Diplomarbeit

Ein Multiagentensystem
zur Erstellung eines
persönlichen Pressespiegels

Georg Veltmann

Diplomarbeit
am Fachbereich Informatik
der Universität Dortmund

20. Mai 1997

Betreuer:

Prof. Dr. Katharina Morik
Dipl. Inform. Volker Klingspor

Inhaltsverzeichnis

Kapitel 1

Einleitung und Überblick

Where is the wisdom we lost in knowledge,
where is the knowledge we lost in information,
where is the information we lost in data?

T.S. Elliot

In den letzten Jahren erlebte das Internet mit seinen verschiedenen Diensten ein ungeheures Wachstum. Was in den siebziger Jahren als Kommunikationsnetzwerk für eine Handvoll Wissenschaftler in den USA begann, hat sich in den neunziger Jahren weltweit zu *dem* Trendmedium überhaupt entwickelt. Vom Kleingärtnerverein bis zu den Vereinten Nationen, vom Pizzataxi bis zum global operierenden Lebensmittelkonzern, fast alle Aspekte des menschlichen Lebens sind mittlerweile „online" vertreten, und täglich kommen neue Anbieter hinzu. Die einfach zu bedienende und optisch ansprechende Benutzungsoberfläche des World Wide Web (WWW) und die Möglichkeit der schnellen und billigen weltweiten Kommunikation durch E-Mail machen das Internet für viele Anwender attraktiv. Jeder Benutzer des Internets hat direkten Zugang zu einer riesigen Menge von Daten, die auf Computern rund um den Globus gespeichert sind. Zu vielen Fragen lassen sich im Internet Antworten finden. Das Problem besteht nicht mehr darin, überhaupt eine Information zu bekommen, sondern in der Schwierigkeit, aus den vielen zur Verfügung stehenden Datenquellen diejenige auszuwählen, die die Information bereitstellen kann. Ist eine Quelle gefunden, taucht ein weiteres Problem auf: Wo ist in dem Wust der angebotenen Daten die gesuchte Information verborgen?

In traditionellen Datenspeichern wie Archiven oder Büchereien unterstützen menschliche Ansprechpartner wie Archivare oder Bibliothekare den Informationssuchenden. Sie beantworten Fragen zur Benutzung und geben erste Hinweise, wo eine Information zu finden ist, selbst wenn der Fragesteller noch nicht einmal genau weiß, wonach er eigentlich sucht.

Die heutige „Informationsflut" schwemmt diese Möglichkeit der Unterstützung bei der Informationssuche hinweg. Immer schneller entstehen immer mehr Daten in konventioneller oder elektronischer Form. Das Angebot z.B. auf dem Internet ist so groß, daß es für einen Einzelnen kaum möglich ist, sich einen Überblick zu verschaffen. Zudem haben die Daten oft eine sehr kurze Lebensdauer und werden schnell nutzlos. Dadurch wird es selbst für Spezialisten schwer, mit der Entwicklung Schritt zu halten.

Es gibt verschiedene Ansätze, den Datenstrom zu bewältigen. Viele Benutzer haben sich eine „Scheuklappenmentalität" angeeignet. Sie beschränken die Informationssuche auf eine kleine, überschaubare Anzahl ihnen bekannter Datenquellen und blenden den größten Teil des Angebots aus. Ist das Interesse stark spezialisiert oder sind die ausgewählten Quellen reich an Information, ist diese Lösung einfach und effektiv und funktioniert sowohl bei konventionellen als auch bei elektronischen Medien. Nichtsdestotrotz nutzt sie die angebotene Vielfalt an Daten kaum aus.

Sind die Daten in elektronischer Form gespeichert, liegt die Idee nahe, Computer als Hilfsmittel bei der Suche nach Information zu benutzen. Die Quellen werden nicht mehr direkt vom Benutzer angesprochen, stattdessen vermittelt ein Rechner zwischen Datenquellen und Benutzer. *Interface-* oder *Informationsagenten* sind Programme, die diese Vermittlerrolle übernehmen. Ziel ist es, über eine einheitliche Schnittstelle aus vielen Quellen Information bereitzustellen.

In der vorliegenden Arbeit wird das *PZ*-System (*PZ*für **P**ersönliche **Z**eitung), ein Interface-Agent für Tageszeitungen, beschrieben. Tageszeitungen bieten sich besonders für den Einsatz von vermittelnden Agenten an. Täglich erscheint eine Vielzahl von Artikeln zu unterschiedlichsten Themen, von denen nur ein Bruchteil für einen Benutzer interessant sind. Zudem bieten verschiedene Zeitungen je nach politischer Einstellung der Herausgeber auch verschiedene Sichtweisen auf ein und dasselbe Thema. Die meisten Leser entscheiden sich aber aus Zeitgründen gegen die große Meinungsvielfalt und lesen nur eine Zeitung, die sie u.a. nach Übereinstimmung mit ihrer eigenen Einstellung und nach dem Informationsgehalt auswählen. Meist werden auch aus der gewählten Zeitung nicht alle Artikel gelesen, sondern Artikel aus bestimmten Ressorts oder über gewisse Themen herausgepickt. Jeder Leser filtert also aus dem großen Angebot die für ihn interessanten Artikel heraus.

Falls der Leser nun gerade daran interessiert ist, ein Thema aus verschiedenen Blickwinkeln zu betrachten und einen Pressespiegel darüber zusammenzustellen, muß er eine große Anzahl von Artikeln aus mehreren Zeitungen lesen und bewerten. Diese Aufgabe, die viel Zeit in Anspruch nimmt, lohnt sich erst dann, wenn der Pressespiegel für mehrere Abnehmer erstellt wird. Sind mehrere Themen und viele Zeitungen zu berücksichtigen, wird die Arbeit sehr aufwendig.

Das *PZ*-System erledigt die Filterung mehrerer Tageszeitung automatisch. Täglich wird für jeden Benutzer ein persönlicher Pressespiegel zusammengestellt, der über das WWW gelesen werden kann. Das *PZ*-System lernt die individuellen Interessen seiner Benutzer durch die Beobachtung ihres Leseverhaltens und explizite Rückmeldungen über die Relevanz von Zeitungsartikeln. Es verfolgt die Veränderung der Interessen über die Zeit, die durch neue Nachrichten hervorgerufen wird. Das Interesse jedes einzelnen Benutzers wird im *PZ*-System durch ein Multiagentensystem mit verschieden spezialisierten Agenten modelliert. Jeder Agent stellt dabei einen Aspekt des vielseitigen Benutzerinteresses dar.

Diese Arbeit ist in vier Teile gegliedert. Im nächsten Kapitel werden umfassend die technischen Grundlagen des *Information Filtering* – der Selektion bestimmter Dokumente aus einem Dokumentenstrom – vorgestellt. Kapitel 3 führt in das weite Feld der *Agentensysteme* ein. Das *PZ*-System wird ausführlich in Kapitel 5 beschrieben, dem die Auswertung von Tests mit fiktiven Benutzern und des praktischen Systembetriebs über einen Zeitraum von knapp drei Wochen in Kapitel 6 folgt. Abschließend werden mögliche Erweiterungen des *PZ*-Systems diskutiert und die Ergebnisse der Arbeit zusammengefaßt.

Kapitel 2

Informationsfilter

Schon am Anfang der Elektronischen Datenverarbeitung, vor dem Einzug der Personal Computer in fast jedes Büro, der die Generierung und Speicherung von riesigen Informationsmassen zur Folge hatte, äußerte Luhn 1958 die Idee eines „Business Intelligence System“ zur Bewältigung der anwachsenden Menge zur Verfügung stehender Information [Luhn, 1958]. Spezielle Dokumentationsmitarbeiter sollten für jeden Benutzer eine Liste der von ihm z.B. aus einem Archiv angeforderten Dokumente pflegen und daraus die Interessen des Benutzers charakterisierende Schlüsselwörter ableiten. Mittels eines automatisierten Systems sollten diese Schlüsselwörter in neu veröffentlichten Dokumenten gesucht werden. Listen mit Zusammenfassungen der durch die Suche gefundenen Dokumente sollten dann den einzelnen Benutzern zugestellt werden. Bei Bedarf würden daraufhin einzelne Dokumente vollständig anfordern werden könen[1]. Die damals zur Verfügung stehenden Speichertechnologien (Mikrofilm und Lochkarten) und die geringe verfügbare Rechenleistung erschwerten die volle Realisierung dieser Ideen.

Speichersysteme mit großer Kapazität und schnelle Digitalrechner erlauben heute nicht nur die Speicherung von großen Datenmengen und schnellen Zugriff darauf, sondern ermöglichen es auch, einige der Aufgaben, die in Luhns System noch von Menschen ausgeführt wurden, nun von Computerprogrammen erledigen zu lassen. *Informationssysteme* ist die Disziplin der Informatik, die sich mit Fragestellungen der Datenverwaltung und -abfrage beschäftigt. Dazu wurden im Laufe der Zeit verschiedene Datenbankkonzepte (relational, deduktiv, objektorientiert und verteilt, um einige zu nennen) und -systeme zur Speicherung stark strukturierter Daten entwickelt. Wenig strukturierte Daten wie Freitext und Bilder erfordern jedoch spezielle Verfahren zur Speicherung und Abfrage. Deshalb entstand der Forschungszweig der *Dokumentdatenbanken.* Dokumente können sowohl Texte, Bilder, Töne, Videosequenzen als auch Fakten einer Wissensbasis sein. *Information Retrieval* (IR) und *Information Filtering* (IF) sind zwei Untergebiete, die sich besonders mit der *Suche* von Information in Dokumentdatenban-

[1] Beschreibung nach [Oard, 1996].

ken beschäftigen. Da eine effiziente Suche aber auch geeignete Datenstrukturen erfordert, betrachten IR und IF ebenfalls Aspekte der *Speicherung* von Daten.

Dieses Kapitel gibt eine Einführung in das Gebiet *Information Filtering*. Im ersten Abschnitt werden allgemein verschiedene Informationssuchprozesse beschrieben und voneinander abgegrenzt, bevor in Abschnitt 2.2 die Beziehung zwischen den eng verwandten Gebieten Information Filtering und Information Retrieval näher untersucht wird. Der Schwerpunkt des Kapitels liegt auf dem Abschnitt 2.3, der verschiedene im Information Filtering benutzte Techniken und Methoden vorstellt.

2.1 Informationssuchprozesse

Informationsfilter sind eine mögliche Ausprägung einer ganzen Klasse von Informationssuchprozessen. Ein *Informationssuchprozeß* ist jeder Prozeß, durch den Benutzer mit Hilfe eines automatisierten *Informationssystems* an Information gelangen möchten [Marchionini, 1995]. An dieser Stelle ist es sinnvoll, den Unterschied zwischen *Daten*, *Wissen* und *Information* deutlich zu machen. *Daten* sind Werte ohne Bedeutung, die mit syntaktischen Verfahren manipuliert werden können. Erst durch das Hinzufügen von Semantik entsteht aus Daten *Wissen* [Fuhr, 1997]. Selbst bei der Modellierung von Datenbanken fließt immer ein Teil der Semantik des Anwendungsgebietes in das Modell ein, so daß Datenbanken nicht nur Daten, sondern auch Wissen enthalten. Formalismen wie semantische Netze [Brachman und Schmolz, 1985] besitzen wesentlich mächtigere Ausdrucksmöglichkeiten für die Semantik von Daten. Wissen kann also in geeigneten Repräsentationen abgespeichert werden. Das in einer konkreten Situation zur Lösung von Problemen benötigte Wissen ist *Information*. Auf dieser pragmatischen Ebene existiert Information nur solange, bis die Aufgabe gelöst ist, und wird dann wieder zu Wissen.

Die Situation, in der ein Anwender bei der Verfolgung seiner Ziele an einen Punkt gelangt, an dem sein vorhandenes Wissen nicht mehr ausreicht, um die gesteckten Ziele zu erreichen, wird von Belkin und Croft als „anomalous state of knowledge" bezeichnet [Belkin und Croft, 1992]. Aus diesem Zustand fehlenden Wissens entsteht ein Informationsbedarf, der schließlich zur Initiierung eines Informationssuchprozesses führt. Taylor definiert vier Stadien der Entwicklung des Informationsbedarfs [Taylor, 1962]:

1.) **Unbewußter Informationsbedarf:** Zur Lösung eines Problems fehlt dem Anwender das nötige Wissen, er kommt nicht mehr weiter.

2.) **Bewußter Informationsbedarf:** Der Anwender erkennt, daß ihm Wissen fehlt.

3.) **Formalisierter Informationsbedarf:** Es wird bestimmt, welches Wissen zur Problemlösung beschafft werden muß.

4.) Formulierter Informationsbedarf: Das benötigte Wissen wird mittels einer kommunizierbaren Repräsentation spezifiziert.

Für Informationssuchprozesse sind das erste und letzte Stadium interessant. Der ursprüngliche, unbewußte Informationsbedarf soll durch das Suchergebnis befriedigt werden. Im Information Filtering wird deshalb unter Informationsbedarf oft nur dieses erste Stadium verstanden. In dieser Arbeit wird dafür der Ausdruck *Interesse* verwendet, um klare Abgrenzungen zu schaffen. Der vom Benutzer in einer geeigneten Sprache formulierte Informationsbedarf, der einem Informationssystem als Eingabe dient, wird bei Datenbanken und im Information Retrieval als *Anfrage* (engl. *Query*) bezeichnet. Die Repräsentation des Interesses in einem Informationsfilter hingegen wird im folgenden *Profil* genannt.

Informationssuchprozesse können nach einer Fülle von Merkmalen unterschieden werden, die in Tabelle 2.1 aufgeführt sind (siehe auch [Belkin und Croft, 1992] und [Loeb, 1992]). Einige bekannte Informationssuchprozesse sind [Oard und Marchionini, 1996]:

- Browsing
- Datenbankabfrage
- Information Extraction
- Information Retrieval
- Information Filtering

Browsing geht von einem breiten Benutzerinteresse aus. Größere Mengen vom Informationssystem bereitgestellter Dokumente werden vom Benutzer kurz durchgesehen. Basierend auf deren Inhalt werden Aktionen (z.B. das Folgen eines Hyperlinks oder das Lesen des vollständigen Dokumentes) für diese Dokumente gewählt. Die zeitliche Gestalt und die Struktur der Informationsquelle ist dabei unerheblich, da selbst verschiedene Dokumenttypen beim Browsing überblickt werden können. Der Benutzer übernimmt die kognitive Arbeit der Selektion derjenigen Dokumente, die die gewünschte Information beinhalten und das Informationssystem unterstützt ihn durch die geschickte Anzeige der Dokumente (siehe z.B. [Chimera und Shneiderman, 1994] oder [Cutting et al., 1992]). Beispiele für Browsing sind das „am Regal Entlanggehen" in einer Bibliothek oder das „Surfen" im World Wide Web[2]. Browsing ist als letzter Schritt in vielen anderen Informationssuchprozessen eingeschlossen: der Benutzer geht die Liste der gefundenen Dokumente durch und entscheidet dann, in welchem Dokument sich die gewünschte Information befindet.

[2]Die Programme, die zur Anzeige der HTML-Hypertextdateien des World Wide Web dienen, heißen daher *Browser*.

Gruppe	Merkmal	Ausprägung	Beipiel
Informations-quelle	Umfang	bekannt	Bibliothek
		unbekannt	World Wide Web
	zeitliche Gestalt	statisch	CD-ROM
		erweiterbar	Datenbank
		dynamisch	E-Mails
	Struktur	Menge	Textsammlungen
		Folge/Strom	News
		Graph/Netz	WWW
	Dokumenttypen	einheitlich	Artikel-Abstracts
		unterschiedlich	UNIX-Dateien
		zusammengesetzt	Multimediadokumente
Dokumenttyp	Art	strukturiert	Datenbankeinträge
		semi-strukturiert	E-Mails
		unstrukturiert	Zeitungsartikel
	Vokabular	beschränkt	Ganzzahlen
		unbeschränkt	Freitext
	Lebensdauer und Aktualität	Minuten	Börsennachrichten
		Jahre	Bücher
Benutzertyp	Benutzungs-häufigkeit	Einmalig	POI-Systeme
		Gelegentlich	Literaturdatenbank
		Ständig	Newsfilter
	Interessen-richtung	fokussiert	Datenbankabfrage
		gestreut	„Surfen“ im WWW

Tabelle 2.1: Merkmale von Informationssuchprozessen.

Bei der *Datenbankabfrage* wird von einem sehr eng fokussierten Benutzerinteresse ausgegangen. Im Extremfall ist es genau eine Information in einem Datum, die der Benutzer aus der Datenbank entnehmen möchte. Datenbanken verlangen in der Regel strukturierte Dokumente mit eingeschränktem Vokabular und einer festgelegten Semantik in der Repräsentation der Dokumente. Der Struktur in den Daten entspricht die Benutzung einer formalen, strukturierten Sprache (z.B. SQL) zur Formulierung einer Anfrage[3]. Vage Anfragen sind bei Datenbanken nicht möglich, nur exakt die Fragespezifikation erfüllende Objekte werden zurückgegeben. Das Ergebnis einer Datenbankabfrage ist die Information selber, nicht das Dokument, in dem die Information abgelegt ist.

Das Ziel von *Information Extraction* (IE) [Riloff und Lehnert, 1994] ist ebenfalls die Bereitstellung der reinen Information für den Benutzer. Aus einer Kollektion von üblicherweise wenig strukturierten Dokumenten sollen – angepaßt an die Ziele des Benutzers – Informationen herausgezogen werden. So bestand die Aufgabe im Rahmen der *Message Understanding Conferences* (MUC) (siehe [MUC3, 1991], [MUC4, 1992] und [MUC5, 1993]), einer von der US-Regierung unterstützten Konferenzreihe zur Evaluation verschiedener Information Extraction Systeme[4], darin, aus Nachrichtentexten des Foreign Broadcast Information Service Infomationen über terroristische Vorfälle in Südamerika zu extrahieren. Die IE-Systeme sollten bestimmte Felder eines generischen Templates für Terrorakte[5] mit Daten aus den Texten (z.B. den Namen von Opfern) oder mit Symbolen aus Mengen von fest vorgegebenen, erlaubten Feldinhalten (z.B. der Art des Terrorakts) füllen.

Die Informationssuchprozesse *Information Retrieval* und *Information Filtering* werden im nächsten Abschnitt ausführlich beschrieben und analysiert.

2.2 Die Beziehung Information Filtering – Information Retrieval

Information Retrieval und Information Filtering haben eine besondere Beziehung zueinander: die zugrundeliegenden Modelle gleichen sich sehr, nur die Gewichtung der einzelnen Teile ist verschieden. Beide verwenden eine Menge gleicher Techniken, ja sogar die gleichen Systeme. Wo liegen also die Unterschiede, die die

[3] Datenbanken, die natürlichsprachliche Anfragen erlauben, legen lediglich eine für menschliche Benutzer leichter verständliche Schicht um den eigentlichen Kern einer Datenbank, das Datenbankmanagementsystem (DBMS), das weiterhin nur strukturierte Anfragen bearbeiten kann.

[4] Neben der Message Understanding Conferences entstand 1993 eine weitere Konferenzreihe zur Evaluation von Information Retrieval und Information Filtering Systemen, die Text REtrieval Conferences (TREC).

[5] Das Konzept der MUC-Templates als Formalismus zur Wissensrepräsentation ähnelt den von Marvin Minsky entwickelten Frames [Minsky, 1975].

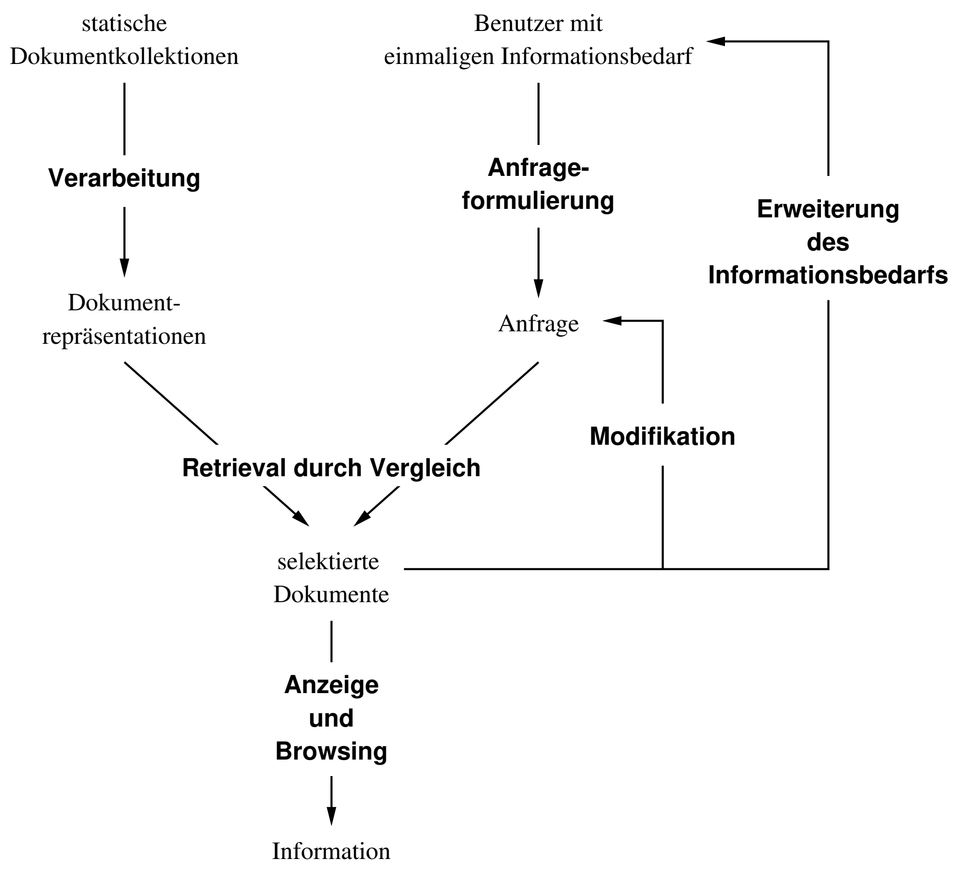


Abbildung 2.1: Ein allgemeines Modell für Information Retrieval.

beiden Gebiete voneinander trennen?

