oder zu seinem eigenen Profil ähnliche Benutzer zu ermitteln. Für den Benutzer ist immer nur die ausgewählte Funktion sichtbar, also entweder die Anfragefunktion oder die Benutzer-Matchingfunktion. Die jeweils andere Funktion wird nach der Auswahl einer Option ausgeblendet. In Abbildung 32 ist die graphische Umsetzung des Matchings einer Anfrage dargestellt. Die Anfrage wird vom Benutzer in das vorgegebene Textfeld eingegeben und der Knopf „Matching durchführen“ betätigt.

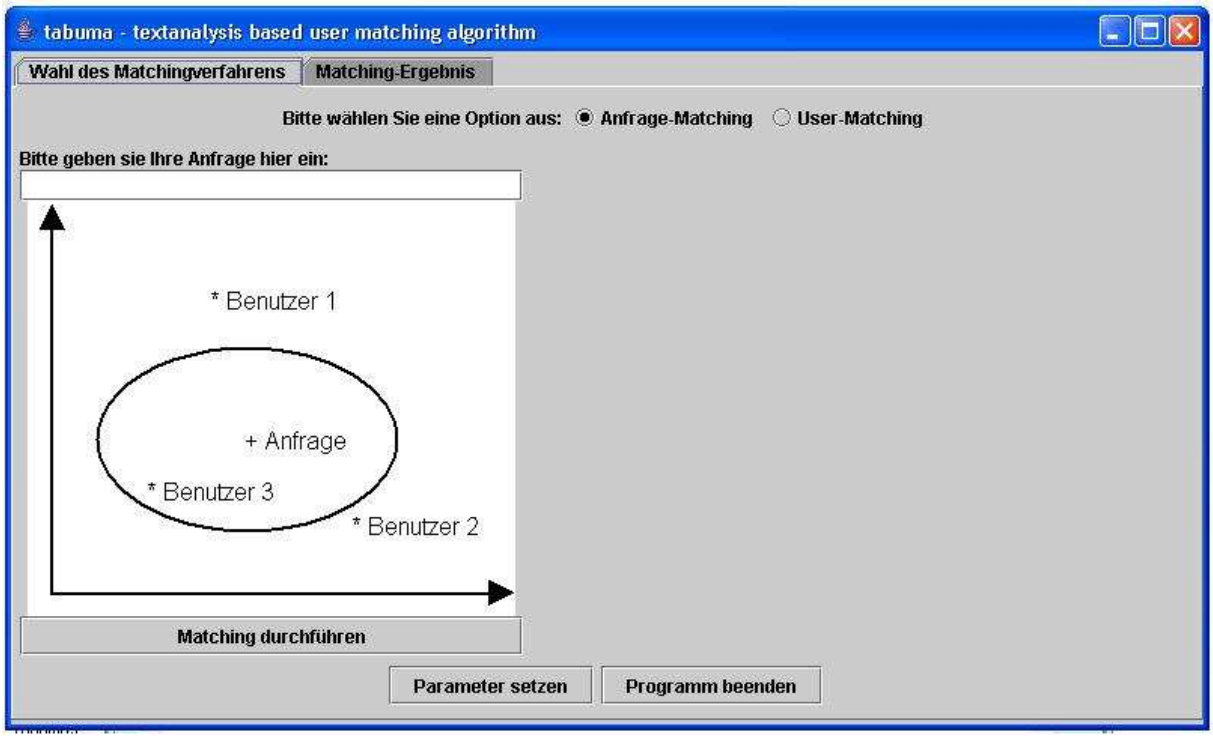


Abbildung 32: Anfrage-Matching

In Abbildung 33 ist die andere Option, das Matching von verschiedenen Benutzern, dargestellt. In einer Combobox sind alle in der Datenbank gespeicherten Benutzer ausgeführt. Aus diesen Benutzern kann nun einer, in der Regel man selbst, ausgewählt werden. Das Profil des ausgewählten Benutzers wird daraufhin mit allen anderen gespeicherten Benutzerprofilen gematcht und die ähnlichsten Benutzer als Empfehlung zurückgegeben. Das Ergebnis ist also eine Liste von Empfehlungen, die dem Benutzer angeboten werden. Jede Empfehlung besteht aus einem Benutzernamen und dem dazu gehörigen Ähnlichkeitswert zwischen 0 und 1.