Belkin und Croft stellen in einem grundlegenden Artikel [Belkin und Croft, 1992] ein allgemeines Modell für Information Retrieval neben ein Modell für Information Filtering.

Eine meist große, statische Dokumentkollektion ist die Ausgangsbasis des in Abbildung 2.1 gezeigten Information Retrieval Modells. Die Dokumente werden mittels bestimmter Verarbeitungsschritte (im Information Retrieval Indexierung genannt) in eine für den Retrievalprozeß gut nutzbare Repräsentation umgewandelt. Benutzer wenden sich mit ihrem akuten Informationsbedarf an das System und formulieren eine Anfrage. Da die Art und Größe der Antwort nicht vorher bekannt ist (anders als z.B. bei der Datenbankabfrage), ist die Anfrage immer vage. Durch Vergleich der indexierten Dokumenten mit dieser Anfrage versucht das System, Dokumente zu selektieren, die die gewünschte Information beinhalten. Das System zeigt diese Dokumente an, so daß der Benutzer durch Browsing an die Information gelangen kann. Häufig sind aber eine Modifikation der Anfrage und eine erneute Selektion nötig, weil das Suchergebnis den Informationsbedarf

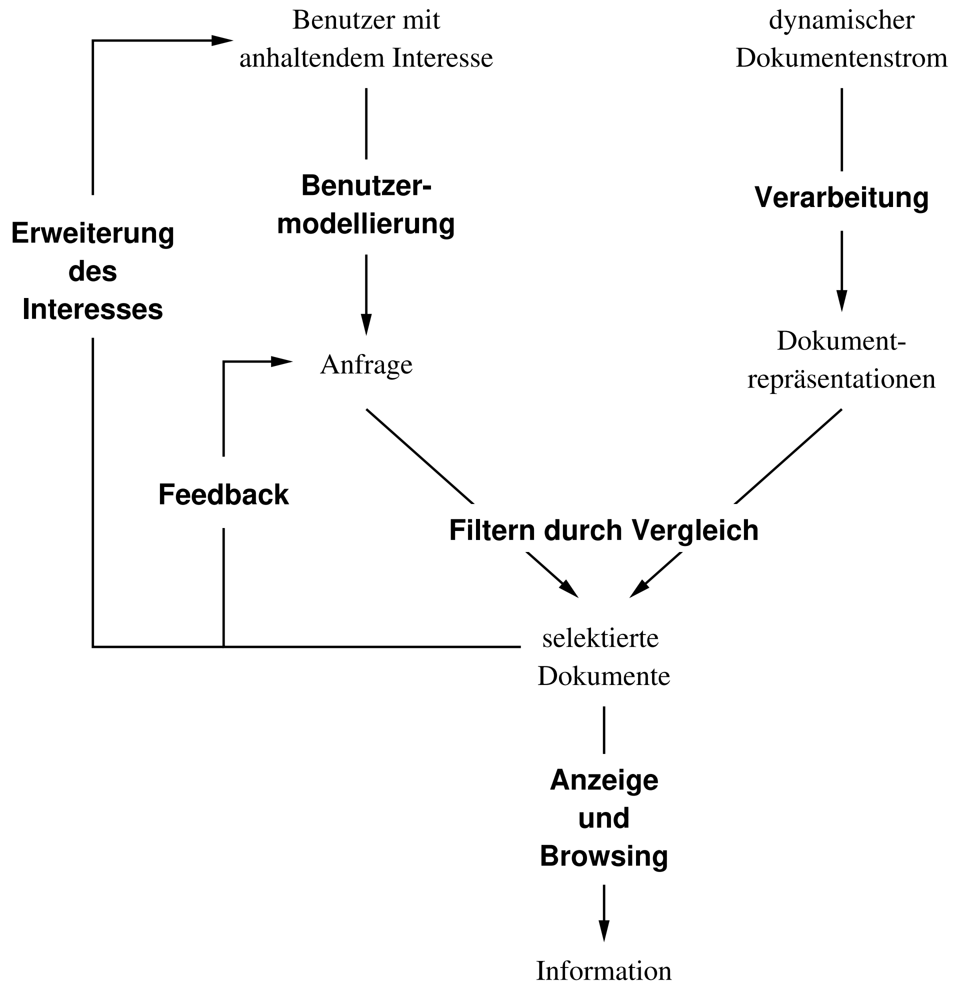


Abbildung 2.2: Ein allgemeines Modell für Information Filtering.

noch nicht gänzlich befriedigt. Durch die Evaluation der angezeigten Dokumente kann es aber auch dazu kommen, daß der Anwender weitere Lücken in seinem Wissen feststellt, die er mit neuen Anfragen an das Retrievalsystem zu schließen versuchen kann.

Das Information Filtering Modell (siehe Abbildung 2.2) fängt bei einem Benutzer an, der durch seine Ziele oder Wünsche ein anhaltendes Interessse an Information hat. Durch den Schritt der Benutzermodellierung legt das IF-System ein Profil an, das eine Repräsentation des Informationsbedarfs des Benutzers darstellt. Die Dokumente gelangen in Form eines dynamischen Stroms in den Informationsfilter, wo sie genau wie im Information Retrieval in eine für die weitere Verarbeitung geeignete Repräsentation gewandelt werden. Das Filtern erfolgt wieder über den Vergleich des Profils und der Dokumentrepräsentationen, indem entweder relevante Dokumente selektiert oder irrelevante aus dem Strom entfernt werden. Durch Feedback des Benutzers über die Relevanz der gefundenen und angezeigten Dokumente kann das System das Profil modifizieren und besser an die Interessen des Benutzers anpassen. Die Information ergibt sich erneut erst beim Browsen der gefundenen Dokumente.

Die beiden Modelle ähneln sich von der Struktur sehr stark, die Unterschiede der zwei Informationssuchprozesse stecken im Detail. Ein Vergleich anhand der Merkmale aus Tabelle 2.1 zeigt dies. Die Informationsquellen beim Information Retrieval sind relativ stabile, in ihrer Größe bekannte Mengen von Dokumenten, deren Lebensdauer gegenüber der des Informationsbedarfs der Benutzer hoch ist. Der Informationsfilter bezieht die aktuellen, eher kurzlebigen Dokumente nach und nach aus einem dynamischen Strom unbekannter Länge. Verfahren beider Modelle verarbeiten einheitliche, unstrukturierte Dokumente mit unbeschränktem Vokabular. Die Motivation der Benutzer ist wiederum unterschiedlich. Während IR-Systeme von einer wohldefinierten Gruppe aktiv suchender Benutzer mit jeweils für die Interaktion mit dem System einmaligem, eng fokussiertem Informationsbedarf verwendet werden, geht Information Filtering von Benutzern mit einem breit gestreuten, über mehrere Interaktionen stabil bleibenden Interesse aus. Das Informationssuchverhalten ist hier eher passiv. Da zu jedem Anwender ein Profil gespeichert wird, das Unbefugten Auskunft über das Verhalten und die Interessen eines Benutzers geben könnte, ist Datenschutz ein Thema im Information Filtering, im Information Retrieval wird es jedoch kaum beachtet, da der Benutzer dort in der einzelnen Suchepisode wenig Information über sich preisgibt.

Ungeachtet der Differenzen zwischen den beiden Gebieten teilen sie eine Menge von Verfahren und Methoden. Diese Parallelen erstrecken sich beispielsweise auf die Erstellung von Dokumentrepräsentationen und zum Vergleich zwischen der Repräsentation des Informationsbedarfs des Benutzers als Anfrage oder Profil und der Repräsentation der Dokumente. Einige Techniken werden im nächsten Abschnitt vorgestellt.

2.3 Techniken für Informationsfilter

Informationsfilter bestimmen über einen Vergleich der Repräsentationen von Dokumenten und Benutzerinteresse, ob ein Dokument dem Anwender angezeigt wird oder nicht. Die Art des Vergleiches bedingt angepaßte Repräsentationsformalismen für Dokumente und Interessen und ist daher zentral bei der Gestaltung eines Informationsfilters. Dabei lassen sich drei verschiedene Paradigmen zur Informationsselektion unterscheiden [Malone et al., 1992]:

Inhaltsbasiertes Filtern (*Content-based Filtering* oder *Cognitive Filtering*): Die Dokumente werden anhand ihres Inhalts und einer inhaltsbasierten Repräsentation des Benutzerinteresses selektiert.

Kollaboratives Filtern (*Collaborative Filtering* oder *Social Filtering*): Die Anmerkungen anderer Benutzer zu den Dokumenten und ein Maß für die Ähnlichkeit von Benutzerinteressen bilden die Basis für diesen Selektionsmechanismus.

Ökonomisches Filtern (*Economic Filtering*): Die Entscheidung für oder gegen die Selektion eines Dokumentes wird über Kosten-Nutzen-Abwägungen und implizite oder explizite Preismechanismen getroffen.

Der erste und zweite Ansatz haben weite Verbreitung in vielen Informationsfiltern gefunden, während das ökonomische Filtern ein Schattendasein führt. Der Nutzen eines Dokumentes kann entweder durch die Bewertungen anderer oder durch Analyse des Inhalts berechnet werden, erst dann ist es möglich, die Kosten für das Dokument damit zu vergleichen. Economic Filtering setzt also eigentlich einen der beiden anderen Mechanismen voraus und ist kein eigenständig verwendbares Filterverfahren. Inhaltsbasiertes und kollaboratives Filtern werden im folgenden getrennt in den Abschnitten 2.3.1 bzw. 2.3.2 erläutert.

Bis hierhin wurde bei der allgemeinen Beschreibung von Information Filtering immer von Dokumenten als Träger von Information gesprochen. Im weiteren Verlauf der Arbeit werden allerdings nur noch spezielle Dokumente, nämlich Texte, betrachtet. Einige der vorgestellten Techniken lassen sich ohne weiteres auf verschiedene Dokumenttypen übertragen, andere sind speziell auf die Eigenschaften von Texten zugeschnitten. Besonders im kollaborativen Filtern ist der Typ der zu filternden Dokumente oftmals unerheblich, das Filtern geschieht nur über Annotationen zu den Dokumenten in einem speziellen Format.

2.3.1 Inhaltsbasiertes Filtern

Die Eingabedaten eines inhaltsbasierten IF-Systems sind Texte in computerlesbarer Form[6], also erst einmal „nur“ Ketten von alphanumerischen Zeichen. Durch den Einsatz von Hintergrundwissen über den Aufbau von natürlicher Sprache können diese Urdaten zu Repräsentationen des Inhalts der Texte verarbeitet werden. Die einzelnen Einheiten der Inhaltsrepräsentation werden im Informations Retrieval und auch im Information Filtering Terme genannt. Wie aus Zeichenketten Terme entstehen, beschreibt Abschnitt 2.3.1.1, dem ein Abschnitt über verschiedene Techniken zur Speicherung der Terme folgt. *Termauswahl* und *Reparametrisierung* sind zwei in Abschnitt 2.3.1.3 beschriebene Methoden, um aus der großen Menge an Termen geeignete Eingaben für Lernalgorithmen zu erstellen. Der Einsatz von Lernverfahren ist für allgemeine Informationsfilter nicht zwingend nötig, ermöglicht aber eine schnelle und einfache Adaptation den an

[6]Die bis zu diesem Punkt nötigen Verarbeitungsschritte, z.B. die Benutzung eines OCR (Optical Character Recognition) Systems oder eines Internetroboters, sollen an dieser Stelle unberücksichtigt bleiben.

Benutzer auf der Grundlage von Dokumenten, die bereits durch den Benutzer gelesen und bewertet wurden. Die Vielfalt der bisher für inhaltsbasiertes Filtern eingesetzten Lernverfahren wird in Abschnitt 2.3.1.4 ausgebreitet.

2.3.1.1 Termgenerierung

Natürliche Sprache besteht auf verschiedenen Ebenen aus Einheiten. Zeichen setzen sich zu Silben zusammen, Silben zu Wörtern, Wörter zu Sätzen, Sätze zu Aussagen und Aussagen zu Dialogen. Die Linguistik unterscheidet ähnlich dieser Aufteilung *morphologische*, *lexikalische*, *syntaktische*, *semantische* und *pragmatische* Ebenen der Sprache, die eng miteinander verwoben sind. Die Repräsentation von Texten kann prinzipiell auf allen Ebenen modelliert werden, in Informationsfiltern gebräuchlich sind aber nur die ersten vier.

Zwar sind auf der morphologischen Ebene eigentlich Silben (als Morpheme) die interessanten Einheiten, doch wird in Informationsfiltern auf der Sub-Wortebene oft eine einfacherere Einteilung in Einheiten benutzt, die n-Gramme. Dazu wird ein Wort in alle Zeichenketten der Länge n zerlegt[7], die dann die Indexierungsterme ergeben. Trigramme, also die Spezialisierung von n auf 3, werden häufig benutzt, da sie eine gute Filterleistung erzielen.

Wörter sind mit Abstand die meistverwendeten Indexierungsterme in Informationsfiltern. Durch eine Tokenisierung an Leer- und Interpunktionszeichen lassen sich recht einfach die einzelnen Worte eines Textes identifizieren. Probleme ergeben sich auf der lexikalischen Ebene durch Homonyme. Dies sind Wörter, die trotz gleicher Schreibweise verschiedene Bedeutungen haben und erst durch die Erschließung des Kontextes auf einer übergeordneten Ebene disambiguiert werden können[8].

Syntax beschreibt den Aufbau von Sätzen aus Wörtern. Parser werden dazu benutzt, die syntaktische Struktur von Sätzen zu bestimmen. Bestimmte syntaktische Konstrukte (z.B. Nominalphrasen) können dann als Indexierungsterme benutzt werden [Lewis, 1992]. Eine einfachere Methode, die ohne Parsing auskommt, ist die Benutzung von Wort-n-Grammen, also aller Kombinationen von n aufeinanderfolgenden Wörtern des Textes [Sorensen und McElligott, 1995].

Die Terme sollen die Bedeutung des Textinhaltes reflektieren. Die Bedeutung von Sätzen ist Gegenstand der Semantik. Eine Möglichkeit der semantischen Indexierung ist die manuelle Zuweisung von festgelegten Deskriptoren gemäß einer speziellen hierarchischen Dokumentationssprache[9]. Wissensbasen wie WordNet [Miller, 1995] mit hierarchischen `ISA`- oder `HASA`-Relationen zwischen Konzep-

[7] **Zeitung** zerfällt also in die 4-Gramme **_Zei**, **Zeit**, **eitu**, **itun**, **tung** und **ung_**.

[8] Ein Beispiel für ein Homonym ist „Bank“, ein Wort, daß sowohl eine Sitzgelegenheit als auch ein Geldinstitut bezeichnen kann. Darüber hinaus gibt es noch andere, unüblichere Bedeutungen wie Gesteinsschicht, Wolkenballung, Sandbank, Arbeitstisch, Eisfeld, Einsatz im Glücksspiel, Uferstufung, Sonderschicht, Ringerstellung und Turngerät (siehe [Mackensen, 1986]).

[9] Die Stichwörter in Literaturdatenbanken sind Beispiele für Deskriptoren.

ten können zur automatischen Bestimmung von Synonymen (Wörtern mit gleicher Bedeutung), Meronymen (Wörtern, die Teile eines übergeordneten Begriffs bezeichnen) oder Hypernymen (Oberbegriffen) dienen. Speziell die Hypernyme eignen sich auf semantischer Ebene als Deskriptoren für Texte [Mock, 1996].

2.3.1.2 Speicherung der Terme

Durch die systematische Umwandlung von Zeichenketten in Terme stehen Dokumente nun als Sequenzen von Termen zur Verfügung. Wie aber wird diese Repräsentation in einem Informationsfilter abgespeichert?

Zunächst wird meist eine Vereinfachung durchgeführt: die Reihenfolge der Terme wird als vernachlässigbar betrachtet. Empirische Befunde belegen, daß die Filterqualität durch diese Annahme nicht beeinträchtigt wird[10] [Lewis, 1992]. Die Dokumente werden nicht mehr als Sequenzen (Abbildung 2.3a), sondern als Mengen von Termen (Abbildung 2.3b) betrachtet. Zur Speicherung dieser Mengen bieten sich nicht nur gebräuchliche Datenstrukturen wie Listen oder B-Bäume an. IR und IF benötigen ein schnelles Suchverfahren für einzelne Terme in der gesamten Dokumentkollektion. Für diesen Zweck wurden spezielle Speicherverfahren entwickelt. *Invertierte Listen* [Harmann et al., 1992] kehren die Darstellung einer Dokumentkollektion als eine Menge von Termmengen in eine Menge von Dokumentnummernmengen um. Zu jedem Term der Dokumentenmenge wird eine aufsteigend sortierte Liste von Dokumentnummern derjenigen Dokumente, die den Term beeinhalten, gespeichert (siehe Abbildung 2.3c). Zu einem gegebenen Profil können nun gezielt die Dokumente herausgesucht werden, in denen ein oder mehrere Terme aus dem Profil vorkommen.

Signaturen [Foloutsos, 1992] bilden Wörter oder ganze Textpassagen auf Bitstrings fester Länge ab. Dies wird i.a. mit Hashfunktionen durchgeführt. Die Suche über Signaturen ist durch spezielle Speicherungsformen deutlich effizienter als die Suche auf den ursprünglichen Termen. Überlagerungsfähige Signaturen erlauben die Darstellung ganzer Textpassagen in Form einer ODER-Verknüpfung der Signaturen der enthaltenen Wörter. Durch die Abbildung von mehreren Termen auf die gleiche Signatur kommt es bei der Suche nach Termen zu *false drops*, d.h. zur fälschlichen Selektion eines Dokuments. Diese Fehler müssen durch Nachbearbeitung anhand der Originaldokumenttexte beseitigt werden. Insgesamt läßt sich sagen, daß sich Signaturen vor allem als Vorselektionsmechanismus für die Suche von Termen in großen Dokumentkollektionen eignen.

Zusätzlich zum (geringen) Informationsverlust durch die Annahme, daß die Reihenfolge der Terme im Text vernachlässigt werden kann, wurde von den bisherigen Speicherverfahren noch eine weitere, durchaus wichtige Information aus

[10] Für deutsche Texte ist u.U. sogar eine Steigerung der Filterqualität möglich, da einer Suche nach festen Wortsequenzen die relativ freie Satzstellung im Deutschen entgegensteht. Da es aber bisher kaum Evaluationen von IR bzw. IF-Systemen mit deutschen Texten gibt, ist diese These noch nicht empirisch nachzuweisen.

a)
1) Die Nacht war kalt
2) Der Mond schien in der Nacht.
3) Kalt blies der Wind.
4) Der Mond war aufgegangen.

b)
1) {die, kalt, nacht, war}
2) {der, in, mond, nacht, schien}
3) {blies, der, kalt, wind}
4) {aufgegangen, der, mond, war}

c)

aufgegangen	4
blies	3
der	2, 3, 4
die	1
in	2
kalt	1, 3
mond	2, 4
nacht	1, 2
schien	2
war	1, 4
wind	3

Abbildung 2.3: Umwandlung von Dokumenten in Invertierte Listen: a) Ausgangsdokumente, b) Termmengen der Dokumente, c) Invertierte Liste der Dokumentkollektion.

den Dokumenten unterschlagen. Die Häufigkeit, mit der ein Term im Dokument auftritt, geht bei einer Textrepräsentation in Form einer Termmenge verloren. Das *Vektorraummodell* (engl. Vector Space Model, VSM) behebt dieses Manko, indem es eine einfache Darstellung dieser Information erlaubt. Bei der Anwendung des VSM nimmt man an, daß jeder mögliche Term t eine Dimension eines Vektorraumes aufspannt. Im Information Retrieval, das von einer statischen Dokumentkollektion ausgeht, bezieht sich dies auf alle Terme, die in der Kollektion vorkommen. Bei einem dynamischen, nicht vorhersehbaren Dokumentenstrom, wie er beim Informationsfiltern vorliegt, ist allerdings a priori keine solche Festlegung des Termraumes möglich. Um dessen Größe dennoch definieren zu können, bieten sich zwei mögliche Näherungslösungen an: die Einschränkung des Termraumes auf die Terme, die in einer großen, für den Strom repräsentativen Testkollektion auftauchen, oder die Bestimmung des Termraumes für Blöcke von Dokumenten aus dem Strom.

Nach der Festlegung der Dimensionalität des Termraumes können die Dokumente als Vektoren in diesem Raum aufgefaßt werden. Der Wert des Dokumentvektors in der Dimension (also zu dem Term) t heißt Termgewicht und muß zusätzlich, z.B. in der invertierten Liste, gespeichert werden. Die bisherige Textspeicherung als Menge entspricht der Vektorraumdarstellung mit binären Termgewichten. Die Gewichtung mit der *Termfrequenz* $TF(t, d)$, die angibt, wie oft der Term t im Dokument d vorkommt, berücksichtigt Häufigkeitsinformation nur innerhalb eines einzelnen Dokuments. Zu jedem Term kann aber auch die *Dokumentfrequenz* $DF(T)$ berechnet werden. Dies ist die Anzahl der Dokumente, in denen der Term t mindestens einmal vorkommt. Diese beiden Maße werden im Information Retrieval oft zur *TFIDF-Gewichtung* kombiniert ($n = |K|$ ist Anzahl der Dokumente in der Kollektion K) :

$$TFIDF(t,d) = TF(t,d) \cdot \log \frac{n}{DF(t)}$$

Charakteristisch für die TFIDF-Gewichtung gegenüber der Termfrequenz ist, daß Terme, die in vielen Dokumenten vorkommen, durch den zweiten Faktor eine Abschwächung erfahren, während Terme, die sehr speziell sind und nur in wenigen Dokumente vorkommen, ein stärkeres Gewicht erhalten.

2.3.1.3 Dimensionsreduktion des Termraumes

Zur automatischen Adaptation eines Informationsfilters an einen Benutzer werden Verfahren des Maschinellen Lernens eingesetzt. Dafür werden die einzelnen Terme als Attribute betrachtet, deren Attributwert das Termgewicht ist. Für viele Lernverfahren hat es sich als günstig erwiesen, nicht alle vorhandenen Attribute zum Lernen zu benutzen. Erstens erhöht ein hochdimensionaler Attributraum, wie er oft bei Informationsfiltern vorliegt[11], die Laufzeit vieler Lernalgorithmen so sehr, daß sich die praktische Anwendung verbietet[12]. Zweitens werden z.B. bei probabilistischen Lernverfahren entsprechend viele, oft aber nicht vorhandene Trainingsbeispiele benötigt, um statistisch sichere Aussagen über die Verteilung jedes Attributs machen zu können.

Die Verkleinerung des Attributraumes wird als *Dimensionsreduktion* bezeichnet. Ziel dabei ist, irrelevante Attribute zu entfernen, ohne das Lernergebnis negativ zu beeinflussen. Es gibt zwei Möglichkeiten, an dieses Problem heranzugehen. Entweder wird aus der gsamten Termmenge eine Untermenge der wichtigsten Terme ausgewählt, oder aus den vorhandenen Termen werden durch Reparametrisierung neue Terme generiert.

Termauswahl: Die gebräuchlichste Methode zur Auswahl von Termen ist die Entfernung von *Stoppwörtern.* Die Annahme ist, daß häufige Wörter wie „der“ oder „und“, die in den meisten Dokumenten vorkommen, wenig über den Inhalt der Dokumente aussagen. Normalerweise werden statische Standardstoppwortlisten verwendet, es besteht aber auch die Möglichkeit, die n häufigsten Terme der Dokumentkollektion zu entfernen [Lang, 1995].

Genau den entgegengesetzten Weg beschreitet das *Document Frequency Thresholding* [Yang und Pedersen, 1997]. Dabei werden alle Terme mit einer Dokumentfrequenz unterhalb eines festgelegten Schwellwertes entfernt. Es wird

[11] Bei der Wahl von Wörtern als Termen hat der Termraum prinzipiell eine Dimensionalität, die von der Anzahl der Wörter in der Sprache abhängt. Der neue Rechtschreibeduden listet z.B. mehr als 115000 Stichworte auf und berücksichtigt dabei Eigennamen kaum.

[12] Neuronale Netzwerke haben zum Beispiel eine polynomielle Laufzeit in bezug auf die Dimensionalität der Eingabe.

angenommen, daß diese seltenen Terme quasi als „Rauschen“ wenig informativ für das Filtern sind. Für eine Dokumentkollektion von 13000 Dokumenten mit 16000 verschiedenen Termen erhöhte eine Reduktion um 90% (dies entspricht der Entfernung aller Terme, die seltener als fünfzigmal vorkamen) sogar die Genauigkeit des Lernergebnisses [Yang und Pedersen, 1997].

Andere Termselektionskriterien wie *Information Gain*, *Mutual Information* und χ^2-*Statistik* (siehe [Schütze et al., 1995], [Yang und Pedersen, 1997]) verwenden nicht nur Kollektionsstatistiken, sondern beziehen die zu lernende Filteraufgabe in die Termauswahl ein.

Eine weitere Methode zur Termauswahl ist die Selektion von auf Wörtern basierenden Termen nach der Wortart. Genau wie der Index von Büchern meist nur aus Eigennamen, Substantiven, Verben und Adjektiven besteht, kann man auch nur Terme aus gewissen Wortklassen zur Indexierung zulassen. Mittels eines *Part-of-Speech-Taggers* wird die Wortart eines Terms festgestellt und dann entschieden, ob er zur Dokumentrepräsentation verwendet wird oder nicht.

Reparametrisierung: Eine morphologische Methode zur Zusammenfassung von Termen ist die *Grund-* oder *Stammformreduktion*. Während erstere nur Flexionen auf die Grundform reduziert (z.B. „stand“ auf „stehen“, „Wörter“ auf „Wort“), führt letztere Derivationen auf den Wortstamm zurück (z.B. „Rechner“, „Rechnung“ und „rechnete“ auf „rechn-“). In Sprachen mit Kompositabildung wie Deutsch oder Türkisch führt die Stammformreduktion durch die Zerlegung zusammengesetzter Wörter in mehrere Stämme unter Umständen aber auch zu einer Erhöhung der Dimensionalität des Termraumes.

Latent Semantic Indexing (LSI) [Deerwester et al., 1990] hat sich im Information Retrieval [Hull, 1995] und Information Filtering [Foltz und Dumais, 1992], [Oard, 1996] für die Reparametrisierung bewährt. LSI versucht die „latente“ verborgene Struktur im Gebrauch von Wörtern über Dokumente hinweg zu erfassen. Dazu wird aus der gesamten Dokumentkollektion eine Dokument-Term-Matrix X mit t Zeilen (eine für jeden Term) und d Spalten (eine für jedes Dokument) aufgestellt. Die Spalten der Matrix entsprechen dabei den Dokumentvektoren. Auf diese Matrix wird die *Singular Value Decomposition* (SVD) angewendet, eine der Eigenvektorzerlegung verwandte Technik, die X in TSD^T zerlegt. T ist eine $t \times r$ Matrix, deren Spalten die linken singulären Vektoren von X darstellen, D ist eine $r \times d$ Matrix, die die rechten singulären Vektoren von X als Spalten besitzt. Die $r \times r$ Matrix S enthält auf der Diagonalen der Größe nach geordnete, positive singuläre Werte, sonst Nullen (siehe Abbildung 2.4a). r ist der Rang der Matrix X. LSI führt sozusagen r künstliche Konzepte ein, die die gemeinsame Verwendung von verschiedenen Wörtern in verschiedenen Dokumenten repräsentieren. T kann als lineare Funktion betrachtet werden, die Dokumentvektoren (die Beschreibung, welche Terme in einem Dokument vorkommen) in Konzeptvektoren in einem Unterraum der Dimension r (Konzeptraum) überführt. Analog ist D eine

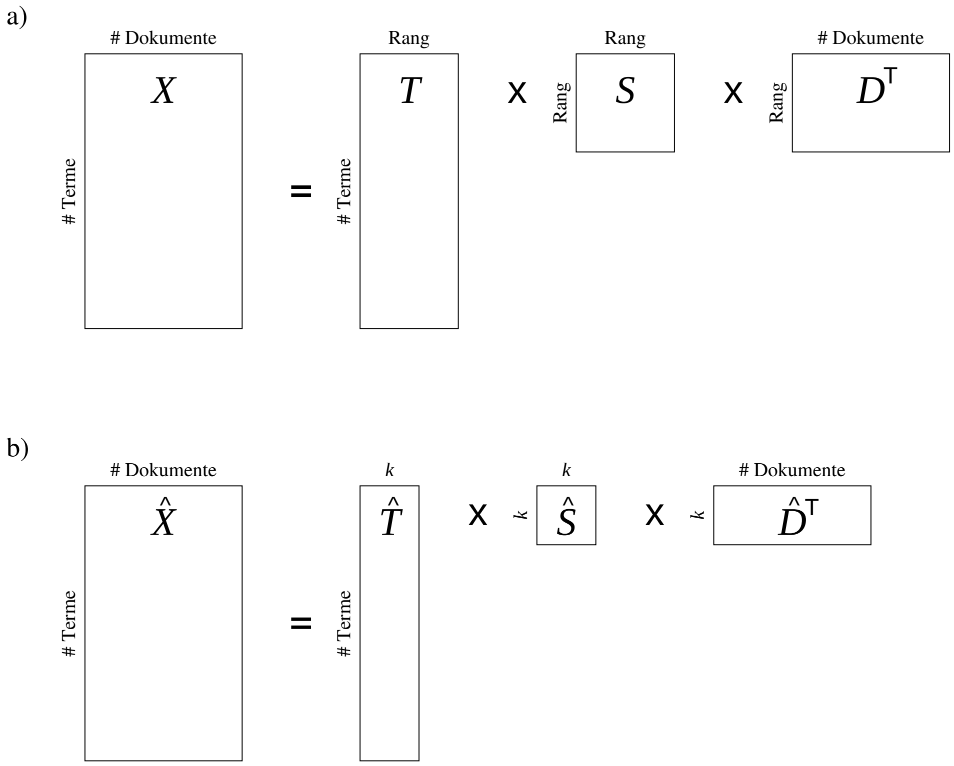


Abbildung 2.4: a) Singular Value Decomposition der Dokument-Term-Matrix X, b) Auswirkung der Reduktion von S auf die Matrixzerlegung.

lineare Funktion von Raum der Termvektoren (die einem Eintrag in der invertierten Liste ähnliche Beschreibung, in welchen Dokumenten ein Term auftaucht) in den Konzeptraum. Die eigentliche Dimensionsreduktion geschieht erst durch die Elimination von unwichtigen (d.h. kleinen) singulären Werten in der Matrix S. Es werden nur die k größten Werte erhalten (k typischerweise $\in [100, 300]$), alle anderen Werte werden auf 0 gesetzt. S kann dann die oberere Matrix $\hat{S}$ der Größe $k \times k$ entnommen werden, was der Entfernung von $r - k$ singulären Vektoren aus den Spalten bzw. Zeilen von T und D entspricht. T und D müssen also ebenfalls angepaßt werden, so daß sich sich eine Annäherung an die ursprüngliche Zerlegung $X \approx \hat{X} = \hat{T}\hat{S}\hat{D}^T$ (siehe Abbildung 2.4b) ergibt. Ein Dokument oder Profil kann durch Multiplikation mit $\hat{S}^{\frac{1}{2}}\hat{T}^T$ in den Konzeptraum überführt werden. Die Reparametrisierung mit LSI hat den Vorteil, daß die resultierenden Konzeptvektoren eine festgelegte Länge haben und schon vor dem Lernen eine Generalisierung über dem Suchraum durchgeführt wird. Lernverfahren, die zum *Overfitting* neigen, profitieren von dieser Eigenschaft. Nachteil beim Einsatz von

LSI sind die hohen Rechenkosten[13]. Außerdem sind die Vektoren im Konzeptraum dicht besetzt, so daß die Benutzung von invertierten Listen keine Vorteile mehr bringt. LSI-Profile müssen mit allen Dokumenten verglichen werden, nicht mehr nur mit den wenigen, die die gleichen Wörter beinhalten.

2.3.1.4 Lernverfahren für inhaltsbasiertes Filtern

Frühe Filtersysteme [Pollock, 1988] [Malone et al., 1992] verwendeten vom Benutzer handkodierte Regeln zur Suche nach Schlüsselwörtern in Texten. Adaptive inhaltsbasierte Filtersysteme versuchen, dem Benutzer die explizite Formulierung des Infomationsbedarfs abzunehmen. Grundlage der Adaptation sind Dokumente, die bereits vom Benutzer klassifiziert wurden. Neben einer binären Klassifikation als relevant/nicht relevant besteht auch die Möglichkeit, Dokumente in mehrere Klassen (z.B. eilig/wichtig/interessant/unwichtig) einzuteilen. Mit den Klassen sind meist Aktionen verbunden, z.B. werden relevante Dokumente vom Benutzer gelesen, nicht relevante dagegen gelöscht. Die klassifizierten Dokumente werden als Beispiele für das Benutzerinteresse betrachtet. Für jedes bewertete Dokument wird ein vollständiger Attributvektor aus den Termen des Dokuments und der Bewertung durch den Benutzer gebildet. Die Menge der Attributvektoren stellt die Trainingsmenge für Verfahren des Maschinellen Lernens dar. Zu neuen Dokumenten, von denen nur die Repräsentation durch Terme bekannt ist, soll anschließend die Bewertung des Benutzers vorhergesagt werden. Dieses Problem ist als *überwachtes Lernen* bekannt. Verschiedenste Verfahren sind zur Lösung dieses Problems erdacht worden. Hier sollen fünf verschiedene Ansätze vorgestellt werden, die bereits im inhaltsbasierten Filtern angewandt wurden.

Begriffslerner: Begriffslerner gehören zur Klasse der induktiven Lernverfahren. Sie versuchen die gegebenen Beispiele z.B. durch kompakte Regeln (dann meist in Form von Bedingungskonjunktionen) zu beschreiben. Bekannte Verfahren sind z.B. C4.5 [Quinlan, 1994][14] oder CN2 [Clark und Niblett, 1989].

Cohen benutzt den von ihm entwickelten RIPPER-Algorithmus in einem Vergleich mehrerer Lernverfahren für Informationsfilter mit binärer Bewertung [Cohen, 1996]. RIPPER startet mit einer leeren Regel, der solange Bedingungen in Form von Attributwerten hinzugefügt werden, bis kein negatives Beispiel mehr abgedeckt wird. Diese Regel wird in der Regelmenge abgespeichert, der Prozeß der Regelgenerierung wird erneut gestartet. Dieser Zyklus wird fortgesetzt, bis alle positiven Beispiele abgedeckt sind. Durch Nachbearbeitung wird die Regelmenge in der Größe optimiert, und durch *Cross-Validation* wird zu starke Spezialisierung auf die Trainingsmenge (*Overfitting*) vermieden. In diesem Schritt kann auch ein Verhältnis zwischen den Kosten einer falschen Negativ- und denen einer falschen Positivklassifikation spezifiziert werden, um den Trade-off zwischen der

[13] Da die Matrix X spärlich besetzt ist, ist eine Laufzeit $O(t,d)$ zu erzielen, obwohl eine vollständige SVD eigentlich $O(t^2,d)$ benötigt.

[14] Hier wird erst ein Entscheidungsbaum gelernt, aus dem Regeln generiert werden können.

Anzahl gefundener relvanter Dokumente und der Genauigkeit der Klassifikation variieren zu können, wie es in instanzbasierten Lernverfahren möglich ist. Cohen setzt den RIPPER-Algorithmus zum Filtern von E-Mails ein und berichtet eine gleichwertige Leistung des Verfahrens im Vergleich mit dem instanzbasierten Rocchio-Algorithmus.

Weitere Anwendungen von Begriffslernern für Informationsfilter finden sich in [Fischer und Stevens, 1991], [Apté et al., 1994] und [Payne und Edwards, 1995a]. Ein Vorteil dieser Klasse von Lernalgorithmen ist, daß die erzeugte Entscheidungsfunktion (z.B. eine Regelmenge) für den Benutzer sehr gut erfaßbar und dadurch einfach nachvollziehbar wird. In InfoScope [Fischer und Stevens, 1991] wird diese Eigenschaft dazu benutzt, im Dialog zwischen lernendem Agenten und Benutzer eine optimale Regelmenge zu erstellen.

Instanzbasierte Lerner: Die Idee des instanzbasierten Lernens besteht darin, im Lernschritt zunächst nur Repräsentationen aller Beispiele zu speichern. Zur Klassifikation eines neuen Dokumentes wird das ähnlichste Beispiel zum Dokument gesucht und dessen Klassifikation übernommen.

Der k-Nearest-Neighbor Algorithmus (k-NN) etwa bestimmt die k nächsten Beispiele zum neuen Dokument anhand eines Distanz- oder Ähnlichkeitsmaßes. Ist C die Klasse mit der größten Anzahl k_C von Beispielen unter den k nächsten Nachbarn, so wird das Dokument mit Konfidenz k_C/k als der Klasse C zugehörig klassifiziert. Durch die Festlegung eines Konfidenzschwellwertes werden unsichere Klassifikationen ausgesondert. Bei großen Trainingsmengen und/oder großen Attributvektoren verbietet sich die Speicherung aller Trainingsinstanzen alleine aus Speicherplatzgründen. Auch die Laufzeit für die Bestimmung der nächsten Nachbarn durch den Vergleich des neuen Dokuments mit allen Beispieldokumenten wächst bei einer großen Trainingsmenge stark an.

Eine Alternative ist die Speicherung von Prototypen (*Prototypical Learning* [Biberman, 1995]). Dazu werden die für die Klassifikation nützlichsten oder signifikantesten Beispiele identifiziert und gespeichert, wohingegen Beispiele, die wenig zur Klassifikation beitragen, gelöscht werden. Das Lernverfahren IBPL2 folgt diesem Ansatz [Payne und Edwards, 1995b].

Noch weiter geht die Zusammenfassung der Beispiele einer Klasse zu *einem* Klassenprototypen. Im Information Retrieval wird diese Methode zur Verbesserung der Anfrage durch *Relevance Feedback* [Salton und Buckley, 1990] mit dem *Rocchio-Algorithmus* [Rocchio, 1971] benutzt. Auch in Informationsfiltern kann der Rocchio-Algorithmus eingesetzt werden. Dieser Ansatz wird im *PZ*-System verfolgt. Aus diesem Grund wird der Rocchio-Algorithmus hier nicht weiter erklärt, sondern erst im Abschnitt 5.3.1 näher besprochen. [Lang, 1995], [Schütze et al., 1995], [Mukhopadhyay et al., 1996] und [de Kroon et al., 1996] beschreiben weitere Informationsfilter, die den Rocchio-Algorithmus benutzen.

Probabilistische Klassifikatoren: Diese Lernverfahren benutzen stochastische Modelle, um für ein neues Beispiel die Wahrscheinlichkeit der Zugehörigkeit zu einer Klasse zu schätzen.

Der *Naive Bayes'sche Klassifikator* (siehe z.B. [Joachims, 1996b]) geht davon aus, daß die Terme eines Dokumentes durch unabhängiges Ziehen anhand einer bestimmten Verteilung erzeugt werden (konditionale Unabhängigkeitsannahme). Diese Annahme ist nicht richtig, als Näherung jedoch sinnvoll und praktikabel [Joachims, 1996a]. Die Verteilung, aus der gezogen wird, ist nicht für alle Dokumente gleich, sondern hängt von der Klasse des Dokuments ab. Der Klassifikator versucht diese Abhängigkeit aufzuspüren. Er weist dem Dokument d die Klasse C aus der Menge aller Klassen $\mathcal{C}$ zu, für die die Wahrscheinlichkeit $P(C|d)$ maximal ist.

$$H_{Bayes}(d) = \operatorname*{argmax}_{C \in \mathcal{C}} P(C|d)$$

Mittels des Satzes von Bayes kann diese Wahrscheinlichkeit zerlegt werden in

$$P(C|d) = \frac{P(d|C) \cdot P(C)}{\sum_{C' \in \mathcal{C}} P(d|C') \cdot P(C')}$$

Dabei kann $P(C)$ durch

$$\hat{P}(C) = \frac{|C|}{\sum_{C' \in \mathcal{C}} |C'|}$$

und $P(d|C)$ unter Hinzunahme der Unabhängigkeitsannahme durch

$$\hat{P}(d|C) = \prod_{t_i \in d} \frac{1 + TF(t_i, C)}{|T| + \sum_{t' \in T} TF(t', C)}$$

geschätzt werden. Dabei ist T ist die Menge aller Terme, $TF(t, C)$ die Häufigkeit des Terms t über alle Dokumente der Klasse C. Das Dokument d wird hier als Menge von Termen angesehen.

Neuronale Netze: Aus der Beschäftigung mit dem Aufbau des menschlichen Gehirns entstand diese Klasse von Lernverfahren. Die Bausteine des Gehirns sind Neuronen, also Gehirnzellen, die über Synapsen sehr stark miteinander verbunden sind. Über die Synapsen werden Reize (Erregung oder Hemmung) ausgetauscht. Übersteigt die Erregung einer Zelle über die Eingangssynapsen einen Schwellwert, so werden die Ausgangssynapsen aktiviert (das Neuron „feuert"). Diese wiederum erregen oder hemmen angeschlossene Neuronen. Lernen geschieht zum einen durch den Aufbau von Verbindungen zwischen Neuronen, und zum anderen durch die Anpassung der Übertragungsstärke an den Synapsen.

Diese Struktur des Gehirns wird durch künstliche neuronale Netze nachgeahmt. Die Werte an den Eingänge eines Neurons werden mit den Eingängen assozierten Gewichten multipliziert, aufsummiert und – abhängig vom Schwellwert – wird das Ergebnis an die Ausgänge des Neurons weitergereicht. Es gibt unterschiedliche Ansätze zur Organisation des Neuronennetzes.