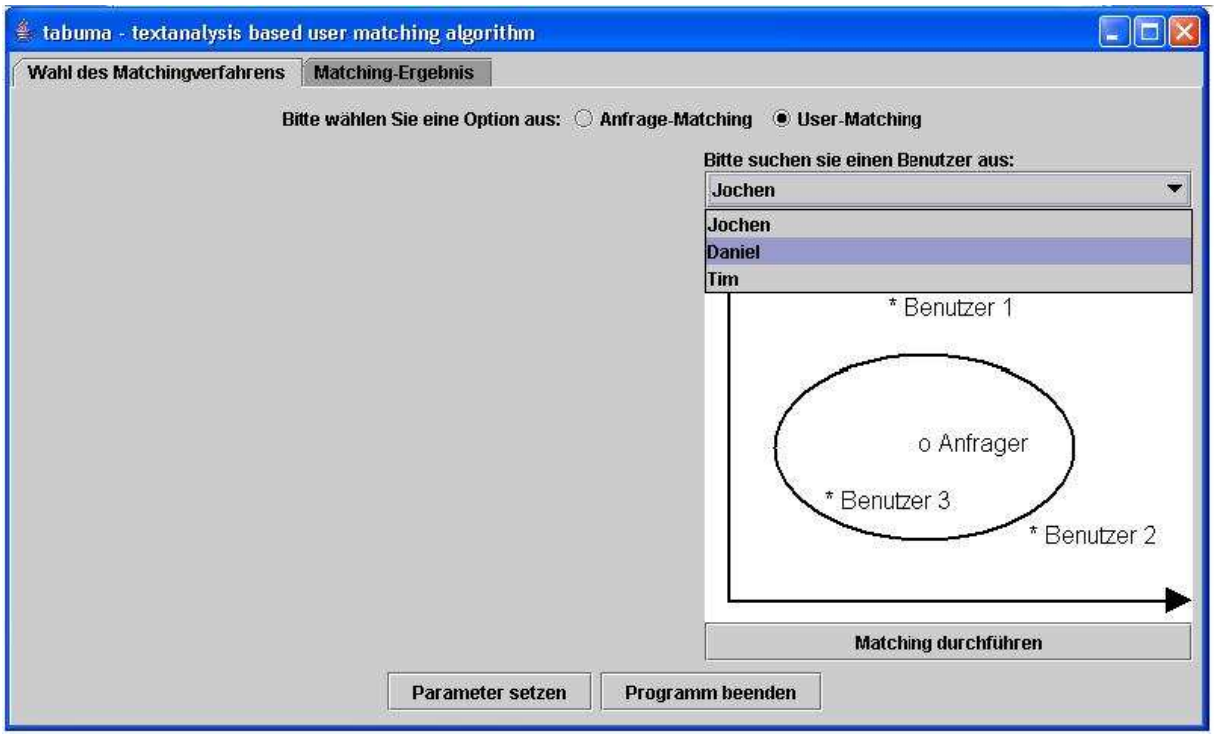


Abbildung 33: Benutzer-Matching

Bei beiden Arten des Matchings besteht die Möglichkeit die Matching-Parameter selbst zu konfigurieren. Bei Betätigung des Knopfes „Paramater setzen“ wird der in Abbildung 34 dargestellte Dialog geöffnet.

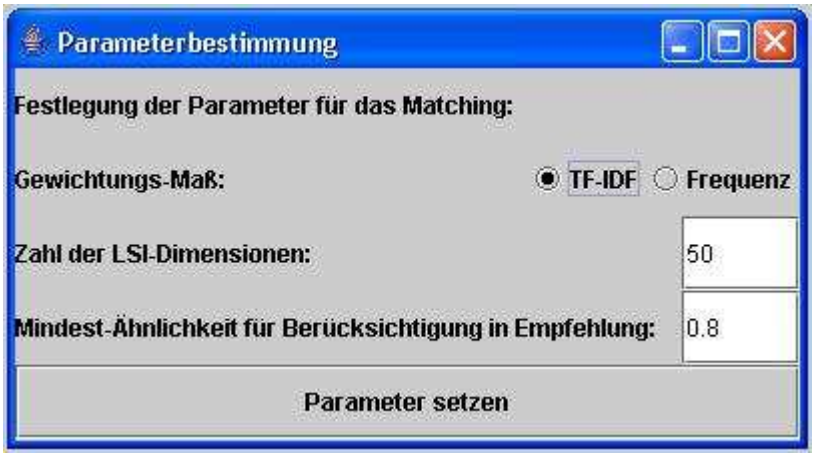


Abbildung 34: "Parameter setzen"-Dialog innerhalb des Matching-Teils

Die manipulierbaren Parameter sind erstens das Gewichtungsmaß, das entweder mit der Termfrequenz oder mit TFIDF angegeben werden kann, zweitens die Zahl der LSI-Dimensionen, d.h. die Zahl der durch LSI entstehenden neuen künstlichen Dimensionen und drittens die minimale Ähnlichkeit eines Benutzerprofils um im Ergebnis aufzutauchen. Durch den Knopf „Parameter setzen“ werden die Parameter aktualisiert.

Das Ergebnis des Anfrage-Matchings ist analog zum Matching von Benutzern eine Empfehlung in Form einer Liste von Benutzern mit ihren Ähnlichkeitswerden.

Anhang C Untersuchungen zur Ergebnisgüte

		Andreas Becks	Helge Kahler	Ralf Klamma	Bernhard Nett	Reinhard Oppermann	Volkmar Pipek	Tim Reichling	Markus Rohde	Gunnar Stevens	Markus Won
Position innerhalb der Empfehlung:	Andreas Becks	X	15	17	9	4	-	1	13	13	17
	Björn Golombek	-	8	3	2	16	14	14	14	7	2
	Joachim Hinrichs	10	10	4	11	13	8	5	15	12	11
	Britta Hofmann	5	13	5	3	3	15	9	5	11	9
	Helge Kahler	-	X	14	10	11	3	13	2	5	4
	Ralf Klamma	4	14	X	5	7	7	3	3	14	-
	Bernhard Nett	9	4	6	X	6	10	10	4	6	10
	Reinhard Oppermann	8	9	10	6	X	5	12	12	8	-
	Andreas Pfeifer	-	7	8	0	14	12	15	10	15	5
	Volkmar Pipek	-	1	13	14	15	X	7	6	2	3
	Tim Reichling	2	12	7	4	12	11	X	9	9	7
	Markus Rohde	6	6	12	13	5	9	8	X	10	1
	Barbara Schmidt-Belz	7	16	2	7	1	13	6	11	16	-
	Michael Spenke	1	17	1	1	9	-	16	17	17	-
	Gunnar Stevens	-	11	11	15	10	1	4	8	X	6
	Oliver Stiemerling	-	5	9	8	17	6	17	16	3	8
	Markus Won	-	3	15	12	8	4	11	7	4	X
	Volker Wulf	3	2	16	16	2	2	2	1	1	4