Das *Perzeptron* ist eine Hierarchie von Neuronen. Die Eingänge der untersten Hierarchieschicht werden mit den Attributen der Beispiele (im IF den Termvektoren) gekoppelt, die Ausgänge der obersten Schicht entsprechen den vorgegebenen Klassen. Das Netzwerk lernt, indem nach und nach die Beispiele am Eingang angelegt werden und die Signale durch das Netzwerk propagiert werden. Der Fehler zwischen Netzwerkergebnis und Vorgabe wird mittels *Backpropagation* minimiert, d.h., die Eingangsgewichte der Neuronen werden durch ein Gradientenabstiegsverfahren angepaßt. Dieser Prozeß wird mit den veränderten Gewichten wiederholt, bis das Netzwerk einen Gleichgewichtszustand erreicht und die Klassifikationsaufgabe optimal löst. Anwendungen dieser Art von neuronalen Netzen in Informationsfiltern finden sich in [Mock und Vemuri, 1994] [Schütze et al., 1995] und [Pannu und Sycara, 1996].

Ein anderer Ansatz benützt nicht-hierarchische Netzwerke von Neuronen zum Lernen eines Benutzerprofils für Informationsfilter [Jennings und Higuchi, 1992]. Hier stellt jedes Neuron einen Term dar. Die Häufigkeit des gemeinsamen Vorkommens zweier Terme in den Beispieldokumenten spiegelt sich in der Stärke der Verbindung zwischen den entsprechenden Neuronen wider, während die Erregungsenergie eines Neurons die Häufigkeit des Terms repräsentiert. Schwache Neuronen und Neuronenverbindungen werden entfernt. Bei der Klassifikation werden durch ein neues Dokument alle Neuronen im Netzwerk aktiviert, deren Terme im Dokument vorhanden sind. Die aktivierten Neuronen leiten ihre Energie an die angeschlossenen Neuronen weiter. Ist deren Eingangsschwellwert überschritten, werden diese Neuronen ebenfalls aktiviert. Diese *spreading activation* setzt sich solange fort, bis keine Neuronen mehr feuern können. Die Summe der Energien der aktiven Neuronen im Endzustand kann als Maß der Ähnlichkeit zwischen Dokument und Benutzerprofil verwendet werden. Ein ähnliches Verfahren wird in [Sorensen und McElligott, 1995] verwendet.

Genetische Algorithmen: In der Natur passen sich Lebewesen ständig durch Evolution an veränderte Umgebungssituationen an. Genetische Algorithmen (GA) versuchen die Evolutionsmechanismen abzubilden. Dies geschieht durch die zwei der Natur abgeschauten Operatoren *Crossover* (Austausch von Daten zwischen Hypothesen) und *Mutation* (zufällige Veränderung eines Datums innerhalb einer Hypothese) zur Erzeugung neuer Klassifikationshypothesen aus bereits vorhandenen. Mit Hilfe einer *Fitness*-Funktion zur Beurteilung der Güte einer Hypothese wird eine Population von mehreren konkurrierenden Hypothesen optimiert [Holland, 1975]. In jedem Schritt – bei GA Generation genannt – des Lernverfahrens werden schlechte Hypothesen mit geringer Fitness aus der Population entfernt, während gute Hypothesen „Nachkommen“ durch Mutation und

Crossover erzeugen dürfen und so die Population wieder auffüllen. Der Vorteil der genetischen Algorithmen bei der Suche nach einer optimalen Klassifikationshypothese ist die gute Balance zwischen der Erforschung des Hypothesenraums durch die genetischen Operatoren und der Ausnutzung des bisherigen Suchergebnisses durch die Erhaltung guter Hypothesen. GA verwenden jedoch kein Hintergrundwissen über den Aufbau der Hypothesen, um die Suche im Hypothesenraum schnell zu einem guten Ergebnis zu leiten. Die Schwierigkeit bei GA's ist die Definition einer geeigneten Fitness-Funktion und die Festlegung von Parametern wie Populationsgröße oder Mutationsrate.

Sheth benutzt in seinem News-Filtersystem Profile aus Termvektoren als Hypothesen [Sheth, 1994]. Beim Crossover werden Abschnitte zweier Vektoren ausgetauscht, Mutation ist auf einige ausgewählte Attribute beschränkt. Die Klassifikation eines neuen Dokumentes geschieht durch Vergleiche mit allen Profilen der Population, wobei das Vergleichsergebnis eines Profils mit seiner Fitness multipliziert wird. Anschließend werden die Ergebnisse zu einem Konfidenzwert für die Klassifikation zusammengefaßt. „Fitte" Profile tragen also gemäß ihrer Fitness stärker zur Klassifikation bei.

2.3.2 Kollaboratives Filtern

In den bisher beschriebenen Informationsfiltern wurde das Filtern immer auf einen einzelnen Benutzer zentriert durchgeführt. Zur Unterstützung von mehreren Benutzern wird einfach der Filterprozeß parallel für jeden Benutzer gestartet. Yan und Garcia-Molina versuchen im SIFT-System, durch die gemeinsame Speicherung ähnlicher Profile mehrerer Benutzer in einer Art invertierter Liste eine Effizienzsteigerung zu erreichen [Yan und Garcia-Molina, 1994] [Yan und Garcia-Molina, 1995]. Dies ist ein rein technischer Ansatz, der die Beziehung der Benutzer untereinander nicht modelliert, aber die Tatsache ausnutzt, daß für mehrere Benutzer gefiltert wird.

Kollaboratives Filtern benutzt das Wissen über andere Benutzer des Systems zum Filtern der Dokumente für einen einzelnen Benutzer. Das in einem Informationsfilter in Form des Profils gespeicherte Wissen über einen Benutzer kann neben einer inhaltlichen Repräsentation des Benutzerinteresses auch aus einer Liste der empfangenen Dokumente und Anmerkungen zu ihnen bestehen. Bewertungen des Dokuments oder Aktionen, die auf das Dokument angewendet wurden (z.B. Lesen, Speichern, Löschen, Weitersenden, Ausdrucken), sind mögliche Annotationen. Erhält nun der Filter eines Benutzer Zugriff auf diese Daten anderer Benutzer, lassen sich Ähnlichkeiten zwischen Profilen (und damit Benutzern) berechnen. Das Filtern besteht nun darin die Reaktion des aktuellen Benutzers auf dieses Dokument, aufgrund der Anmerkungen ähnlicher Benutzer zu einem Dokument vorherzusagen.

Die Erschließung des Dokumentinhaltes wird also nicht vom Filtersystem durchgeführt, sondern den Benutzern überlassen und vom System nur in Form

Art der Anmerkung	binäre Bewertung
	Bewertung auf Skala $(1 \ldots n)$
	Beschreibungen des Inhalts durch Terme
	Aktionen für das Dokument
Benutzer-bezug	anonym
	Benutzung von Pseudonymen
	Klarnamen

Tabelle 2.2: Unterschiede von kollaborativen Filtersystemen nach Anmerkungsmerkmalen (vergleiche [Resnick und Varian, 1997]).

von Anmerkungen aufgezeichnet. Dies ermöglicht einerseits das Filtern von Dokumentarten wie Musik, Bildern oder Videofilmen, deren Inhalt bisher kaum durch automatische Verfahren erschlossen werden kann, als auch die Beachtung von Textmerkmalen wie Lesbarkeit, Stil oder Klarheit des Gedankengangs, die ebenfalls nur schwer – falls überhaupt – durch Rechner erfaßt werden können [Maltz, 1994].

Kollaborative Filtersysteme besitzen den weiteren Vorteil, daß sie nicht so stark zur Spezialisierung neigen wie inhaltsbasierte Systeme. Dort werden dem Benutzer nur Dokumente angeboten, die sich mit dem bereits geäußerten Interesse decken. Kollaborative Systeme können durch die Generalisierung über mehrere Benutzer auch verborgene Interessen entdecken und dafür Dokumente bereitstellen, ohne daß der aktuelle Benutzer vorher jemals (inhaltlich) ähnliche Dokumente als interessant markiert hätte.

Durch den Rückgriff auf das Wissen anderer Benutzer beim Filtern ergibt sich jedoch das Problem, daß das zu filternde Dokument schon von jemandem mit Annotationen versehen worden sein muß. Ein rein kollaborativ arbeitendes Filtersystem kann also keine Aussage über ein ganz neues Dokument machen, sondern muß erst auf eifrige Benutzer warten, die auch ungefilterterte Dokumente lesen und dabei Anmerkungen erstellen. Diese Lücke könnte durch hybride Informationsfilter geschlossen werden. Im Falle eines bisher unannotierten Dokuments würde die inhaltsbasierte Komponente die Vorhersage der Relevanz übernehmen, ansonsten könnten inhaltsbasiertes und kollaboratives Filtern gleichberechtigt nebeneinander stehen. Erste Ansätze zur Realisierung dieses Konzepts (bisher ohne Ergebnisse) in einem adaptiven Filtersystem finden sich im GroupLens-Projekt [Konstan et al., 1997].

Kollaborative Filtersysteme unterscheiden sich nach der Art der Anmerkungen, die zu einem Dokument gemacht werden dürfen, und in dem Maße, wie Annotationen auf einen Benutzer zurückführbar sind (siehe Tabelle 2.2). Im folgenden werden zunächst zwei nicht-adaptive Systeme, die nicht nur kollaborativ arbeiten, sondern auch inhaltsbasierte Attribute zur Dokumentselektion benutzen, und im Anschluß drei rein auf kollaborativen Anmerkungen basierende adaptive Filtersysteme vorgestellt.

Im Tapestry-System [Goldberg et al., 1992] kann ein Benutzer eine binäre Bewertung (relevant/nicht relevant) für Dokumente vergeben, die zusammen mit seinem Namen getrennt von den Dokumenten in einer Datenbank gespeichert werden. Mit Hilfe von Regeln in einer SQL-ähnlichen Abfragesprache, die von den Benutzern selbst erstellt werden müssen, können nun Dokumente selektiert werden. Dabei kann die Selektion gezielt nach Bewertungen bestimmter anderer Anwender (z.B. solchen, denen der Benutzer besonders vertraut oder von denen er weiß, daß sie die gleichen Interessen haben) vorgenommen werden. Neben der Bewertung gibt es auch noch eine Reihe von Inhaltsattributen (z.B. bei News-Artikeln Absender, Adressat, Subject oder News-Gruppe), die ebenfalls in Regeln verwendet werden dürfen. Der Ansatz von Tapestry funktioniert nur bei einer relativ kleinen Gruppe von Systembenutzern, da die Quelle der Bewertung in Regeln explizit genannt wird und daher auch der zugehörige Benutzer bekannt sein muß. Der Schutz von persönlichen Daten (hier Bewertungen von Dokumenten) ist nicht möglich, ein Benutzer muß der Gruppe vertrauen.

Ebenfalls nur für kleine Gruppen ist der Newsreader URN gedacht [Brewer und Johnson, 1994]. Die Annotationen sind sind hier anonym und bestehen aus Wörtern, mit denen der Benutzer ein Dokument beschreibt. Die Menge aller Annotationen zu einem Dokument bildet eine kollaborativ erstellte Beschreibung des Inhalts, die zusammen mit einer maschinell erstellten Beschreibung bei der Selektion des Dokumentes verwendet werden kann. URN führt quasi eine kollaborative Komponente in den Termgenerierungsteil eines inhaltsbasierten Filtersystems ein.

Das System GoodNews [Maltz, 1994] ist ebenfalls zum Filtern von Usenet News gedacht und sammelt dazu anonyme, mehrstufige Bewertungen (*votes*) zu den Dokumenten. Der Durchschnitt der bisherigen Bewertungen zu einem Dokument ist der Schätzwert für die Bewertung durch Benutzer, die das Dokument noch nicht gesehen haben. Das Hauptaugenmerk der Arbeit von Maltz liegt bei der effizienten Sammlung und Verteilung der Bewertungen durch spezielle *vote server*. Das System berücksichtigt weniger die individuellen Interessen einzelner Benutzer, sondern bildet ein gemitteltes Interesse aller Benutzer. e Ringo [Shardanand und Maes, 1995] und GroupLens [Resnick et al., 1994], [Konstan et al., 1997] verwenden Bewertungen auf einer Skala $1 \dots 7$ bzw. $1 \dots 5$, die von Benutzern unter einem Pseudonym an das System geschickt werden. Das System kann also einzelne Benutzer identifizieren[15], während andere Benutzer keinen Zugriff auf personenbezogene Daten haben[16]. Die Bewertungen eines Benutzers zu

[15]Eine Eigenschaft, die ein Filtersystem mit mehreren Benutzern sowieso besitzen muß, um die für einen bestimmten Benutzer gefilterten Dokumente auch an die richtige „Adresse“ abzuliefern.

[16]Im kommerzialisierten Ringo-System (Firefly [Firefly Network Inc., 1997]) wurde mittlerweile von einigen Benutzern der Wunsch geäußert, die Quellen von Bewertungen kennenzu-

den Dokumenten bilden das Benutzerprofil in Form eines Vektors. Mit Hilfe eines Ähnlichkeitsmaßes für diese Vektoren (es wurden u.a. die durchschnittliche quadratische Differenz und der Pearson r Korrelationskoeffizient verwendet) werden einige sehr ähnliche Benutzer identifiziert und deren Bewertungen mit der Ähnlichkeit gewichtet. Anschließend wird pro Dokument der Durchschnitt der gewichteten Bewertungen gebildet. So wird ein Vektor mit Bewertungsprognosen für alle Dokumente erzeugt. Die Dokumente mit den höchsten Werten werden dann dem Benutzer vorgelegt.

lernen, um die Empfehlungen des System noch besser einschätzen zu können. Die Benutzer vertrauen anscheinend ungern einem gesichtslosen, anonymen Filtersystem.

Kapitel 3

Agentensysteme

Das *PZ*-System ist – wie bereits erwähnt – in zweifacher Hinsicht ein Agentensystem: zum einen stellt es sich dem Benutzer als Interface- oder Informationsagent dar, andererseits werden die Zeitungsartikel von einem Multiagentensystem mit verschieden spezialisierten Agenten gefiltert. Aber was genau ist ein Agent? Der nächste Abschnitt stellt verschiedene Definitionen aus unterschiedlichen Bereichen vor und nennt Schlüsselaspekte des Begriffs „Agent". Abschnitt 3.2 zählt Eigenschaften von Agenten auf und erläutert sie ausführlich. Zwei gegensätzliche Ansätze zur Modellierung von Agenten werden in Abschnitt 3.3 beschrieben. Die Vorstellung einiger Agentenarten in Abschnitt 3.4 schließt das Kapitel ab.

3.1 Definition von Agenten

Die Frage „Was ist ein Agent?" ist ähnlich schwer zu beantworten wie die Kardinalfrage der Künstlichen Intelligenz (KI) „Was ist Intelligenz?". Im Terminus „Intelligente Agenten" werden dieses beiden Wörter zudem eng zusammen gebraucht. Bei der Klärung der ersten Frage stößt man auf einige Schwierigkeiten. Forscher und Firmen bezeichnen heute höchst unterschiedliche Systeme mit dem Wort „Agent". Auch im täglichen (englischen) Sprachgebrauch wird das Wort häufig mit verschiedenen Bedeutungen benutzt. Im Collins English Dictionary zum Beispiel wird „agent" folgendermaßen definiert:

agent (n). -

1. A person who acts on behalf of another person, group, business, government etc.
2. A person or thing that acts or has the power to act.
3. A substance or organism that exterts some force or effect.
4. The means by which something occurs or is achieved.
5. A person representing a business concern (i.e. a travelling salesman).

Diese Definition enthält einige Aspekte des Begriffs „Agent“, wie er in der Informatik verwendet wird, würde aber auch fast jedes einfache Programm als Agent klassifizieren. Eine weitere Eingrenzung ist also notwendig und erwünscht. Es gab viele Versuche einer umfassenden Definition des Begriffs, bis jetzt ist aber noch keine allgemein anerkannt worden. Hermans kommt in seiner Doktorarbeit zu dem Schluß, daß eine formale Definition des Konzepts „Agent“ weder möglich noch praktikabel sei [Hermans, 1996].

Franklin und Graesser stellen verschiedenste Definitionen vor und geben selber eine weitere eigene Definition von autonomen Agenten [Franklin und Graeser, 1996]:

> „An *autonomous agent* is a system situated within and a part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future.“

Nach dieser Definition sind Menschen und viele Tiere Agenten, aber selbst Heizungsthermostate und Bakterien mit simpelsten Kontrollstrukturen fallen (von den Autoren gewollt) in ihren Geltungsbereich. Trotz ihrer großen Reichweite und der daraus resultierenden Unklarheit nennt die Definition aber wichtige Aspekte des Begriffs „Agent“ und macht so das Konzept etwas greifbarer:

- Zeitlich kontinuierliche Einbindung des Agenten in eine Umwelt
- Möglichkeiten zu Wahrnehmung der Umwelt
- Gezieltes Handeln
- Einwirkung auf die Umwelt
- Rückkopplung der Aktionen auf die Wahrnehmung

Besonders der erste und letzte Punkt grenzen normale Programme wie Textverarbeitungs- oder Lohnbuchhaltungsprogramme von Agenten ab. Obwohl diese ebenfalls ihre Umwelt wahrnehmen (z.B. die Tastatureingaben des Benutzers) und durch Ausgaben auf ihre Umwelt einwirken (z.B. Veranlassung von Lohnüberweisungen), werden diese Programme im Normalfall gezielt vom Anwender aufgerufen, für eine Aufgabe benutzt und dann beendet. Eine dauerhafte Einbindung in ihre Umwelt ist nicht gegeben. Die gewählten Aktionen wirken sich meistens nicht auf die zukünftige Wahrnehmung aus, so daß keine Rückkopplung entsteht.

Programme sind also nicht automatisch Agenten, ebenso sind Agenten nicht automatisch Programme. Autonome Roboter sind ein gutes Beispiel für künstliche Agenten. Im weiteren Verlauf dieser Arbeit werden allerdings nur *Softwareagenten* betrachtet, also Programme, die als Agent bezeichnet werden können.

Obwohl es keinen Konsens über eine Definition von „Agenten“ gibt, sind sich die meisten Forscher einig, daß Agenten bestimmte Eigenschaften besitzen (siehe Abschnitt 3.2). Wooldrigde und Jennings unterscheiden zudem noch zwischen einem anerkannten *schwachen Agentenbegriff*, der weitgehend den weiter unten vorgestellten essentiellen Eigenschaften entspricht, und einem darüber hinausgehenden, daher auch strittigeren, *starken Agentenbegriff* [Wooldridge und Jennings, 1995].

Letzterer definiert Agenten als Systeme, die nicht nur eben jene essentiellen Eigenschaften besitzen, sondern durch Konzepte entworfen und/oder implementiert wurden, die sonst eher Menschen zugeordnet werden. So können mentalistische Begriffe wie Wissen, Verpflichtung, Überzeugung, Wunsch und Absicht dazu benutzt werden, Agenten zu beschreiben. Rao und Georgeff weisen zum Beispiel nach, daß in bestimmten Anwendungsdomänen (meist Domänen, in denen Entscheiden und Handeln unter Ressourcenbeschränkungen nötig ist) Überzeugungen, Wünsche und Absichten (*beliefs, desires, intentions* = BDI) erforderlich sind, um das Verhalten eines Systems zu modellieren [Rao und Georgeff, 1995]. Diese drei Attribute repräsentieren

1. das unvollständige Wissen über den Zustand der Umwelt[1], von dem die möglichen Aktionen des Agenten abhängig sind,

2. die Prioritäten unter den verschiedenen, u.U. widersprüchlichen Zielen, die das System verfolgen soll,

3. den aktuell gewählten Handlungsplan, d.h. die Ausgabe einer Aktionsselektionsfunktion oder eines Planers[2].

Rao und Georgeff verwenden spezielle BDI-Logiken, die an modale Logiksysteme wie das KD45-System oder das S4-System anknüpfen. Sie geben außerdem die Grundschleife eines Interpreters für BDI-Systeme an. Die BDI-Architektur wurde von ihnen z.B. dazu benutzt, einen Agenten zum Luftverkehrsmanagement am Flughafen Sydney zu spezifizieren und zu implementieren. Weitere Anwendungen umfassen ein Space Shuttle Diagnosesystem, Telekommunikationsnetzmanagement und Luftkampfmodellierung.

Über die mentalistischen Begriffe hinaus werden Agenten auch Gefühle zugeschrieben [Bates, 1994]. Die Idee ist, daß der Charakter eines Agenten für einen

[1] Die Einschränkung, daß der Zustand der Umwelt vom Agenten nur lokal wahrgenommen werden kann, oder die Tatsache, daß die Umwelt sich nichtdeterministisch entwickelt, können die vollständige Modellierung der Umwelt in Form von Fakten unmöglich machen.

[2] Die Umgebung ändert sich ständig, also auch während der Zeit, in der Aktionen ausgewählt werden. Während der Ausführung des Handlungsplans ändert die Umwelt sich nicht nur durch die Aktionen, sondern auch unabhängig davon. Aufgrund dieses ständigen Wandels sind Pläne aufgrund veränderter Voraussetzungen nach der Auswahl nicht mehr optimal oder erreichen einmal selektierte Pläne bei ihrer Ausführung nicht das gesteckte Ziel. Es muß anhand des gewählten Handlungsplanes und der damit zu erreichenden Ziele die Balance zwischen keiner und ständiger Neuüberlegung gefunden werden, um trotzdem die gesetzten Ziele zu erreichen.

Benutzer durch Emotionen leichter zu erfassen ist und daher glaubhaft wird. Dazu muß der Gefühlszustand des Agenten immer klar definiert sein, die Gefühle müssen sich auf den Denk- bzw. Planungsprozeß des Agenten auswirken und dadurch in den Aktionen niederschlagen. Die Emotionen sollten durch den Agenten akzentuiert geäußert werden, so daß der Benutzer sie erkennen und richtig deuten kann.

Der starke Agentenbegriff sieht Agenten als *intentionale Systeme*. Der Begriff wurde von dem Philosophen Daniel Dennett geprägt und beschreibt Entitäten, „deren Verhalten durch die Zuweisung von Überzeugungen, Wünschen und rationaler Überlegung vorhersagbar wird“ (zitiert nach [Wooldridge und Jennings, 1995]). Die Methode der intentionalen Systeme bietet eine sehr mächtige Abstraktionsmöglichkeit für Agenten. Besonders nützlich ist die Zuweisung von mentalistischen Eigenschaften dann, wenn die Struktur der zu beschreibenden Systeme völlig unbekannt ist und daher eine mechanistische Beschreibung des Verhaltens etwa in Form eines Algorithmus nicht möglich ist. Wooldridge definiert folglich Agenten als Systeme, deren einfachste konsistente Beschreibung ein intentionales System ist. Damit löst er die Grenze zwischen starkem und schwachem Agentenbegriff wieder ein wenig auf, da auch „konventionell“ konzipierte Agenten sich u.U. einfacher mit mentalistischen Begriffen beschreiben ließen, dies nur von den Entwicklern unterlassen wurde.

3.2 Agenteneigenschaften

Wenn auch eine Definition von „Agenten“ bisher fehlt, gibt es doch einen Konsens über bestimmte Eigenschaften, die Systeme besitzen müssen, um als Agent bezeichnet werden zu dürfen. Nwana nennt als drei essentielle Eigenschaften [Nwana, 1996]:

- Autonomie
- Lernfähigkeit
- Kooperation

Von diesen müssen mindestens zwei in starker Ausprägung vorhanden sein, während die dritte schwächer vertreten sein kann.

Nwana leitet ausgehend von diesen drei essentiellen Eigenschaften die Agententypen Interface-Agenten, kollaborative Agenten, kollaborative lernende Agenten und *Smart Agents* ab (siehe Abbildung 3.1), wobei letztere das Ziel der Agentenforschung – Automomie, Lernfähigkeit und Kooperation jeweils auf hohem Niveau und in einem System vereint – darstellen sollen.

Weitere Eigenschaften, die in verschiedener Ausprägung bei Agenten vorkommen, aber nicht zwingend sind, lauten (siehe [Foner, 1993], [Franklin und Graeser, 1996] und [Wooldridge und Jennings, 1995]):

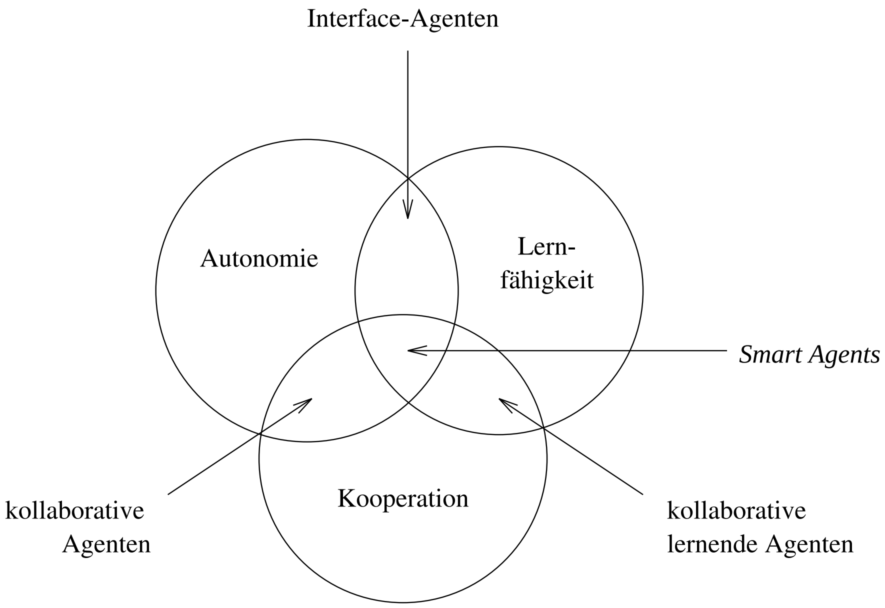


Abbildung 3.1: Eine Typologie von Agenten [Nwana, 1996].

- Mobilität
- Verläßlichkeit
- Charakter

Die folgenden Abschnitte erläutern diese Eigenschaften näher.

3.2.1 Essentielle Agenteneigenschaften

3.2.1.1 Autonomie

Agenten können ohne direkte Intervention von außen (z.B. von Menschen) agieren. Sie erfassen ihre Umwelt durch verschiedene Arten von Sensoren[3]. Veränderungen in der Umwelt werden so registriert. Agenten reagieren in angemessener Zeit auf diesen Wandel.

Benutzer oder andere Agenten werden als spezielle Teile der Umwelt betrachtet. Benutzer geben Oberziele (z.B. die Erledigung einer bestimmten Aufgabe) vor, während durch die Struktur des Agenten (z. B. die gegebenen Aktionsmöglichkeiten und Planungsalgorithmen) vom Programmierer die Basisziele

[3] Das Einsatzgebiet eines Agenten bestimmt, welche Art von Sensorik er besitzt. Bei einem Roboter in der physikalischen Welt können dies Ultraschallsensoren oder Videokameras sein, ein Agent in einem UNIX-Dateisystem [Song et al., 1996], [Etzioni und Weld, 1995] hat Kommandos wie `ls`, `du` oder `pwd` zur Verfügung, um seine Umgebung zu erspüren.

definiert sind. Zwischen diesen Polen wählen Agenten *eigene* Unterziele. Sie planen und kontrollieren selbständig Aktionen und sind nicht von Anleitung oder Überwachung durch einen Benutzer abhängig. Dadurch ergibt sich ein *zielgerichtetes Verhalten*. Bei der Verfolgung der Ziele handeln Agenten *vernünftig*, d.h. die Erfüllung der Ziele wird durch die Handlungen nicht unmöglich gemacht.

Besonders zeichnen sich Agenten dadurch aus, daß sie die *Initiative* ergreifen und nicht nur einfachen Reizschemata gehorchen. Vorausschauend erkennen sie, daß Handeln nötig ist, um die gesteckten Ziele zu erreichen. Mit dieser Eigenschaft unterscheiden sich Agenten von reinen *Servern* [Petrie, 1996]. Diese erzeugen nur auf eine Anfrage eines *Klienten* eine Antwort. Im *Client/Server-Protokoll* ist es dem Server nicht möglich, Nachrichten zu initiieren. Erst ein *Peer-to-Peer-Protokoll* mit gleichberechtigten Partnern erlaubt diese Art der Kommunikation. Datenbanksysteme, elektronische Bibliothekskataloge oder die meisten momentanen WWW-Suchmaschinen wie Altavista [Altavista, 1997] oder Lycos [Lycos, 1997] sind Beispiele für reine Server.

3.2.1.2 Lernfähigkeit

Eine wichtige Komponente von Intelligenz ist Lernen, also die Fähigkeit, vergangene Erfahrungen zu organisieren und nutzbar zu machen [Morik, 1993, S. 248-249]. Agenten zeigen intelligentes Verhalten, indem sie sich Veränderungen in ihrer Umgebung anpassen. Sie verfolgen ihre Interaktion mit der Umwelt und versuchen, ihre Zielerfüllung nach Analyse dieser Interaktion zu optimieren.

Besonders interessant ist die Lernfähigkeit bei der Anpassung an einen Benutzer. Diese kann auch durch explizite Kodierung der spezifischen Aufgaben im Agenten geschehen, wünschenswert ist aber eher eine Möglichkeit, dem System zu zeigen, wie eine Aufgabe erfüllt werden soll, z.B. durch *Programming by Demonstration* (vgl. [Cypher, 1993]). Noch weiter geht der Ansatz des *Learning Apprentice* [Mitchell et al., 1994]. Dieser lernt, indem er den Benutzer bei der Erledigung der Aufgabe „über die Schulter schaut“ und daraus ableitet, wie der Benutzer vorgeht und welche Ziele er verfolgt. Nach und nach kann der Lehrling dem Benutzer die Aufgabe abnehmen und selber durchführen.

3.2.1.3 Kooperation

Agenten arbeiten bei der Verfolgung ihre Ziele mit anderen Agenten zusammen. Dies können andere Softwareagenten, aber auch menschliche Benutzer sein. Gute Zusammenarbeit erfordert Kommunikation, Koordination von mehreren Agenten ist jedoch bis zu einem gewissen Grad ohne direkten Informationsaustausch möglich [Sen et al., 1994].

Der Diskurs der zusammenarbeitenden Agenten beginnt mit der Offenlegung ihrer jeweiligen Absichten und Fähigkeiten und endet in einer Einigung über eine Art Vertrag, wer welche Aufgabe erfüllt. Die Komplexität einer solchen Interaktion kann von einem einfachen Frage-Antwort-Paar bis zu einem fortgesetzten, komplexen Austausch von Information zwischen gleichberechtigten Partnern reichen. Für die Kommunikation zwischen Agenten gibt es spezialisierte Sprachen.

ACL (Agent Communication Language) ist eine solche Sprache, deren Syntax und Semantik unabhängig von einzelnen Agenten festgelegt ist [Genesereth und Ketchpel, 1994]. ACL besteht aus drei Teilen:

1. Ein offenes Vokabular ermöglicht verschiedene Beschreibungen für unterschiedliche Anwendungsbereiche.

2. KIF (Knowledge Interchange Format) ist eine Präfixversion der Prädikatenlogik erster Stufe mit einigen Erweiterungen. Es erlaubt die Kodierung von Inhalten wie Daten, Constraints, Regeln, quantifizierte Ausdrücke, Prozeduren und Wissen über Wissen.

3. KQML (Knowledge Query and Manipulation Language) stellt eine sprachliche Schicht zum Ausdruck von Kontext bereit. Nachrichten in KQML setzen sich aus einen Kommunikationstyp und dessen Argumenten zusammen und sind immer Teil eines Dialogs zwischen Sender und Empfänger. Die unterstützten Kommunikationstypen beziehen ihre Motivation aus der Sprechakttheorie und umfassen u.a. Benachrichtigungen, konditionale und verzögerte Kommandos, Anfragen, Angebote und Versprechen.

Ein Agent kann *wohlwollend* oder *konkurrierend* mit anderen zusammenarbeiten. Im ersten Fall ist er bereit, anderen Agenten beim Erreichen ihrer Ziele zu helfen, im zweiten berücksichtigt er nur die eigenen Ziele bei der Auswahl von Aktionen. In einigen Situationen (z.B. bei Nullsummenspielen) beeinträchtigt die Zielerfüllung des einen Agenten den Erfolg der anderen, in bestimmten Konstellationen (z.B. im sogenannten Gefangenen-Dilemma) ist dagegen die wohlwollende, uneigennützige Kooperation aller Beteiligten notwendig, um eine optimale Gesamtlösung zu erhalten.

3.2.2 Weitere Agenteneigenschaften

Mobilität: Einige (Software-)Agenten[4] besitzen die Fähigkeit, sich in einem Netzwerk von einem Rechner zum anderen zu bewegen. Ein Vorteil der Mobilität ist die Reduktion von Kommunikationskosten. Falls für eine Aufgabe aus einer großen Menge von Daten eine kleine Menge herausgesucht werden muß (z.B. Suche nach einer Flugverbindung in einem Reservierungssystem), ist es sinnvoll, diese Verarbeitung am Datenspeicher direkt durchzuführen, anstatt alle Daten über das Netz zu transportieren und lokal zu verarbeiten. Auch wenn es erforderlich ist, daß viele Agenten intensiv miteinander kommunizieren (z.B. bei einer

[4] Die Fähigkeit von Robotern, ihren Standort zu verlagern, soll hier nicht weiter behandelt werden.

Tauschbörse, bei der Agenten passende Abnehmer für ihre Güter suchen), sollten diese sich an *einem* Platz treffen, um die Kommunikation zu vereinfachen.

Grundlage für Mobilität sind standardisierte Sprachen und Plattformen wie Java und Telescript, die Möglichkeiten bieten, Programme rechnerunabhängig zu schreiben und auszuführen. Dabei müssen Aspekte der Sicherheit, wie Authentifizierung, Geheimhaltung und der Schutz der Privatsphäre, beachtet werden.

Verläßlichkeit: Agenten übernehmen bestimmte Aufgaben für Benutzer. Delegation beinhaltet immer das Risiko, daß die Aufgabe nicht genau nach Wunsch ausgeführt wird. Der Benutzer muß dem Agenten das *Vertrauen* entgegenbringen, daß seine Ziele bestmöglichst erfüllt werden. Vertrauen kann dadurch gebildet werden, daß der Agent seine Handlungen erklären kann und diese so für den Benutzer nachvollziehbar und vorhersagbar werden. Abrupte Verhaltensänderungen zerstören die Erwartungen des Benutzers. Beim Auftreten von Fehlern zum Beispiel sollte der Agent also nicht völlig versagen, sondern *robust* trotz der Fehler weiterarbeiten und *zuverlässig* diejenigen Teile der Aufgabe vollenden, die von den Fehlern unbeeinträchtigt bleiben. Robustheit und Zuverlässigkeit werden in der Softwaretechnologie als Qualitäten für jede Art von Computerprogrammen gefordert. Agenten sollten darüber hinaus nicht bewußt falsche Informationen weitergeben, sondern immer *wahrheitsgemäß* auf Anfragen antworten.

Charakter: Ein Agent kann in seinem Verhalten eine erkennbare Persönlichkeit[5] zeigen, die seinen Zustand widerspiegelt. Dies erleichtert die Interaktion mit dem Benutzer, der Erwartungen über Handlungen des Agenten aufbauen und so zukünftige Aktionen vorhersagen kann. Damit steigt das Vertrauen in den Agenten. Anthropomorphe Agenten (z.B. Julia [Foner, 1993]) benutzen aus der menschlichen Kommunikation vertraute Mechanismen, um eine Persönlichkeit zu präsentieren.

3.3 Deliberative und reaktive Agentenarchitekturen

Eine *Agentenarchitektur* ist eine Methodik, konkrete Systeme zu konstruieren, die Eigenschaften von Agenten besitzen. Dazu spezifiziert die Architektur eine Zerlegung des Agenten in einzelne Module und beschreibt, wie diese Module miteinander interagieren. Im Idealfall schließt eine Agentenarchitektur Techniken und Algorithmen zur Unterstützung der Methodik ein [Maes, 1991].

[5] Die Persönlichkeiten von bestehenden Agentensystemen haben nicht annähernd die Komplexität derjenigen eines Menschen oder eines Tieres. Deswegen mag der Gebrauch der Begriffe *Persönlichkeit* und *Charakter* übertrieben erscheinen. Andererseits neigen auch heute schon viele Anwender dazu, bestehenden Programme oder ihrem ganzen Rechner ein gewisses Eigenleben zuzuschreiben oder sogar mit ihnen zu sprechen.

Wie oben beschrieben handeln Agenten unabhängig und reagieren auf Veränderungen in ihrer Umgebung. Eine Möglichkeit dazu ist die Anpassung einer internen, expliziten symbolischen Repräsentation der Umwelt und die anschließende Erstellung neuer Pläne zur Erreichung der Ziele. Dieser Ansatz leitet sich aus der klassischen KI-Forschung (z.B. Wissensrepräsentation, Planung und Maschinelles Lernen) her. Die grundlegenden Annahmen hinter diesem Denkansatz hat Newell mit der *physical symbol system hypothesis* zu formulieren versucht [Newell, 1980]. Symbolsysteme bestehen aus Symbolen und einem Satz von Regeln, mit denen diese Symbole zu komplexen Strukturen kombiniert werden können. Symbole stehen für Dinge aus der Welt und haben so eine Bedeutung. Wenn Symbole wieder physikalisch (z.B. als Ladungen in RAM-Bausteinen oder Aktivitätsmuster von Neuronen) codiert werden, ergibt sich ein physikalisches Symbolsystem. Kognitive Prozesse bestehen aus der Manipulation von Symbolstrukturen. Der Kernpunkt der *physical symbol system hypothesis* ist, daß ein solches System unabhängig von der Materie, in der Symbole codiert werden, notwendige Voraussetzung für intelligentes Verhalten ist. Obwohl diese These nie unumstritten war (z.B. in der Debatte um den *Konnektionismus*), steckt ihre praktische Umsetzung hinter vielen Erfolgen der KI-Forschung.

Deliberative Agentenarchitekturen benutzen das symbolorientierte Paradigma zur Konstruktion von Agenten. Typische Module in einer solchen Architektur sind Wahrnehmung, Modellbildung, Planung, Ausführung und Kontrolle, die in dieser sequentiellen Reihenfolge, aber auch kaskadiert oder hierarchisch (in diesem Fall zusätzlich mit einem Steuerungsmodul) angeordnet sein können. Eine weitere Möglichkeit der parallelen Anordnung der Module ist die Einrichtung einer zentralen, gemeinsamen Datenstruktur (*Blackboard* [Corkill, 1991]), auf die alle Module Zugriff haben.

Der Internet Softbot von Etzioni und Weld ist ein Beispiel für ein System mit einer deliberativen Agentenarchitektur [Etzioni und Weld, 1994], [Etzioni und Weld, 1995]. Der Softbot dient seinem Benutzer als eine Art „Concierge" oder Sekretär zu den vielfältigen Diensten des Internets. Der Benutzer spezifiziert, „*Was*" getan werden soll und überläßt das „*Wie*" dem Agenten. Der Softbot nimmt komplexe, unvollständige Anfragen und Kommandos wie „Sende die neuen Berichte an Mitchell an der CMU!" entgegen, löst Mehrdeutigkeiten selbständig unter Rückgriff auf geeignete Datenbanken oder durch Rückfrage beim Benutzer auf, plant geeignete Aktionen und führt diese anschließend durch. Zentrale Komponente der Softbot-Architektur ist der XII Planer, der auch mit unvollständiger Information umgehen kann, indem er die Informationsbeschaffung durch Ausführung von Aktionen in den Planungsprozeß integriert. Dem Planer zur Seite steht der Modellmanager, der alle Beobachtungen des Softbots in eine für den Planer nutzbare symbolische Repräsentation wandelt und für zukünftige Zwecke speichert. In Internet Domain Models ist statisch das Wissen über die Verwendung der zur Verfügung stehenden Sensoren und Effektoren abgelegt. Zusätzlich beinhalten sie Informationen über Personen und Datenbanken. Ein

Scheduler kontrolliert sowohl die kognitiven (z.B. Planung) als auch die aktiven (z.B. Versenden von E-Mails) Prozesse des Softbots.

Im Gegensatz zur traditionellen KI-Forschung steht die Richtung der *Verhaltensbasierten Künstlichen Intelligenz* (behaviour-based AI), die an vorderster Stelle von Rodney Brooks vertreten wird. Brooks zentrale Thesen lassen sich folgendermaßen zusammengefassen [Wooldridge und Jennings, 1995]:

1. Intelligentes Verhalten kann *ohne* explizite Repräsentation (wie in der klassischen Künstlichen Intelligenz) erzeugt werden.

2. Intelligentes Verhalten kann *ohne* explizites Folgern und Schließen (wie in der klassischen Künstlichen Intelligenz) erzeugt werden.

3. Intelligenz ist eine emergente Eigenschaft bestimmter komplexer Systeme.

Brooks geht dabei davon aus, daß Intelligenz nur in körperlichen und in einer Umgebung situierten Systemen existiert, erst in der Interaktion mit dieser Umwelt entsteht und keine eigene, isolierte Eigenschaft ist (siehe auch [Maes, 1994b]). Durch die Körperlichkeit und die Situiertheit des Systems verringert sich der Zwang zu einer Repräsentation der Welt, die Welt selbst wird als ihr eigenes Modell benutzt, das mit Sensoren und Effektoren gespürt und verändert werden kann.

Die Anwendung dieser Ideen auf die Modellierung von Agenten führt zu *reaktiven Agentenarchitekturen*. Brooks persönlich entwickelte die *Subsumtionsarchitektur*, die aus einer Hierarchie von übereinander geschichteten Verhaltensmustern (competencies) besteht [Brooks, 1986]. Jede Schicht hat Zugriff auf alle Sensoren und Effektoren, höhere Schichten können aber Wahrnehmung (d.h Eingabe von Sensoren) und Aktionen (d.h. die Benutzung von Effektoren) niedrigerer Schichten unterdrücken. Die Verhaltensmuster sind ansonsten unabhängig, neue Schichten sollen ohne Änderung der vorhandenen hinzugefügt werden können. Innerhalb der einzelnen Schichten werden meist nur sehr einfache Verfahren gebraucht, um die Wahrnehmung in Aktion umzusetzen. Trotzdem entsteht ein Gesamtverhalten des Systems, das als „intelligent“ bezeichnet werden kann.