Tabelle 4: Selbsteinschätzung der Gruppe "Publikationen"

Experteneinschätzung ohne Kenntnis der Dokumente		Andreas Becks	Helge Kahler	Ralf Klamma	Bernhard Nett	Reinhard Oppermann	Volkmar Pipek	Tim Reichling	Markus Rohde	Gunnar Stevens	Markus Won
Position innerhalb der Empfehlung:	Andreas Becks	X	16	3	9	13	15	1	11	16	14
	Björn Golombek	16	3	17	15	8	10	17	12	6	6
	Joachim Hinrichs	4	11	10	3	17	14	7	5	10	15
	Britta Hofmann	15	17	16	14	7	16	16	13	14	16
	Helge Kahler	14	X	15	6	9	2	13	6	4	3
	Ralf Klamma	3	12	X	10	16	8	4	2	15	11
	Bernhard Nett	11	13	9	X	15	7	8	3	9	10
	Reinhard Oppermann	8	7	11	11	X	13	11	4	13	9
	Andreas Pfeifer	13	8	12	16	11	9	14	16	8	7
	Volkmar Pipek	5	1	2	4	10	X	3	7	5	4
	Tim Reichling	1	10	6	13	14	12	X	8	12	13
	Markus Rohde	10	9	4	2	3	6	9	X	7	8
	Barbara Schmidt-Belz	17	15	13	7	1	4	15	14	11	12
	Michael Spenke	9	14	7	17	4	17	6	17	17	17
	Gunnar Stevens	6	4	8	5	5	5	10	10	X	5
	Oliver Stiemerling	12	5	14	12	12	11	12	15	1	1
	Markus Won	7	2	5	8	6	1	5	9	2	X
	Volker Wulf	2	6	1	1	2	3	2	1	3	2

Tabelle 5: Experteneinschätzung der Gruppe "Publikationen"

Experteneinschätzung mit Kenntnis der Dokumente		Andreas Becks	Helge Kahler	Ralf Klamma	Bernhard Nett	Reinhard Oppermann	Volkmar Pipek	Tim Reichling	Markus Rohde	Gunnar Stevens	Markus Won
Position innerhalb der Empfehlung:	Andreas Becks	X	16	3	17	3	18	2	17	17	16
	Björn Golombek	16	4	17	15	5	11	17	14	8	7
	Joachim Hinrichs	6	15	10	5	16	1	4	9	11	14
	Britta Hofmann	15	3	5	4	2	14	6	10	10	8
	Helge Kahler	17	X	15	9	6	10	16	4	7	6
	Ralf Klamma	1	14	X	14	17	16	3	7	15	15
	Bernhard Nett	8	12	4	X	8	3	14	2	4	9
	Reinhard Oppermann	14	13	7	13	X	15	7	8	13	11
	Andreas Pfeifer	10	7	16	7	14	7	10	12	9	4
	Volkmar Pipek	5	6	9	6	13	X	5	3	6	5
	Tim Reichling	4	11	6	11	15	13	X	15	14	13
	Markus Rohde	7	10	2	3	7	8	11	X	5	10
	Barbara Schmidt-Belz	13	9	12	12	1	12	15	13	12	12
	Michael Spenke	2	17	8	16	4	17	8	16	16	17
	Gunnar Stevens	12	8	11	1	12	2	9	2	X	2
	Oliver Stiemerling	11	1	14	8	10	5	12	5	1	1
	Markus Won	9	5	13	16	11	6	13	11	2	X
	Volker Wulf	3	2	1	2	9	4	1	1	3	3

Tabelle 6: Experteneinschätzung der Gruppe "Publikationen" mit Kentnis der Dokumente

	Unähnlichkeit Verahren-SE						Unähnlichkeit Verahren-EE1						Unähnlichkeit Verahren-EE2					
Benutzer/Dimension	16	13	10	7	5	3	16	13	10	7	5	3	16	13	10	7	5	3
Andreas Becks	72	108	62	74	58	48	80	73	62	92	100	80	72	76	70	84	98	73
Helge Kahler	94	99	98	84	92	78	86	100	78	76	80	68	94	82	68	56	64	64
Ralf Klamma	106	90	88	94	82	84	98	102	122	80	78	86	108	84	102	56	62	56
Bernhard Nett	92	94	92	116	100	101	104	114	106	98	98	76	104	110	94	74	80	102
Reinhard Oppermann	106	110	120	85	82	66	84	99	120	96	88	120	86	104	104	78	74	86
Volkmar Pipek	65	98	110	78	66	78	74	96	116	86	80	81	78	82	98	54	48	72
Tim Reichling	100	82	54	110	102	70	80	76	80	90	106	78	98	78	60	100	100	72
Markus Rohde	74	86	100	92	96	82	112	110	96	96	104	78	114	126	104	98	98	74
Gunnar Stevens	94	98	72	78	66	68	96	84	84	78	60	50	98	90	84	62	52	72
Markus Won	106	92	97	84	84	62	74	56	73	90	84	66	96	80	73	80	72	50
Durchschnitt	91	96	89	90	83	74	89	91	94	88	88	78	95	91	86	74	75	72