SUMPY ist ein Agent, der nach der Subsumtionsarchitektur aufgebaut ist [Song et al., 1996]. Der Agent soll ein UNIX-Dateisystem pflegen, indem er selten benutzte Dateien komprimiert und regelmäßige Bandsicherungen durchführt. Dabei soll er nur zu Zeiten geringer Prozessornutzung arbeiten. SUMPY besteht aus vier Schichten. Die unterste Schicht läßt den Agenten zufällig im Verzeichnisbaum herumwandern. Die darüberliegende Schicht stoppt das Herumwandern, sobald ein neues Verzeichnis betreten wird. Dann wird mit einer Fuzzy-Entscheidungsfunktion festgestellt, welche Dateien komprimiert werden sollen. Die dritte Schicht sperrt die Kompression, falls im Verzeichnis Dateien längere Zeit nicht mehr gesichert wurden, und sichert diese Dateien. Die oberste Schicht prüft, wie hoch die Prozessorbelastung ist. Falls die Belastung über einem zuvor festgelegten Schwellwert liegt, werden Sicherung, Kompression und Wandern unterdrückt, bis die CPU-Last wieder gesunken ist. Keine der Schichten kennt interne Vorgänge der anderen, die Zusammenarbeit erfolgt durch die Unterdrückung von Sensoreingaben und Sperrung von Aktionsausgaben. Neue Aufgaben sollen durch das Plazieren neuer Schichten in die Hierachie einfach hinzugefügt werden können. Fehler in einer Schicht hindern andere Schichten nicht an der Erfüllung ihrer Aufgabe.

Die beiden oben beschriebenen unterschiedlichen Paradigmen werden neuerdings auch in hybriden Systemen vereint. Man versucht, jeweils die Nachteile des einen mit den Vorteilen des anderen auszugleichen. Reaktive *low-level* Komponenten mit direkter Koppelung von Wahrnehmung zu Aktion ermöglichen diesen Systemen eine schnelle Reaktion auf Veränderungen in der Umwelt, ohne erst komplexes Folgern zu benutzen. Deliberative *high-order* Komponenten mit einem expliziten Weltmodell erzeugen Pläne, treffen Entscheidungen und dienen so der Realisierung der Langzeitziele des Systems. Das Zusammenspiel der beiden Komponenten muß über eine geeignete Kontrollstruktur geregelt werden.

TouringMachines sind ein Beispiel für ein hybrides System [Ferguson, 1992]. Zusätzlich zu einem Wahrnehmungs- und einem Aktionssubsystem, die Ein- und Ausgabe des Systems darstellen, bestehen TouringMachines aus 3 Schichten, zwischen denen ein Kontrollsubsystem vermittelt und mit Kontrollregeln Konflikte zwischen Aktionsvorschlägen der verschiedenen Schichten auflöst. Die *Reaktionsschicht* ist eine Menge von Situation-Aktion-Regeln wie in Brooks Subsumtionsarchitektur. Die *Planungsschicht* besteht aus einem Planer und einem Filter für planungsirrelevante Informationen. Die Zustände von anderen Objekten (z.B. anderen Agenten) in der Umwelt werden von einer *Modellierungsschicht* symbolisch repräsentiert und dazu benutzt, Konflikte bei der Zielerreichung zu erkennen und aufzulösen. Die Schichten sind horizontal angeordnet, d.h. alle haben Zugriff auf dieselben Ein- und Ausgabekanäle. Auch vertikale Anordnungen sind möglich, z.B. im InteRRaP-System [Müller et al., 1995].

3.4 Betrachtung einiger Agentenarten

Der Begriff „Agent“ wird oft nicht alleinstehend verwendet, sondern durch zusätzliche Begriffe spezialisiert. So wurden *Softwareagenten* weiter oben schon als Computerprogramme, die die essentiellen Eigenschaften von Agenten besitzen, eingeführt. Teilweise wird durch das zusätzliche Wort eine Agenteneigenschaft besonders hervorgehoben (wie bei *Autonome Agenten* [Franklin und Graeser, 1996] oder *Mobile Agenten*), andere Zusammensetzungen stellen die Architektur bzw. den Aufbau von Agenten in den Vordergrund (z.B. *reaktive Agenten* oder *Multiagentensystem*). Eine weitere Möglichkeit ist die Betonung des Aufgabenbereichs oder der Umgebung, in denen der Agent eingesetzt wird. Beispiele hierfür sind *Interface-Agenten*, *Suchagenten* oder *Internetagenten*.

Im folgenden werden nur die Agentenarten *Multiagentensystem* und *Interface-Agenten* näher erläutert, da das *PZ*-System diesen beiden Begriffen zugeordnet werden kann. Weitere Agentenarten werden z.B. von Nwana beschrieben [Nwana, 1996].

3.4.1 Multiagentensysteme

Aus der Verbindung der Forschungsfelder „Künstliche Intelligenz“ und „Verteilte Systeme“ entstand in den späten siebziger Jahren das Unterfeld *„Verteilte Künstliche Intelligenz“* (DAI - Distributed Artificial Intelligence). Die Forschung dort verteilt sich auf drei Stränge: *Distributed Problem Solving*, *Parallel AI* und *Multiagentensysteme* (MAS). Hauptthema der DAI ist Informationsmanagement, also die Dekomposition von Aufgaben in Teilaufgaben und die anschließende Synthese einer Gesamtlösung aus den Teillösungen. Dazu werden starke Festlegungen für die Zusammenarbeit der einzelnen Problemlösekomponenten getroffen. Der Schwerpunkt der Multiagentensystemforschung liegt im Gegensatz dazu auf dem Management des Verhaltens der Agenten untereinander. Die individuellen Verhaltensweisen einzelner Einheiten (der Agenten) sind einfach gegenüber der Interaktion zwischen diesen Agenten.

Es gibt verschiedene Gründe für den Einsatz von Multiagentensystemen (siehe [Stone und Veloso, 1996]). In Anwendungsfällen, in denen verschiedene Organisationen zusammenarbeiten müssen, ihre Interna aber nicht nach außen gelangen lassen wollen, ist eine Modellierung über verschiedene, verkapselte Agenten die einzige Alternative. Die Verteilung der Aufgabe ermöglicht eine schnellere Lösung durch Parallelisierung. Falls Steuerung und Verantwortung ebenfalls verteilt werden, kann eine größere Robustheit erreicht werden, da Fehler eines Agenten nicht mehr den Ausfall des ganzen Systems bedeuten. Die Modularisierung durch Agenten erlaubt es, ein bestehendes System relativ einfach mit neuen Fähigkeiten auszustatten, indem neue Agenten eingefügt werden. Dies macht Multiagentensysteme gegenüber monolithischen Systemen besser skalierbar. Nicht zuletzt können Multiagentensystem zur Untersuchung von Intelligenz dienen. Menschliche Intelligenz ist nach einigen Theorien durch die Notwendigkeit der sozialen Interaktion mit anderen Menschen entstanden. Diese Interaktion kann mit Multiagentensystemen modelliert und untersucht werden.

Multiagentensysteme können nach dem Grad der Kommunikation zwischen einzelnen Agenten und der Unterschiedlichkeit der Agenten klassifiziert werden [Stone und Veloso, 1996]:

Homogene, nicht kommunizierende Agenten: Alle Agenten haben dieselbe interne Struktur, d.h. ihre Ziele und ihr Wissen sind ursprünglich gleich. Erst durch die verschiedene Einbeziehung in die Umwelt, die unterschiedliche Sensordaten zur Folge hat, beginnen sich individuelle Verhaltensweisen der Agenten zu entwickeln. Diese Ausgangsbasis ermöglicht dem einzelnen

Agenten die Voraussage der Handlungen der anderen Agenten ohne die Modellierung ihrer Struktur, da er selber ein Modell für die anderen ist. Agenten können sich durch Veränderungen der Umgebung, die mit Sensoren gespürt werden können (z.B. das Hinterlassen von Marken) oder die die Ergebnisse der Handlungen anderer Agenten verändern, auch ohne direkte Kommunikation gegenseitig beinflussen.

Heterogene, nicht kommunizierende Agenten: In diesem Fall unterscheiden sich die Agenten schon von Beginn an. Nicht nur ihr Platz in der Welt ist verschieden voneinander, auch ihr Wissen, ihre Ziele und die ihnen zur Verfügung stehenden Handlungen sind nicht gleich. Zur Vorhersage der Aktionen anderer Agenten müssen diese unterschiedlichen Eigenschaften von jedem Agenten nun modelliert werden. Dazu beobachten sich die Agenten gegenseitig und bauen so Modelle voneinander auf. Trotz der weiterhin fehlenden Kommunikation können die Agenten über sich entwickelnde soziale Konventionen (z.B. die häufigsten Aktionen in der Vergangenheit) kooperieren.

Heterogene, kommunizierende Agenten: Durch das Hinzufügen der Fähigkeit zur Kommunikation wird die Komplexität eines Multiagentensystems weiter erhöht, gleichzeitig steigt aber auch die Fähigkeit, Probleme zu lösen. Wissen und Ziele können nur zwischen Agenten ausgetauscht werden, wenn ein gemeinsames Kommunikationsprotokoll definiert ist. Kommunikation kann als eine spezielle Art der Handlung (Sprechakt) gesehen werden und muß dementsprechend geplant werden. Ein Agent kann dann in Absprache mit anderen Agenten eine gewisse *Rolle* im Agententeam übernehmen.

3.4.2 Interface-Agenten

Die zunehmende Komplexität von Computersystemen und ihrer Benutzungsschnittstellen zeigt die Schwächen der momentan vorherrschenden Interaktionsmetapher „direkte Manipulation“[6]. Interface-Agenten dagegen arbeiten mit dem Benutzer zusammen in derselben Arbeitsumgebung [Maes, 1994a]. Ihre Rolle ist die eines autonomen *persönlichen Assistenten*, der die Schwierigkeit einer Aufgabe verbirgt, den Benutzer bei der Erledigung von Aufgaben unterstützt, Aufgaben im Auftrag des Benutzers erfüllt oder auch Hilfe oder Training zu bestimmten Themen bietet. Die Metapher für diese Art der Interaktion ist „indirekte Manipulation“.

[6] Der Benutzer initiiert jede Aufgabe explizit und überwacht die Durchführung. „Direkte Manipulation“ erfordert vom Benutzer viel Wissen über die Ausführung der Aufgabe mittels eines Computersystems, Problemlösewissen in der eigentlichen Domäne steht erst an zweiter Stelle.

Voraussetzungen für den Einsatz von Interface-Agenten sind das Vorhandensein von sich wiederholendem Verhalten bei der Benutzung des Systems und Unterschiede bei der Erledigung der Aufgabe zwischen verschiedenen Benutzern. Wird die Aufgabe von allen Benutzern gleich bewältigt, lohnt es sich, von Experten eine Wissensbasis zur Lösung der Aufgabe erstellen zu lassen. Sind keine Regelmäßigkeiten im Benutzerverhalten gegeben, kann ein lernender Agent kein Modell für das Verhalten aufbauen. Zur Anpassung des Interface-Agenten an einen Benutzer kann außer den in Punkt 3.2.1.2 schon erwähnten Methoden auch der Rat von Agenten anderer Benutzer verwendet werden.

Es gibt verschiedene Untergruppen von Interface-Agenten (siehe [Nwana, 1996] und [Beale und Wood, 1994]):

Sekretäre: Im täglichen Arbeitsablauf tauchen immer wieder sich ähnelnde Aufgaben auf, die an einen Sekretär delegiert werden können. Agenten ersparen dem Benutzer die Erledigung von Routineaufgaben und verbergen komplizierte Vorgänge hinter einfachen Kommandos.

Systeme wie Calendar Agent [Maes und Kozierok, 1993] oder CAP [Mitchell et al., 1994] übernehmen das Management eines Terminkalenders für einen Benutzer. Diese Sekretäre interagieren mit Agenten anderer Benutzer, akzeptieren oder lehnen Anfragen nach den Prioritäten des Benutzers und der momentanen Auslastung des Zeitplans ab, handeln Zeit und Ort von Treffen aus und erinnern den Benutzer rechtzeitig an seine Termine. Beide Agentensysteme erkennen und erlernen durch Beobachtung die Vorlieben und Prioritäten des Benutzers. Dazu werden die Lernverfahren Memory-based Reasoning bzw. Induktion von Entscheidungsbäumen benützt.

Ein weiteres Beispiel für einen assistierenden Sekretäragenten ist der bereits vorgestellte Internet Softbot (siehe Seite 35).

Führer: Komplexe Systeme wie moderne Textverarbeitungsprogramme oder das World Wide Web (WWW) sind für unerfahrene Benutzer schwer zu überblicken. Hier bieten Interface-Agenten Hilfen. Sie „nehmen den Benutzer an die Hand“ und unterstützen ihn aktiv durch die Empfehlung von Aktionen. WebWatcher [Armstrong et al., 1995] [Joachims, 1996a] [Mladenic, 1996] ist ein „Fremdenführer“ (Tour Guide) für das WWW. Nach der Eingabe eines Suchinteresses durch den Benutzer markiert WebWatcher auf jeder WWW- Seite interessante Hyperlinks. Der Benutzer kann jetzt selber entscheiden, welchen Links er folgt. WebWatcher lernt, indem zu jedem Hypertextverweis eine Liste von Interessensbeschreibungen der Benutzer, die diesem Hyperlink gefolgt sind, gespeichert wird. Diese Beschreibung und der Anchor-Text des Verweises werden mit dem Suchinteresse des momentanen Benutzers verglichen. Bei hinreichender Ähnlichkeit (die mittels verschiedener Verfahren zur Textkategorisierung berechnet werden kann [Joachims, 1996a]) wird ein Hyperlink von WebWatcher vorgeschlagen. Reinforcement

Learning [Kaelbling et al., 1996] ist eine weitere Möglichkeit, die Erstellung optimaler (d.h. mit möglichst hohem Informationsgehalt) Touren durch das WWW zu lernen. Dazu wird die Struktur des Hypertexts hinter einem Verweis mitbetrachtet. Es werden dann diejenigen Hyperlinks markiert, die bei der Fortsetzung der Tour zu vielen relevanten Seiten führen.

Letizia [Liebermann, 1995] ist ein ähnlicher WWW-Führer, der mittels Breitensuche die Hypertexte hinter den Links der aktuellen WWW-Seite auf Ähnlichkeit mit dem Benutzerinteresse untersucht. Durch Beobachtung der Aktionen des Benutzers beim Browsen inferiert *Letizia* selbständig das Suchziel, ohne eine explizit eingegebene Beschreibung vorauszusetzen.

Systeme wie BASAR [Thomas und Fischer, 1996], BOTH [Angelaccio et al., 1996], LAW [Edwards et al., 1996], LIRA [Balabanovic und Shoham, 1995] und Syskill & Webert [Pazzani et al., 1996] versuchen nicht nur, den Benutzer zu begleiten, sondern suchen aktiv interessante Seiten im World Wide Web, die sie dem Benutzer vorschlagen. Dazu werden oft Suchanfragen an große Suchmaschinen wie Altavista [Altavista, 1997], Lycos [Lycos, 1997] oder HotBot [HotBot, 1997] gestellt, deren Ergebnisse mit Hilfe des vorhandenen Benutzerprofils ausgewertet werden.

Filter: Agenten schützen Benutzer vor der Informationsflut, die auf sie einströmt. Interessante und relevante Informationen gelangen bis zum Benutzer, Irrelevantes wird ausgefiltert. Filteragenten können den Eingabestrom auch in Kategorien oder nach Wichtigkeit sortieren und somit strukturieren.

In Abschnitt 5.4 werden einige Interface-Agenten vorgestellt, die intern als Multiagentensysteme realisiert sind.

Vermittler: Agenten können die Interessen ihres Benutzers in einem Markt vertreten. Außer der Fähigkeit, mit anderen Marktteilnehmern zu verhandeln, die auch Sekretäre besitzen, muß ein Vermittleragent passende Verhandlungspartner finden und erkennen können.

Yenta [Foner, 1996] versucht, verschiedene Benutzer mit gemeinsamen Interessen zusammenzubringen. Das Profil eines Benutzers wird anhand von Texten wie E-Mails oder Textdateien festgelegt. Ein Agent wird mit einer aus dem Profil extrahierten Beschreibung des Benutzerinteresses ausgestattet. Er kommuniziert nun mit anderen Agenten und vergleicht die von diesen vertretenen Interessen mit den eigenen. Falls sich zwei Agenten gleichen, bilden sie einen Cluster. Andernfalls tauschen die Agenten aus einem Cache von vorangegangenen Interaktionen Referenzen zu anderen Agenten aus, deren Interessensbeschreibung möglicherweise besser paßt. Durch diese Hinweise gelangen Agenten auf eine dem *Hill-climbing* ähnliche Weise schnell zu einem Partner, ohne mit allen Agenten im System kommunizieren zu müssen.

Kapitel 4

Die Benutzung des *PZ*-Systems

Der Einstieg in das *PZ*-System und die anschließende Benutzung soll in diesem Kapitel anhand eines fiktiven Szenarios gezeigt werden. Stellen wir uns einen Manager eines großen deutschen Konzerns vor. Sein Unternehmen produziert u.a. Software im Multimediabereich, hat Geschäftskontakte in ganz Europa und nach Südafrika und besitzt über Aktienkäufe eine Reihe von Beteiligungen an anderen Firmen. In seiner Freizeit spielt der Manager Tennis und verfolgt die Spiele der Fußballbundesliga. Für seine Arbeit ist er darauf angewiesen, möglichst viel Information über die allgemeine Lage auf den Finanzmärkten und die spezielle Entwicklung bei Firmen, von denen der Konzern Aktien hält, zu bekommen. Die politischen Ereignisse in Europa und Südafrika haben unter Umständen Einfluß auf die geschäftlichen Verbindungen in diese Regionen, deshalb möchte der Manager auch darüber informiert sein.

Bisher las der Manager eine einzige überregionale deutsche Tageszeitung, deren Wirtschaftsteil sehr umfangreich ist, die aber – nach Meinung des Managers – über Politik nur aus einer sehr eingeschränkten Perspektive berichtet[1]. Sein Interesse an Sportthemen wird ebenfalls kaum durch die Berichterstattung dieser Zeitung befriedigt. Das Fehlen bestimmter Informationen und die mangelnde Meinungsvielfalt versucht der Manager in Gesprächen unter Kollegen und durch die Benutzung anderer Informationsquellen (z.B. Fernsehen) zu kompensieren.

Der Manager könnte andererseits auch mehrere Tageszeitungen lesen und dadurch die Informationsdefizite seiner Stammzeitung ausgleichen. Doch dem Vorteil einer größeren Meinungsvielfalt (eine nicht zu einseitige Auswahl der zu lesenden Zeitungen vorausgesetzt) stünde der enorme Zeitaufwand entgegen. Jeden Morgen fünf oder zehn Tageszeitungen durchzublättern hält eine vielbeschäftigte Führungskraft zu lange von ihren sonstigen Aufgaben ab.

Mit Hilfe des *PZ*-Systems ist der Manager in der Lage, die Vorteile einer breiten und ausgewogenen Information aus mehreren Tageszeitungen zu nutzen, ohne die zeitlichen Nachteile in Kauf nehmen zu müssen. Zum Einstieg in das

[1] In diesem Szenario wird keine reale deutsche Zeitung beschrieben.

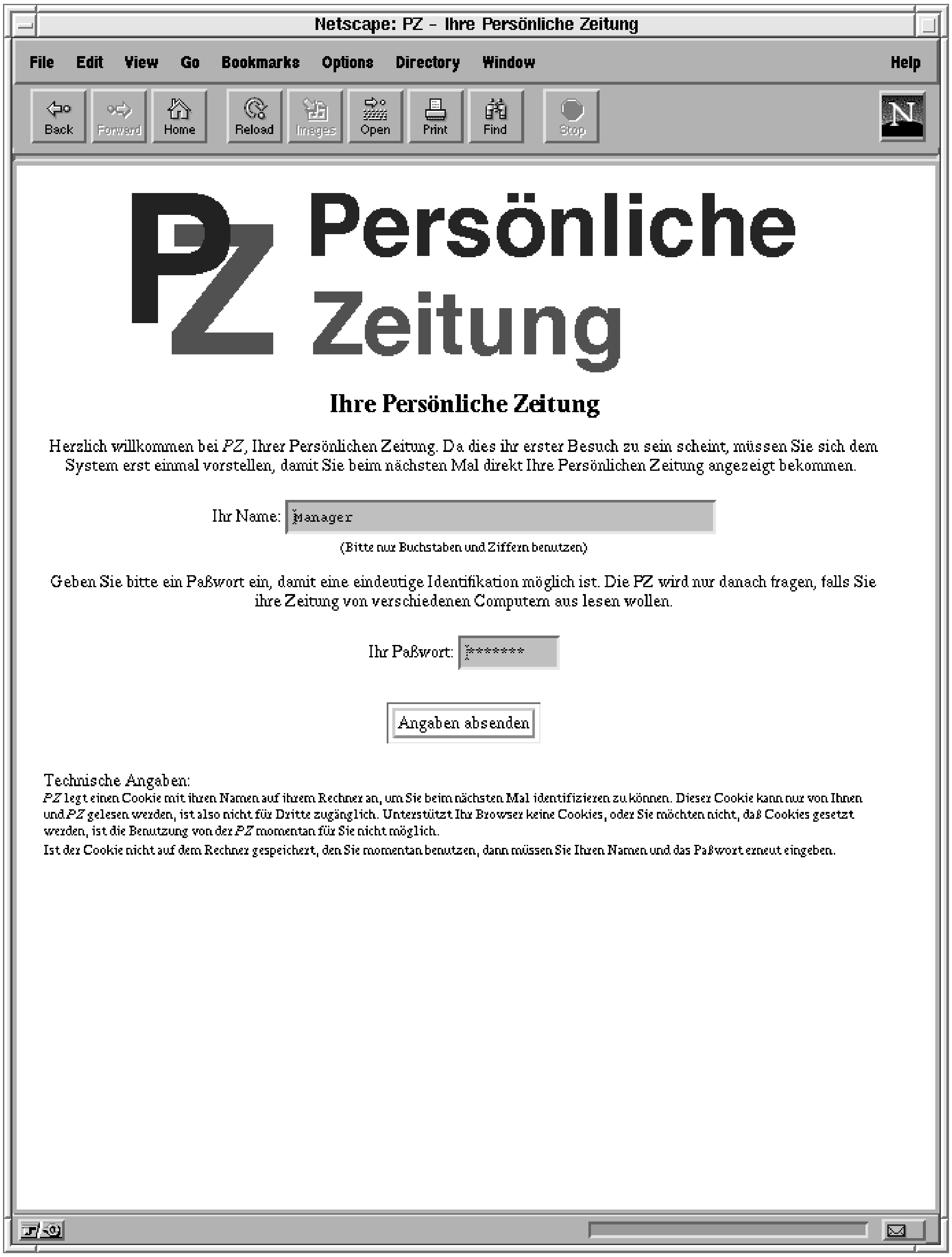


Abbildung 4.1: Eingangsseite des *PZ*-Systems.