Tabelle 7: Unähnlichkeiten der Gruppe "Publikationen"

Anhang D Ergebnisse der Performance-Analyse

Benutzer	Textanalyse (in ms)							
	1.Testlauf	2. Testlauf	3.Testlauf	4.Testlauf	5. Testlauf	Mittelwert	Wortzahl	Worte/ Sekunde
Jan Tams	344	422	407	422	375	394,00	1898	4817,26
Britta Schinzel	407	406	907	375	375	494,00	4498	9105,26
Britta Hofmann	516	531	516	516	563	528,40	4602	8709,31
J. H. Erik Andriessen	719	687	1375	687	718	837,20	5936	7090,30
Marike Hettinga	828	828	875	812	813	831,20	5936	7141,48
Markus Won	1407	1406	1594	1391	1469	1453,40	7420	5105,27
Helge Kahler	1203	1203	1187	1187	1343	1224,60	7886	6439,65
Gloria Mark	594	593	812	593	688	656,00	8133	12397,87
Joachim Hinrichs	1406	1438	1437	1390	1547	1443,60	8585	5946,94
Tim Reichling	2670	2218	2141	2172	2312	2302,60	9579	4160,08
Gunnar Stevens	6616	6704	12156	6890	6610	7795,20	16054	2059,47
Heike Werner	16842	15141	15047	14905	15530	15493,00	17494	1129,16
Björn Golombek	11375	11235	10938	11093	10953	11118,80	19667	1768,81
Barbara Schmidt-Belz	6656	6469	6594	6907	6484	6622,00	20619	3113,71
Michael Spenke	7609	7453	13656	7500	7484	8740,40	20730	2371,75
Andreas Pfeifer	20640	19375	18048	19140	19265	19293,60	22353	1158,57
Andreas Becks	22875	22141	11375	21985	21681	20011,40	22848	1141,75
Matthias Werner	27778	28047	28687	27545	29858	28383,00	23121	814,61
Tim Reichling	28434	28953	28453	28186	32311	29267,40	24486	836,63
Jan Heiko Lenz	42307	49156	43531	43029	47920	45188,60	25001	553,26
Thomas Kaiser	36855	38641	37078	37748	39232	37910,80	28848	760,94
Reinhard Oppermann	34202	33409	33812	33937	34547	33981,40	33954	999,19
Oliver Stiemerling	53068	52035	80828	51250	50609	57558,00	38680	672,02
Volkmar Pipek	75526	71393	69035	71063	70437	71490,80	39269	549,29
Michael Veith	138147	151265	139235	138758	137381	140957,20	41766	296,30
Jörg Paulus	100055	113171	105281	104604	98683	104358,80	47888	458,88
Juergen Wehren	154975	165156	156187	157492	155138	157789,60	49467	313,50
Bernhard Nett	127653	125492	127312	127750	126453	126932,00	50454	397,49
Ralf Klamma	336461	329555	331531	332703	351625	336375,00	75568	224,65
Markus Rohde	1280171	1249525	1297391	1240969	1220172	1257645,60	131512	104,57
Volker Wulf	1648993	1625698	1630938	1673844	1679078	1651710,20	197852	119,79

Tabelle 8: Performance der Textanalyse

User-Matching						
	Dauer (in ms)					
Zahl der Benutzer	1.Testlauf	2.Testlauf	3. Testlauf	4. Testlauf	5. Testlauf	Mittelwert
5	391	437	391	405	391	403,00
10	1282	1171	1188	1172	1172	1197,00
15	4547	4657	4454	4438	4438	4506,80
20	7860	7985	7703	7719	7781	7809,60
25	12234	11921	12032	12093	12047	12065,40
30	15828	15672	15641	15656	15609	15681,20
User-Matching ohne Matrixerzeugung						
5	110	78	94	78	93	90,60
10	250	250	266	250	250	253,20
15	750	875	766	781	766	787,60
20	1500	1531	1500	1531	1562	1524,80
25	2828	2906	2828	2813	3031	2881,20
30	4000	3968	4110	3984	3969	4006,20
Query-Matching						
5	438	406	422	453	453	434,4
10	1187	1188	1281	1203	1187	1209,2
15	5797	5860	5688	5906	5578	5765,8
20	8812	8625	8703	8656	8609	8681
25	12437	12469	12531	12391	12469	12459,4
30	17547	17578	17860	17984	18063	17806,4

Tabelle 9: Performance des Matchings

Literaturliste

[AaNy95]
Aamodt, Agnar; Nygard, Mads: Different roles and mutual dependencies of data, information and knowledge – an AI perspective on their integration.
In: Data and Knowledge Engineering 16, Elsevier 1995, S. 191-222.