PZ-System muß er sich zunächst auf der Eingangsseite (siehe Abbildung 4.1) mit Namen und Paßwort registrieren lassen. Nun wird mittels eines *Cookies* eine Art Schlüssel für das *PZ*-System auf dem Rechner des Managers angelegt. Dies ermöglicht die wiederholte Benutzung des Systems, ohne daß der Benutzer sich ständig identifizieren muß.

Anschließend erscheint die eigentliche Hauptseite des *PZ*-Systems, wie sie in Abbildung 4.2 zu sehen ist. Das *PZ*-System weiß zu diesem Zeitpunkt noch nichts über die Interessen des Managers und erstellt deshalb noch keinen persönlichen Pressespiegel. Artikel können zu diesem Zeitpunkt nur nach der allgemeinen Wichtigkeit ihres Themas ausgewählt werden. Die Annahme ist, daß die Anzahl der zu einem Thema erschienenen Artikel die Bedeutung widerspiegelt, die die Herausgeber der Zeitungen diesem Nachrichtenthema beimessen. In den anfänglichen Pressespiegel wird daher aus den fünfzehn Themengebieten mit der größten Artikelzahl je ein für das Thema repräsentativer Zeitungsartikel aufgenommen.

Der Pressespiegel wird in einem zweigeteilten Anzeigefeld dargestellt. Links ist eine Liste der Zeitungsressorts zu sehen, aus denen Artikel im Pressespiegel vorhanden sind. Das Ressort *TopNews* nimmt dabei eine Sonderstellung ein. Darin befinden sich zehn Artikel, von denen das System annimmt, daß sie für den Benutzer an diesem Tag die interessantesten sind. Die zehn Artikel können aus allen herkömmlichen Ressorts stammen und werden nur unter dem künstlichen Ressort TopNews zusammengefaßt. Dies entspricht praktisch der Titelseite einer konventionellen Zeitung, auf der die wichtigsten Meldungen des Tages versammelt sind. Darum erscheint im rechten Bereich anfangs immer eine Liste der Artikel des Ressorts TopNews. Durch das Anklicken eines Ressortnamens in der Liste links kann zur Anzeige der Artikel dieses Ressorts gewechselt werden.

Ein Artikel zum Multimedia-Kongreß in Stuttgart erregt die besondere Aufmerksamkeit des Managers, da seine Firma dort für ihre Produkte geworben hat. Er wählt den Hyperlink mit der Schlagzeile des Artikels in der Artikelliste rechts an und bekommt den gesamten Text des Artikels im rechten Bereich angezeigt. Abbildung 4.3 zeigt, daß dabei unter dem Nachrichtentext wieder die restlichen Artikel des Ressorts aufgelistet werden. Dies erlaubt dem Benutzer, schnell zu anderen Berichten zu wechseln, ohne zurück zur Ressortübersicht zu gehen.

Liest der Benutzer einen Artikel des Pressespiegels, so wertet das System dies als Hinweis, daß ihn das Thema des Artikels interessiert. Zusätzlich ist unterhalb der Artikelanzeige eine Leiste mit zwei Links „Das Thema des Artikels interessiert mich sehr!“ und „Diesen Artikel fand ich uninteressant!“ angeordnet. Durch den ersten Verweis kann der Manager dem *PZ*-System gezielt mitteilen, daß er in den nächsten Ausgaben des Pressespiegels weitere Artikel zum Thema Multimedia lesen möchte. Über den zweiten Hyperlink besteht für den Manager die Möglichkeit, sein Desinteresse am Thema des momentan angezeigten Artikels auszudrücken und das System aufzufordern, ähnliche Artikel nicht in zukünftige Pressespiegel aufzunehmen. Der Manager ist nur an für ihn wichtiger Information interessiert, deshalb nutzt er diese Ausschlußmöglichkeit im Falle des Artikels

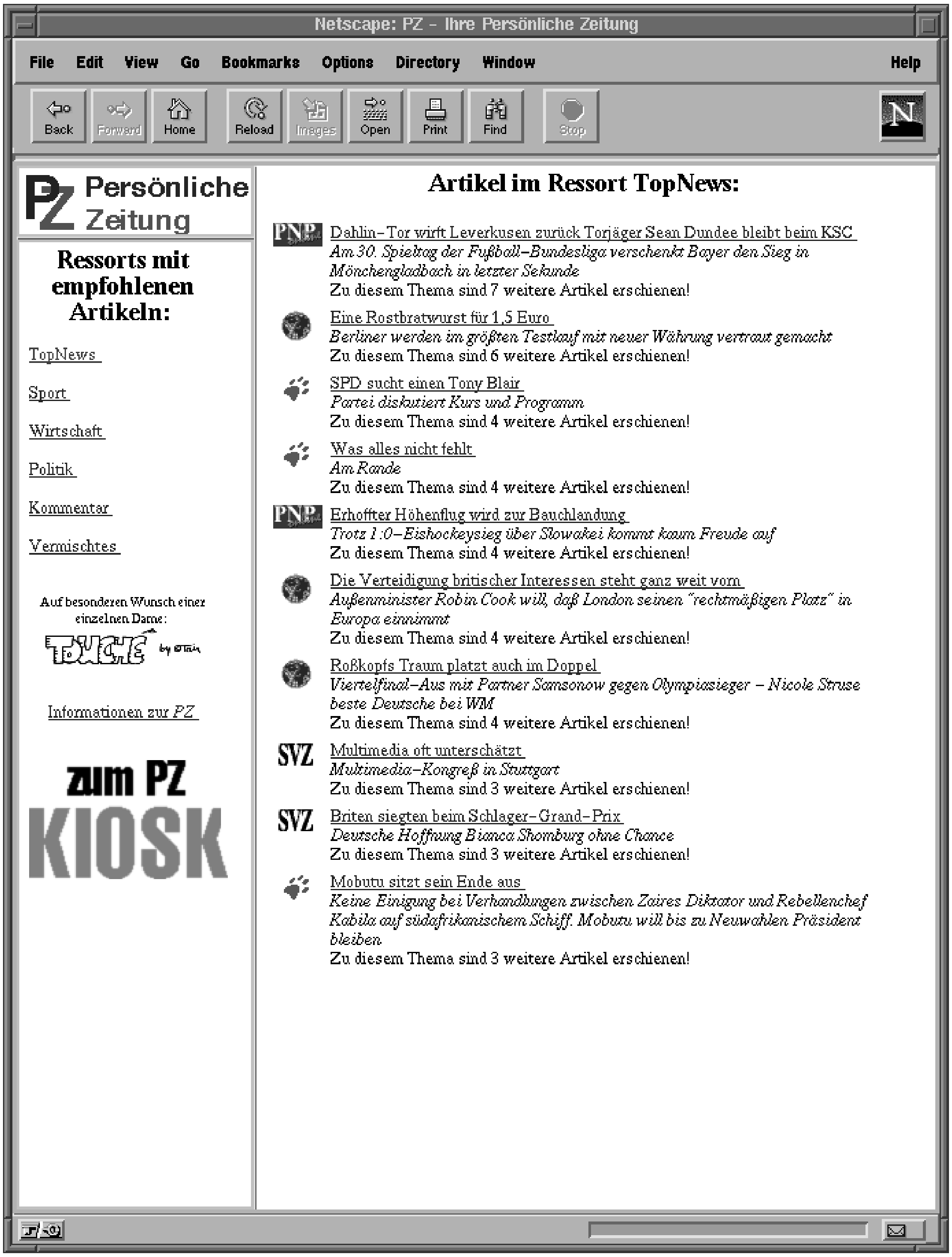
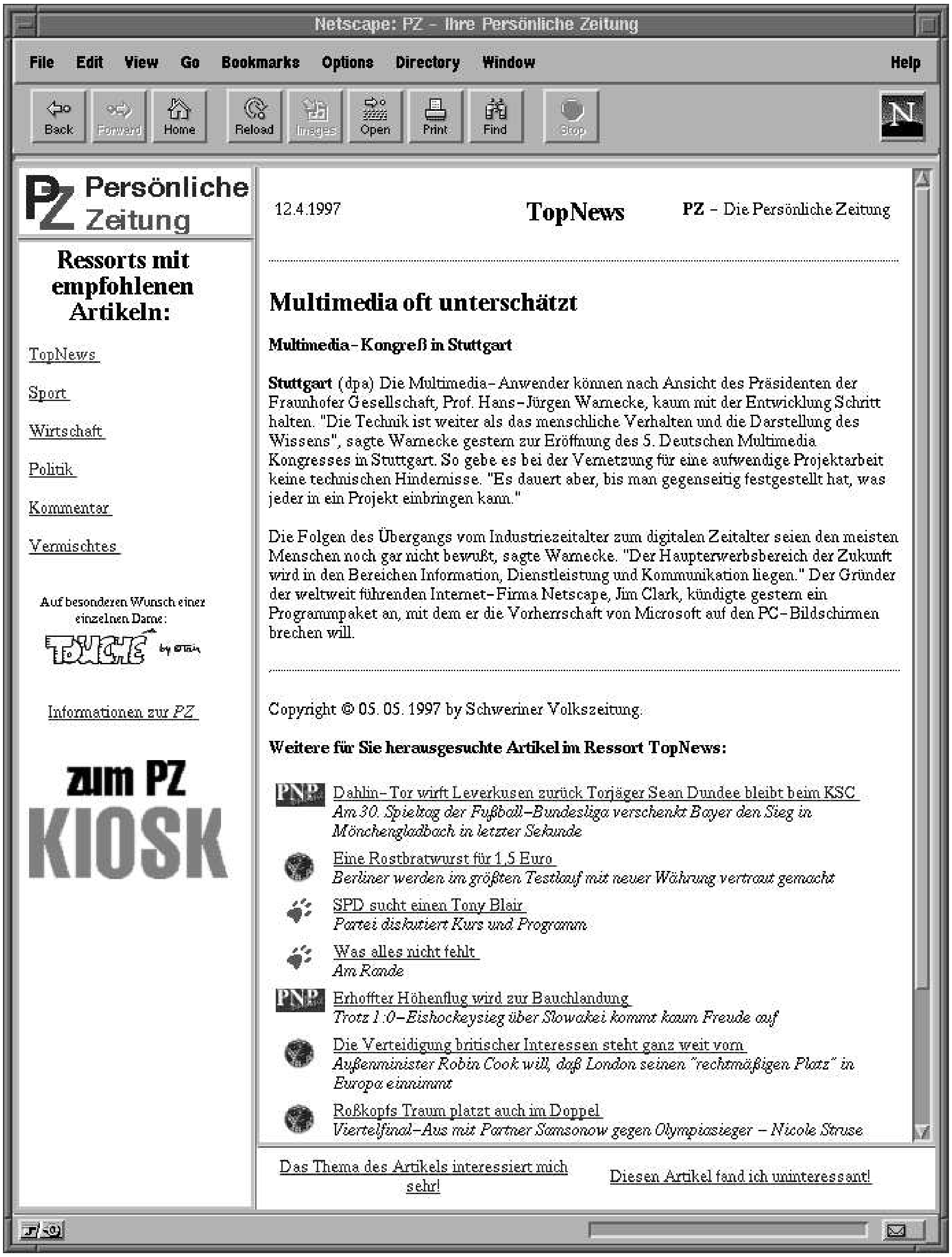


Abbildung 4.2: Hauptseite des *PZ*-Systems bei der ersten Benutzung.



Netscape: PZ - Ihre Persönliche Zeitung

File Edit View Go Bookmarks Options Directory Window Help

Back Forward Home Reload Images Open Print Find Stop

PZ Persönliche Zeitung

Ressorts mit empfohlenen Artikeln:

TopNews

Sport

Wirtschaft

Politik

Kommentar

Vermischtes

Auf besonderen Wunsch einer einzelnen Dame:

TOUCHE by ©TAH

Informationen zur *PZ*

zum PZ KIOSK

12.4.1997 **TopNews** **PZ** – Die Persönliche Zeitung

Multimedia oft unterschätzt

Multimedia-Kongreß in Stuttgart

Stuttgart (dpa) Die Multimedia-Anwender können nach Ansicht des Präsidenten der Fraunhofer Gesellschaft, Prof. Hans-Jürgen Warnecke, kaum mit der Entwicklung Schritt halten. "Die Technik ist weiter als das menschliche Verhalten und die Darstellung des Wissens", sagte Warnecke gestern zur Eröffnung des 5. Deutschen Multimedia Kongresses in Stuttgart. So gebe es bei der Vernetzung für eine aufwendige Projektarbeit keine technischen Hindernisse. "Es dauert aber, bis man gegenseitig festgestellt hat, was jeder in ein Projekt einbringen kann."

Die Folgen des Übergangs vom Industriezeitalter zum digitalen Zeitalter seien den meisten Menschen noch gar nicht bewußt, sagte Warnecke. "Der Haupterwerbsbereich der Zukunft wird in den Bereichen Information, Dienstleistung und Kommunikation liegen." Der Gründer der weltweit führenden Internet-Firma Netscape, Jim Clark, kündigte gestern ein Programmpaket an, mit dem er die Vorherrschaft von Microsoft auf den PC-Bildschirmen brechen will.

Copyright © 05. 05. 1997 by Schweriner Volkszeitung.

Weitere für Sie herausgesuchte Artikel im Ressort TopNews:

PNP Dahlin-Tor wirft Leverkusen zurück Torjäger Sean Dundee bleibt beim KSC
Am 30. Spieltag der Fußball-Bundesliga verschenkt Bayer den Sieg in Mönchengladbach in letzter Sekunde

Eine Rostbratwurst für 1,5 Euro
Berliner werden im größten Testlauf mit neuer Währung vertraut gemacht

SPD sucht einen Tony Blair
Partei diskutiert Kurs und Programm

Was alles nicht fehlt
Am Rande

PNP Erhoffter Höhenflug wird zur Bauchlandung
Trotz 1:0-Eishockeysieg über Slowakei kommt kaum Freude auf

Die Verteidigung britischer Interessen steht ganz weit vorn
Außenminister Robin Cook will, daß London seinen "rechtmäßigen Platz" in Europa einnimmt

Roßkopfs Traum platzt auch im Doppel
Viertelfinal-Aus mit Partner Samsonow gegen Olympiasieger – Nicole Struse

Das Thema des Artikels interessiert mich sehr! Diesen Artikel fand ich uninteressant!

Abbildung 4.3: Ein Beispielartikel.

über den Schlager-Grand-Prix.

Nachdem der Manager die Berichte über eine Erprobung der Euro-Währung in Berlin, über die Äußerungen des neuen britischen Außenministers und zu den Friedensverhandlungen für Zaire unter südafrikanischer Schirmherrschaft gelesen hat, beschließt er, nun ein wichtiges privates Interesse vom *PZ*-System verfolgen zu lassen. Er wählt den ganz oben plazierten Artikel über den zurückliegenden Spieltag der Fußballbundesliga an und bestätigt dem System ausdrücklich, daß er weiterhin über dieses Thema informiert werden möchte.

Im Anfangspressespiegel des Managers ist kein Artikel über die Entwicklung an den Börsen vorhanden, obwohl jede der vertretenen Zeitungen einen Wirtschaftsteil besitzt. Um an Berichte über die Finanzmärkte zu gelangen, kann der Manager den *PZ*-Kiosk benutzen. Dazu klickt er auf die entsprechende Grafik im linken Bereich des Browserfensters. Im *PZ*-Kiosk besteht die Möglichkeit, sämtliche Artikel aller dem *PZ*-System bekannten Zeitungen zu lesen (siehe Abbildung 4.4). Die Artikel sind ähnlich wie im Pressespiegel entweder nach Ressort (hier Wirtschaft) oder Zeitung geordnet über die Hyperlinks auf der linken Seite zugänglich.

Der Manager sucht im Sportressort nach Artikeln über Tennisturniere am letzten Wochenende und wird bei der Passauer Neuen Presse fündig (Abbildung 4.5). Dem Text des Artikels folgt nicht nur die schon gewohnte Liste der restlichen Artikel des Ressorts, sondern zuerst eine Liste von Berichten zum gleichen Thema in anderen Zeitungen bzw. Ressorts. Diese ermöglicht dem Benutzer, die unterschiedliche Berichterstattung der verschiedenen Zeitungen zu diesem Thema schnell zu überblicken. Auch im Pressespiegel wird eine Liste gleichartiger Artikel zu einem Bericht angezeigt, falls mehrere Artikel zu einem Thema vorhanden sind.

Anders als im Pressespiegel deutet das *PZ*-System das Lesen eines Artikels im Kiosk noch nicht als Hinweis, daß dieses Thema den Benutzer interessiert. So hat ein „Herumschmökern“ in den Artikeln des Kiosks keinen unbeabsichtigten Einfluß auf die Erstellung des Pressespiegels. Während ein Bericht angezeigt wird, ist jedoch die Möglichkeit gegeben, explizit Interesse am Thema des Artikels zu äußern. Das Betätigen des in Abbildung 4.5 in der unteren Leiste zu sehenden Hyperlinks „Ähnliche Artikel hätte ich gerne in der nächsten *PZ*!“ wertet das *PZ*-System als genauso starke Interessensbekundung wie das Lesen eines Artikels im Pressespiegel. Stärkere (positive oder negative) Hinweise auf Interessen sind im Kiosk nicht möglich.

Der Manager weist das *PZ*-System im Kiosk nicht nur auf sein Interesse an Tennis hin, sondern läßt sich zusätzlich die Artikelliste des Wirtschaftressorts anzeigen. Er wählt mehrere Berichte über Börse und Aktien aus und teilt dem System sein Interesse an diesem Thema mit. Über Nacht verarbeitet das *PZ*-System nun das gesammelte Feedback und erstellt einen persönlichen Pressespiegel. Besucht der Manager am nächsten Morgen wieder das *PZ*-System, stellt er fest, daß der Pressespiegel nun eher seinen Informationsbedürfnissen entspricht: Im Res-

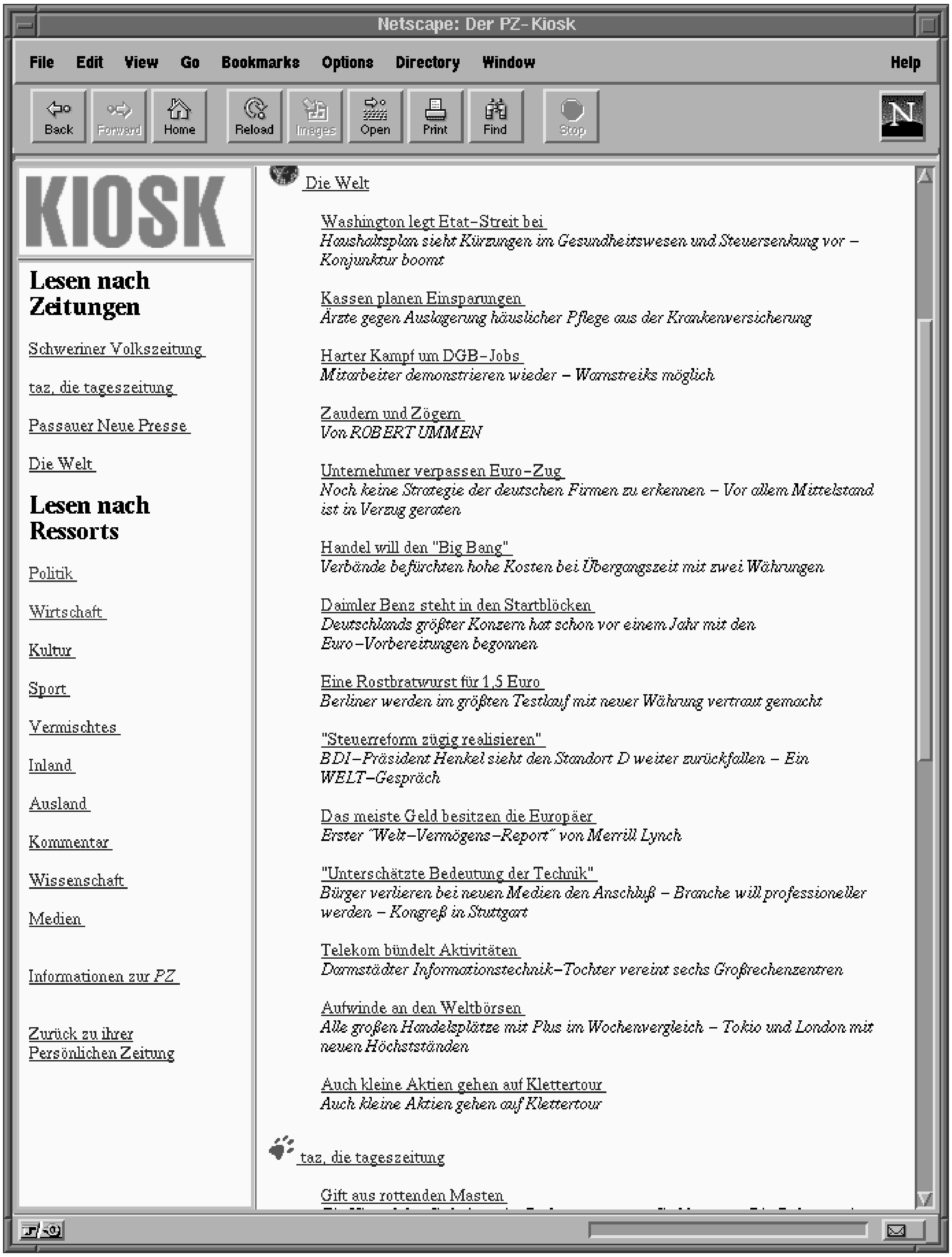


Abbildung 4.4: Artikelüberblick des Ressorts Wirtschaft im *PZ*-Kiosk.