[Acke03]
Ackerman, Mark S. et al.: Who's there? The Knowledge-Mapping Approximation Project.
In: Sharing Expertise beyond Knowledge Management, S. 159-178, MIT Press, Massachusetts 2003.

[Akad04]
akademie.de asp GmbH: Net-Lexikon.
http://www.net-Lexikon.de, Abruf am 18.11.2004.

[AlLa03]
Al-Laham, Andreas: Organisationales Wissensmanagement.
Verlag Franz Vahlen, München 2003.

[Back96]
Backhaus Klaus et al.: Multivariate Analysemethoden – Eine anwendungsorientierte Einführung.
Springer, Berlin 1996.

[Bahn96]
Bahns, Jens: Kollokationen als lexikographisches Problem.
Niemeyer, Tübingen 1996.

[BaSh97]
Balabanovic, Marko; Shoham, Yoav:Fab: Content-Based, Collaborative Recommendation.
In: Communications of the ACM, Bd. 40, Nr. 3, S. 66-72, März 1997.

[Beck01]
Becks, Andreas: Visual Knowledge Management with Adaptable Document Maps.
http://www.gmd.de/publications/research/2001/015/Text.pdf, Sankt Augustin 2001, Abruf am 28.11.2004.

[Beck03]
Becks, Andreas et al.: Supporting Collaborative Learning by Matching Human Actors. In: Sprague, Ralph H. jr.: Proceedings of the Thirty-Sixth Annual Hawaii International Conference on System Sciences (HICSS-36), Big Island, Hawaii 2003.

[Beck04]
Becks, Andreas et al.: Expert Finding: Approaches to Foster Social Capital.
In: *Huysman, Marleen; Wulf, Volker (Hrsg.)*: Social Capital and Information Technology, S. 333-354, Massachusetts 2004.

[BeKe03]
Beierle, Christoph; Kern-Isberner, Gabriele: Methoden wissensbasierter Systeme.
2. Aufl., Vieweg, Wiesbaden 2003.

[Beni78]
Benito, Dr. Francisco: Markov-Modelle mit indirekter Beobachtung.
Verlag Paul Haupt, Bern 1978.

[Bern01]
Berners-Lee, Tim et al.: The Semantic Web – A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. http://www.sciam.com/print_version.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21, Mai 2001, Abruf am 25.11.2004.

[Berr95]
Berry, Michael W. et al: Computational Methods for Intelligent Information Access.
In: Proceedings of the 1995 ACM/IEEE conference on Supercomputing, S. 20, San Diego, USA 1995.

[Bode03]
Bodendorf, Freimut: Daten- und Wissensmanagement.
Springer, Berlin Heidelberg 2003.

[BöKr99]
Böhmann, Thilo; Krcmar, Helmut: Werkzeuge für das Wissensmanagement.
In: Antony, Conny H.; Sommerlatte, Tom:Report Wissensmanagement – Wie deutsche Firmen ihr Wissen profitabel machen, Symposium Publishing, 1999.

[Buch91]
Buchholz, Peter: Die strukturierte Analyse Markovscher Modelle.
Springer-Verlag, Berlin Heidelberg 1991.

[Cars01]
Carstensen, Kai Uwe et al.(Hrsg.): Computerlinguistik und Sprachtechnologie – Eine Einführung.
Spektrum Akademischer Verlag, Heidelberg 2001.

[Deer90]
Deerwester, Scott et al.: Indexing by Latent Semantic Analysis.
In: Journal of the American Society of Information Science, Vol. 41, No. 6, S. 391-407, 1990.

[Endr98]
Endres-Niggemeyer, Brigitte: Summarizing Information – Cognitive Strategies.
Springer, Berlin Heidelberg 1998.

[Elst01]
Elst,Ludger van et al.: Exploiting User and Process Context for Knowledge Management Systems.
http://www.dfki.uni-kl.de/~maus/dok/vanElstAbeckerMaus01.pdf, Kaiserslautern 2001, Abruf am 20.11.2004.

[Ferb03]
Ferber, Reginald: Information Retrieval - Suchmodelle und Data-Mining-Verfahren für Textsammlungen und das Web.
http://information-retrieval.de/irb/, Oktober 2003, Abruf am 07.01.2005.

[Fran03]
Franke, Jürgen et al.: Text Mining Theoretical Aspects and Applications.
Physica-Verlag, Heidelberg 2003.

[Frei96]
Frei, Hans-Peter et al. (Hrsg.): SIGIR 96 – Proceedings of the 19th Annual International ACM SIGIR Conference on Research an Development in Information Retrieval.
Hartung-Gorre Verlag Konstanz, Konstanz 1996.

[Gamm95]
Gamma, Erich et al.: Design Patterns: Elements of Reusable Object-Oriented Software.
Addison-Wesley Professional, Boston 1995.

[GiSh04]
Giunchiglia, Fausto; Shvaiko, Pavel: Semantic Matching.
http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-82/SI_paper_07.pdf, Abruf am 25.11.2004.

[Glüc03]
Glückstein, Alexandra: Wissensmanagement – Eine neo-institutionalistische Perspektive.
Ars et unitas, Bad Feilnbach 2003.

[Grau04]
Grauer, Manfred: Leitlinien zum Thema „wissenschaftliches Arbeiten“.
http://www-winfo.uni-siegen.de/winfo/deutsch/lehre/offiziell/WissArbeiten, Abruf am 20.11.2004.

[Grub93]
Gruber, T. R.: A Translation Approach to Portable Ontology Specification.
In: Knowledge Acquisition 6(2), 1993, S.199-221.

[GüEr99]
Güting, Ralf Hartmut; Erwig, Martin: Übersetzerbau – Techniken, Werkzeuge, Anwendungen.
Springer, Berlin Heidelberg 1999.

[Hall04]
Haller, Johann et al.: Automatische Indexierung von wissenschaftlichen Texten – ein Experiment.
http://www.hwwa.de/Publikationen/Dokumentation/docs/0012-gastmeyer.pdf, Abruf am 28.11.2004.

[Hami93]
Hamilton, Patrick: Künstliche neuronale Netze – Grundprinzipien, Hintergründe, Anwendungen.
vde-Verlag, Berlin 1993.

[Haus00]
Hausser, Roland: Grundlagen der Computerlinguistik – Mensch-Maschine-Kommunikation in natürlicher Sprache.
Springer, Heidelberg 2000.

[Hayk94]
Haykin, Simon: Neural Networks – A Comprehensive Foundation.
Macmillan College Publishing Company, New York 1994.

[HeBr01]
Hesse, Wolfgang; Braun, Hubert v.: Wo kommen Objekte her? Ontologisch-erkenntnistheoretische Zugänge zum Objektbegriff.
http://www.tfh-berlin.de/~giak/arbeitskreise/softwaretechnik/dokumente/Fin8HesBra.pdf, 2001, Abruf am 25.11.2004.

[Heid99]
Heid, Ulrich: A linguistic bootstrapping approach to the extraction of term candidates from German text.
In: Benjamins, Tom: TERMINOLOGY Vol. 5(2), Amsterdam 1999.

[Herl04]
Herlocker, Jonathan L. et al.: Evaluating Collaborative Filtering Recommender Systems.
In: ACM Transactions on Information Systems, Vol. 22, No. 1, S.5-53, Januar 2004.

[Herm02]
Hermanns, Holger: Interactive Markov Chains – And the Quest for Quantified Quality.
Springer, Berlin Heidelberg 2002.

[HeWo98]
Heyer, Gerhard; Wolff, Christian(Hrsg.): Linguistik und neue Medien.
Deutscher Universitätsverlag, Wiesbaden 1998.

[Heye02]
Heyer, G. et al.: Möglichkeiten und Verfahren zur automatischen Gewinnung von Fachbegriffen aus Texten.
http://wortschatz.informatik.uni-leipzig.de/asv/publikationen/HeyFachbegriffe100902.pdf, 2002, Abruf am 20.11.2004.

[Hotz97]
Hotz, Günter: Algorithmische Informationstheorie.
B. G. Teubner Verlagsgesellschaft, Stuttgart Leipzig 1997.

[Hult04]
Hulth, Anette: Improved Automatic Keyword Extraction Given More Linguistic Knowledge.
http://www.dsv.su.se/~hulth/hulth_emnlp03.pdf, Abruf am 28.10.2004.

[Iglu04]
IGLU-Java : Java Library for Information Retrieval Research.
http://iglu-java.sourceforge.net , Abruf am 11.11.2004.

[Ingw92]
Ingwersen, Peter: Information Retrieval Interaction.
http://www.db.dk/pi/iri/files/Ingwersen_IRI.pdf, London 1992, Abruf am 20.11.2004.

[JaBe03]
Jarke, Prof. Dr. Matthias; Becks, Andreas: DocMiner – Text Mining mit Dokumentenlandkarten.
http://www-i5.informatik.rwth-aachen.de/~becks/papers/DocMINERFlyer.pdf, Abruf am 25.11.2004.

[Jaka04]
Jakarta Lucene – The Apache Jakarta Projekt.
http://jakarta.apache.org/lucene/docs/index.html, Abruf am 11.12.2004.

[Jama04]
JAMA: A Java Matrix Package.
http://math.nist.gov/javanumerics/jama, Abruf am 11.11.2004.

[Jark98]
Jarke, Matthias et al.: Scenario Management: An Interdisciplinary Approach.
In: Requirements Engineering 3 1998, S.155-173.

[Java04]
Sun Microsystems - Java Technology.
http://java.sun.com, Abruf am 11.12.2004.

[JaWa87]
Jaeger, Dr. Arno; Wäscher, Dr. Gerhard: Mathematische Probedeutik für Wirtschaftswissenschaftler – Lineare Algebra und Lineare Optimierung.
Oldenbourg Verlag, München 1987.

[Kamp02]
Kamphusmann, Thomas: Text Mining Eine praktische Marktübersicht.
Symposion Publishing, Düsseldorf 2002.

[KaTe01]
Karagiannis, Prof. Dr. Dimitris; Telesko, Dr. Rainer: Wissensmanagement – Konzepte der Künstlichen Intelligenz und des Softcomputing.
Oldenbourg Wissenschaftsverlag, München 2001.

[Kaut97]
Kautz, Henry et al.: ReferralWeb: Combining Social Networks and Collaborative Filtering.
In: Communications of the ACM, special issue, vol. 40, S. 63-65, März 1997.

[Klei99]
Kleiber, Georges et al. (Hrsg.): Kognitive Linguistik und Neurowissenschaften.
Gunter Narr Verlag Tübingen, Tübingen 1999.

[Kons97]
Konstan, Joseph A. et al.: GroupLens: Applying Collaborative Filtering to Usenet News.
In: Communications of the ACM, special issue, vol. 40, S. 77-87, März 1997.

[Kuhl92]
Kuhlen, Rainer (Hrsg.): Experimentelles und praktisches Information Retrieval.
Universitätsverlag Konstanz, Konstanz 1992.

[Lehr96]
Lehr, Andrea: Kollokationen und maschinenlesbare Korpora.
Niemeyer, Tübingen 1996.

[Lend89]
Lenders, Winfried (Hrsg.): Linguistische Datenverarbeitung und neue Medien.
Gunter Narr Verlag, Tübingen 1989.

[Lohm00]
Lohmann, Hartmut: KASCADE: Dokumentanreicherung und automatische Inhaltserschließung. In: Siebert, Irmgard: Schriften der Universitäts- und Landesbibliothek Düsseldorf, Düsseldorf August 2000.

[Maie01]
Maier, Roland: Knowledge Management Systems - Information and Communication Technologies for Knowledge Management.
Springer, Regensburg 2001.

[MaIs04]
Matsuo, Yutaka; Ishizuka, Mitsuru: Keyword Extraction from a Single Document using Word Co-occurrence Statistical Information.
http://www.miv.t.u-tokyo.ac.jp/papers/matsuoFLAIRS03.pdf, Abruf am 28.11.2004.

[Malo87]
Malone, Thomas W. et al.: Intelligent Information-Sharing Systems.
In: Communications of the ACM, Volume 30, Number 5, S. 390-402, Mai 1987.

[MaSt02]
Maedche, Alexander; Staab, Steffen: Measuring Similarity between Ontologies.
http://www.aifb.uni-karlsruhe.de/~sst/Research/Publications/ekaw2002-compare.pdf, 2002, Abruf am 25.11.2004.

[MaWa02]
Marcus, Robert; Watters, Beverly: Collective Knowledge - Das Wissen der Mitarbeiter unternehmensweit nutzen.
Microsoft Press, Redmond,Washington 2002.

[McAc00]
McDonald, David W.; Ackerman, Mark S.: Expertise Recommender: A Flexible Recommendation System and Architecture. In: Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work, Philadelphia, Dezember 2000.

[McBr03]
McArthur, Robert; Bruza Peter: Discovery of implicit and explecit connections between people using email utterance.
In: ECSCW 2003 – Proceedings of the Eighth Conference on Computer Supported Cooperative Work, S. 21-40, Helsinki, Finland, September 2003.

[McDo01]
McDonald, David W.: Evaluating Expertise Recommendations. In: Proceedings of the ACM 2001 International Conference on Supporting Group Work, Boulder, September 2001.

[McDo03]
McDonald, David W.: Recommending Collaboration with Social Networks: A Comparative Evaluation. In: Proceedings of the 2003 ACM Conference on Human Factors in Computer Systems, Ft. Lauderdale, April 2003.

[Midd04]
Middleton, Stuart E. et al.: Ontological User Profiling in Recommender Systems. In: ACM Transactions on Information Systems, Vol. 22, No. 1, S. 54-88, Januar 2004.

[MySQ04]
MySQL – The world's most popular open source database.
http://www.mysql.com, Abruf am 13.11.2004.

[NoTa97]
Nonaka, Ikujiro; Takeuchi, Hirotaka: Die Organisation des Wissens – Wie japanische Unternehmen eine brachliegende Ressource nutzbar machen.
Campus, Frankfurt/Main 1997.

[Patt97]
Patterson, Dan: Künstliche neuronale Netze.
2. Auflage, Prentice Hall Verlag, München 1997.

[Port80]
Porter, M. F.: An algorithm for suffix stripping.
http://www.tartarus.org/~martin/PorterStemmer/def.txt, 1980, Abruf am 06.11.2004.

[Prob99]
Probst, Gilbert et al.: Wissen Managen – Wie Unternehmen ihre wertvollste Ressource optimal nutzen.
3. Auflage, Gabler, Wiesbaden 1999

[Quas98]
Quasthoff, Uwe: Deutscher Wortschatz im Internet.
http://www.wortschatz.uni-leipzig.de/Papers/DeutscherWortschatzimInternet.doc, LDV-Forum 2/98, Abruf am 06.12.2004.

[QuWo00]
Quasthoff, Uwe; Wolff, Christian: Aiding Web Searches by Statistical Classification Tools.
http://wortschatz.informatik.uni-leipzig.de/asv/vortraege/DGfKPassau2000.pdf, Leipzig 2000, Abruf am 10.11.2004.

[Reich04]
Reichling, Tim et al.: Kontaktanbahnung in Lernplattformen: Ein Ansatz zur Förderung von Wissensprozessen.
In: Keil-Slawik, R. et al.: Mensch & Computer 2004: Allgegenwärtige Interaktion, München 2004.

[ReMi96]
Resnick, Paul; Miller, James: PICS: Internet Access Controls Without Censorship.
In: Communications of the ACM, Vol. 39, No. 10, S. 87-93, Oktober 1996.

[Resn94]
Resnick, Paul et al.: GroupLens: An Open Architecture for Collaborative Filtering of Netnews.
In: Proceedings of the 1994 ACM conference on Computer supported cooperative work, S. 175-186, Chapel Hill, North Carolina, United States 1994.

[ReVa97]
Resnick, Paul; Varian, Hal R.: Recommender systems.
In: Communications of the ACM, special issue, vol. 40, S. 56-58, März 1997.

[Runt00]
Runte, Matthias: Personalisierung im Internet – Individualisierte Angebote mit Collaborative Filtering.
Deutscher Universitäts-Verlag, Wiesbaden 2000.

[RuPo97]
Rucker, James; Polanco, Marcos J.: Personalized Navigation for the Web.
In: Communications of the ACM, special issue, vol. 40, S. 73-75, März 1997.

[Salt89]
Salton, Gerard: Automatic text processing: the transformation, analysis, and retrieval of information by computer.
Addison-Wesley, New York 1989.

[SaMc87]
Salton, Gerard; McGill, Michael J.: Introduction to Modern Information Retrieval.
McGraw-Hill, Hamburg 1987.

[Scha01]
Schaeder, Burkhard: Begriff und Benennung – Einführung in die Terminologielehre, Terminologiearbeit und Fachlexikographie.
3. Aufl., Siegen 2001.

[Schm92]
Schmitz, Ulrich: Computerlinguistik: Eine Einführung.
Westdeutscher Verlag, Opladen 1992.

[Snow04a]
Snowball Main Page: German stemming algorithm.
http://www.snowball.tartarus.org/german/stemmer.html, Abruf am 06.11.2004.

[Snow04b]
Snowball Main Page: German Stopwordlist.
http://www.snowball.tartarus.org/german/stop.txt, Abruf am 09.12.2004.

[Snow04c]
Snowball Main Page: French Stopwordlist.
http://www.snowball.tartarus.org/french/stop.txt, Abruf am 09.12.2004.

[Staa02]
Staab, Steffen: Wissensmanagement mit Ontologien und Metadaten.
In: Informatik Spektrum 25 Nr.3 (2002), S. 194-209.

[Stew98]
Stewart, Thomas A.: Der vierte Produktionsfaktor – Wachstum und Wettbewerbsvorteile durch Wissensmanagement.
Carl Hanser Verlag, München Wien 1998.

[StHa02]
Stahlknecht, *Peter; Hasenkamp, Ulrich*: Einführung in die Wirtschaftsinformatik.
Springer, 10. Aufl., Berlin Heidelberg 2002.

[Sull01]
Sullivan, Dan: Document Warehousing and Text Mining.
Wiley, New York 2001.

[Stubb96]
Stubbs, Michael: Text and Corpus Analysis.
Blackwell Publishers Ltd, Oxford 1996.

[StUH04]
Stiftung Universität Hildesheim: Ressourcen für multilinguales Information Retrieval – Stoppwortlisten.
http://www.unine.ch/Info/clef/germanST.txt, Abruf am 09.12.2004.

[TeHi01]
Terveen, Loren; Hill, Will: Beyond Recommender Systems: Helping People Help Each Other.
In: HCI In The New Millenium, Jack Carroll, Addison-Wesley, 2001.
http://citeseer.ist.psu.edu/terveen01beyond.html, Abruf am 20.11.2004.

[Terv97]
Terveen, Loren et al.: PHOAKS: A System for Sharing Recommendations.
In: Communications of the ACM, special issue, vol. 40, S. 59-62, März 1997.

[Tyle03]
Tyler, Joshua R. et al.: Email as Spectroscopy: Automated Discovery of Community Structure within Organizations.
In: Communities and Technologies – Proceedings of the First International Conference on Communities and Technologies, Dordrecht 2004.

[UsGr96]
Uschold, Mike; Gruninger Michael: ONTOLOGIES: Priciples, Methods and Applications.
http://citeseer.nj.nec.com/cache/papers/cs/3214/http:zSzzSzwww.cm.cf.ac.ukzSzUserzSzJ-C.PazzagliazSzReferenceszSzart:Uschold-96.pdf/uschold96ontologie.pdf, Februar 1996, Abruf am 25.11.2004.

[ViLi00]
Vivacqua, Adriana; Lieberman, Henry: Agents to Assist in Finding Help.
In: Proceedings of the SIGCHI conference on Human factors in computing systems, S. 65-72, Amsterdam 2000.

[WeMa00]
Wenzel, Claudia; Maus Heiko: An Approach to Context-driven Document Analysis and Understanding.
http://www.dfki.uni-kl.de/~maus/dok/WenzelMaus_DAS2000.pdf, Kaiserslautern 2000, Abruf am 20.11.2004.

[Wiki05]
Wikipedia – Die freie Enzyklopädie.
http://de.wikipedia.org/wiki/Hauptseite, Abruf am 07.01.2005.

[Wüst91]
Wüster, Eugen: Einführung in die allgemeine Terminologielehre und terminologische Lexographie.
Romanistischer Verlag, Bonn 1991.

[Zaun99]
Zaun, Detlef Peter: Künstliche neuronale Netze und Computerlinguistik.
Max Niemeyer Verlag, Tübingen 1999.

[Zell94]
Zell, Andreas: Simulation Neuronaler Netze.
Addison-Wesley, Bonn 1994.