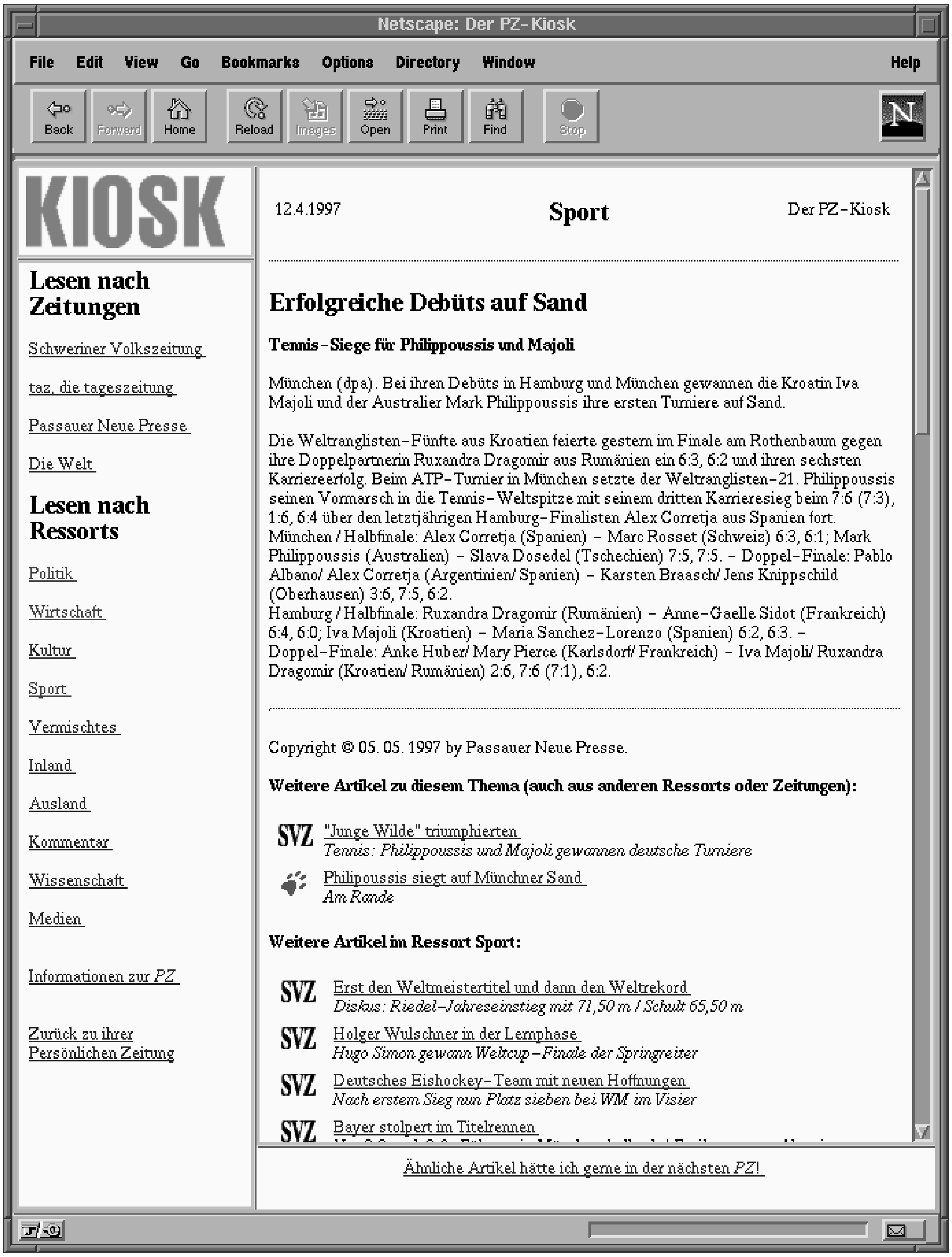


Abbildung 4.5: Artikelanzeige im *PZ*-Kiosk.

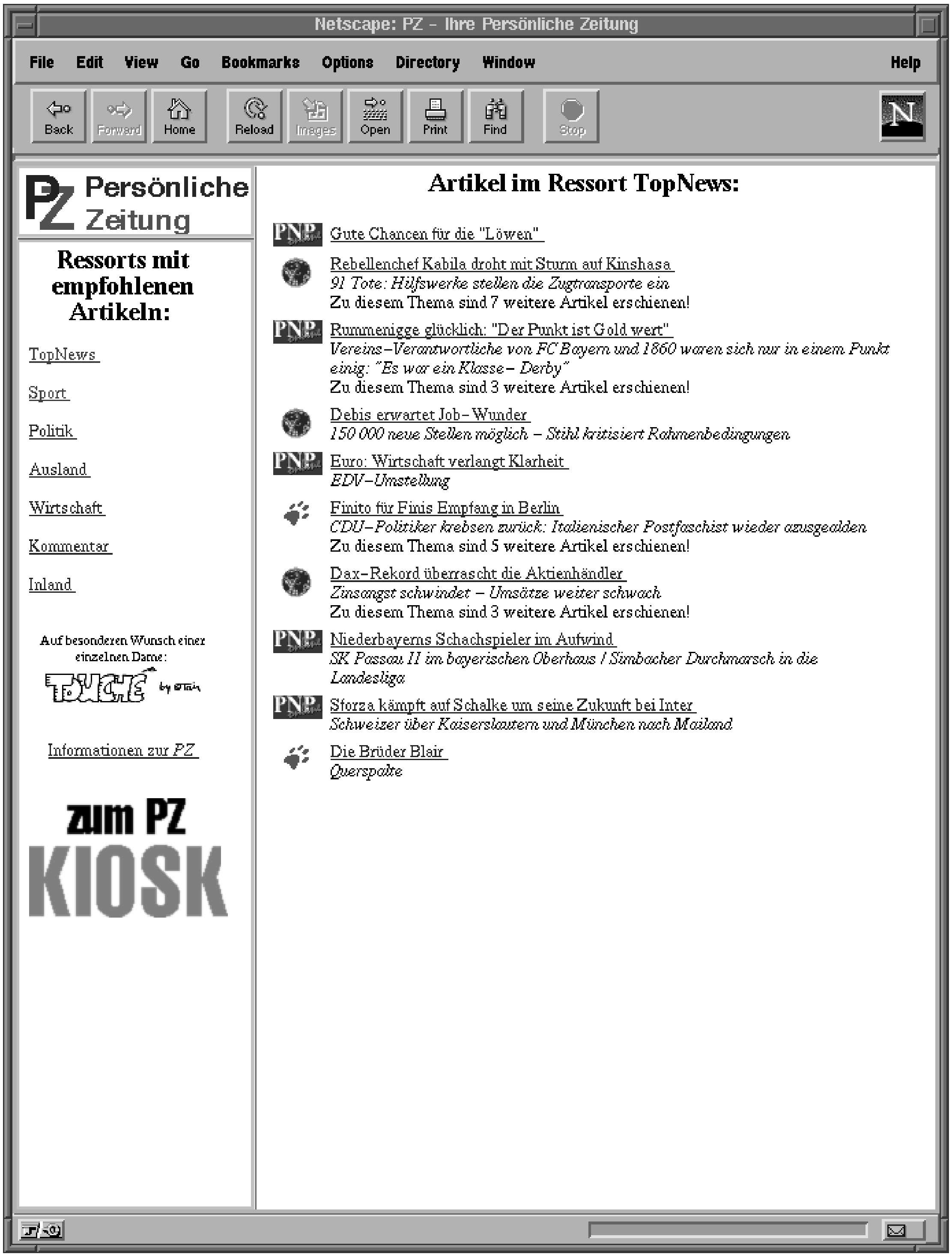


Abbildung 4.6: Der persönliche Pressespiegel am nächsten Tag.

sort TopNews sind Berichte über Fußball, Börsenentwicklungen, Zaire und die neue Regierung in Großbritannien stark vertreten (siehe Abbildung 4.6). Insgesamt besteht der Pressespiegel aus knapp 40 Artikeln und deckt die gewünschten Interessensgebiete schon gut ab. Anhand von erneuten Rückmeldungen von Interessen und Abneigungen für Artikel paßt sich das *PZ*-System in den folgenden Tagen dem Informationsbedarf des Managers weiter an.

Kapitel 5

PZ - Der persönliche Pressespiegel

Der persönliche Pressespiegel erscheint für den Anwender in Form von HTML-Seiten im WWW-Browserfenster. Hinter dieser Fassade steckt das *PZ*-System, das Zeitungsartikel aus dem WWW holt, für mehrere Benutzer filtert und für jeden Benutzer anschließend die entstandene Zeitung zur Anzeige aufbereitet. Die Gestaltung des *PZ*-Systems erfolgte aufgrund einiger Beobachtungen und Annahmen über den Umgang von Anwendern mit konventionellen Zeitungen und Rechnern:

- Das große Themenangebot und die große Meinungsvielfalt der knapp 100 im WWW vertretenen deutschsprachigen Tageszeitungen[1] ist ohne Unterstützung für einen Einzelnen nur ansatzweise nutzbar.
- Zeitungsleser lassen sich bei der Auswahl der Artikel, die sie lesen, von einem breitgefächerten Interesse leiten.
- Teilbereiche des Benutzerinteresses wandeln sich mit der Zeit nur langsam, während andere Interessen kurzfristig erst geweckt werden und anschließend genauso schnell wieder einschlafen.
- Erfordert die Anpassung eines Interface-Agenten an einen Benutzer einen hohen Aufwand, sinkt die Akzeptanz eines solchen Systems.
- Eine sehr dynamische Anzeige eines komplexen Informationsraumes erschwert die Navigation in diesem Raum.

Aufgrund dieser Voraussetzungen wurden folgende Ziele für das *PZ*-System gesteckt:

[1] Stand Mai 1997. Eine Liste deutschsprachiger Tageszeitungen mit Internetangebot kann man bei [Web.de, 1997] finden.

1. Das *PZ*-System soll seinen Benutzern helfen, einen Überblick auf das Informationsangebot mehrerer Zeitungen zu erhalten, indem es die Aufmerksamkeit des Benutzers auf wenige ihn interessierende Artikel lenkt und das Gros der uninteressanten Artikel verbirgt.

2. Das *PZ*-System soll sich mit möglichst wenigen Eingriffen der Benutzer selbständig an deren Interessen anpassen.

3. Das *PZ*-System soll das Interesse jedes Benutzers mit seinen vielen verschiedenen Facetten abbilden können und in der Lage sein, dien zeitlichen Veränderungen des Interesses zu folgen.

4. Das *PZ*-System soll über eine unkomplizierte, leicht durchschaubare Oberfläche einfach zu benutzen sein.

Wie werden diese Ziele nun konkret im *PZ*-System umgesetzt? Das erste Ziel legt die Gestaltung des *PZ*-Systems als auf das Filtern von Zeitungsartikeln spezialisierten Interface-Agenten nahe. Dieser führt in der Art eines *Learning Apprentice* (siehe Seite 32) durch Beobachtung die Adaptation an den Benutzer im Sinne des zweiten Ziels durch. Die unter 3. geforderte Modellierung der verschiedenen Aspekte des Benutzerinteresses geschieht im *PZ*-System durch heterogene Agenten in einem Multiagentensystem, das durch Lernmechanismen auf verschiedenen Ebenen auch zeitliche Veränderungen des Interesses verfolgt. Um das letzte Ziel zu erreichen, wurde eine Benutzerschnittstelle entworfen (siehe Kapitel 4), die das erprobte Design schon im WWW vertretener Zeitungen erweitert. Die Zeitungsartikel werden nicht online gefiltert, da die ständige Aktualisierung des Pressespiegels den Benutzer sehr verwirren würde.

Dieses Kapitel ist wie folgt organisiert. Der erste Abschnitt stellt den Aufbau des gesamten *PZ*-Systems vor. In Abschnitt 5.2 wird dann speziell auf die Durchführung des Filterns im *PZ*-System eingegangen. Der folgende Abschnitt 5.3 widmet sich den Lernverfahren, die zur Anpassung der einzelnen Agenten und des gesamten Agentensystems an den Benutzer verwendet werden. Die Ansätze der vorliegenden Arbeit werden im letzen Abschnitt des Kapitels mit anderen Informationsfiltern verglichen, die ebenfalls als Multiagentensystem aufgebaut sind.

5.1 Aufbau des *PZ*-Systems

Informationsfilter arbeiten mit Strömen von Dokumenten, deshalb haben IF-Systeme fast immer eine *Pipeline-Struktur*: Die Dokumente gelangen an einer Stelle in das System, werden nacheinander verschiedenen Bearbeitungsschritten unterworfen und treten – abhängig von dem Ergebnis der Verarbeitung – wieder aus dem System aus. Im Architekturschaubild (Abbildung 5.1) des *PZ*-Systems ist diese Struktur im Verlauf von oben nach unten gut zu erkennen. Eingangsdaten sind HTML-Seiten mit Zeitungsartikeln, die von WWW-Servern deutscher

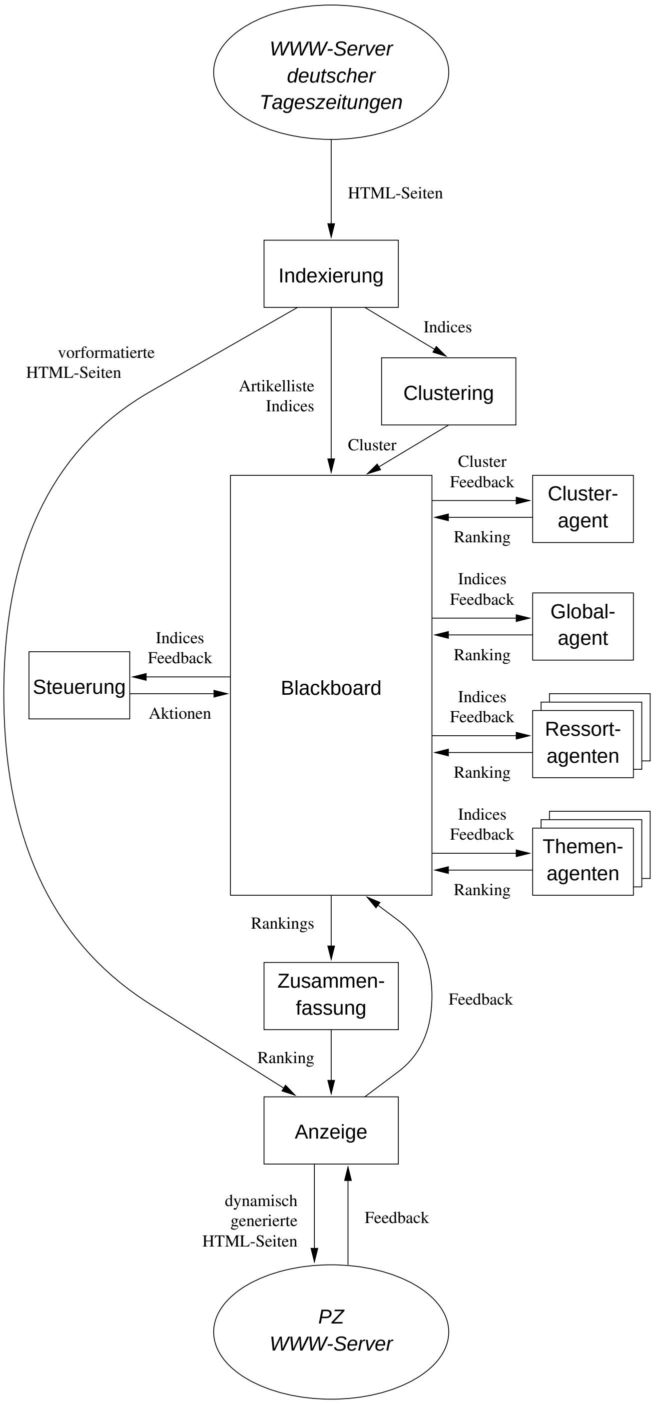


Abbildung 5.1: Architektur des *PZ*-Systems.

Tageszeitungen stammen. Die ersten Verarbeitungsschritte sind die *Indexierung* der Dokumente und die Durchführung eines *Clusterings*. Anschließend werden die Ergebnisse (Indices und Cluster) in ein *Blackboard* gespeichert. Aus diesem werden nach dem Selektionsprozeß die Bewertungen der verschiedenen Filteragenten entnommen und zu einem Ranking der Zeitungsartikel *zusammengefaßt*. Die *Anzeige* der als interessant bewerteten Artikel erfolgt wiederum im WWW über einen eigenen *PZ*-WWW-Server.

Charakteristisch für einen *adaptiven* Informationsfilter ist die Rückführung der Reaktionen der Benutzer in den Filterprozeß. Auch dies erfolgt im *PZ*-System. Die Anzeigekomponente sammelt Feedback und stellt es über das Blackboard allen Agenten zur Verfügung.

Kernstück der *PZ*-Architektur ist das Multiagentensystem, das die Selektion der Artikel durchführt. Es besteht aus einer Population mit je einem *Cluster-* und *Globalagenten* und einer variablen Menge von *Ressort-* und *Themenagenten*. Jedem Benutzer ist eine eigene Population von auf ihn spezialisierten Agenten zugeordnet. Die Agenten benutzen das zentrale Blackboard, um gemeinsam an der Filteraufgabe zu arbeiten. Jeder Agent erstellt auf Grundlage der Artikelindices aus dem Blackboard ein Ranking der Artikel und lernt aus dem Feedback, das er ebenfalls von dort bekommt. Die *Steuerung* fordert die einzelnen Agenten der Reihe nach auf, die Daten im Blackboard zu bearbeiten und Ergebnisse hineinzuschreiben. Bleiben Daten im Blackboard übrig, für die sich kein Agent zuständig erklärt, so sorgt die Steuerung dafür, daß neue Agenten erzeugt werden, die zukünftig die Bearbeitung dieser Daten übernehmen.

5.2 Der Filterprozeß

5.2.1 Indexierung

Das *PZ*-System wurde – wie oben begründet – nicht als Online-Filter ausgelegt, sondern verarbeitet die Dokumente immer nur blockweise. Ein solcher Block besteht aus den Zeitungsartikeln, die an einem Tag in den Internetausgaben von vier deutschen Tageszeitungen erschienen sind. Jede Nacht lädt ein externes Hilfsprogramm[2] die gesamten HTML-Seiten der Zeitungen zu dem aktuellen Datum von den WWW-Servern der Tageszeitungen und legt diese in einem Verzeichnis auf der lokalen Festplatte ab.

Im *PZ*-System besteht jeder Artikel aus den Datenfeldern *Schlagzeile, Kopfzeile, Untertitel, Kurzzusammenfassung, Artikeltext, Zeitungsname, Ressort, Autor* und *Datum*. Die *Textextraktion* zieht in zwei Stufen aus den gespeicherten HTML-Seiten diese Artikeldaten heraus.

[2]Für das *PZ*-System wird das Programm WEBCOPY von Victor Parada verwendet, erhältlich unter `ftp://ftp.inf.utfsm.cl/pub/utfsm/perl/webcopy.tgz`.

Die HTML-Seiten der Zeitungen werden zunächst in drei Kategorien eingeteilt:

1. HTML-Seiten, die den Text eines Zeitungsartikels und u.U. zusätzlich Verweise auf weitere Artikel oder auf andere Angebote des WWW-Servers der Zeitung enthalten,

2. HTML-Seiten, die Überblicke über einzelne Ressorts oder die ganze Zeitung enthalten, die in Form von Listen mit Verweisen auf Seiten der ersten Kategorie gestaltet sind,

3. HTML-Seiten, die sonstige Angebote der Zeitung beschreiben (z.B. die Möglichkeit, per E-Mail einen Leserbrief zu schreiben, oder eine Auflistung der aktuellen Börsenkurse).

Für den Filterprozeß müssen die HTML-Seiten der ersten Kategorie herausgefunden werden. Nach einer Analyse der verschiedenen Formate der HTML-Seiten wurden für das *PZ*-System von Hand Regeln aufgestellt, um die Seiten über das Auftreten bestimmter Schlüsselwörter zu klassifizieren.

Die nach dieser ersten Stufe übrigbleibenden Seiten enthalten nicht nur die Artikeldaten. Z.B. enthält jede Seite für die Darstellung des Artikels im WWW-Browser HTML-Tags wie `<I>` oder `<CENTER>`. Diese Tags sollten nicht zur Indexierung des Artikels verwendet werden, da sie keine inhaltliche Information liefern. Die Texte von Verweisen auf andere Artikel oder auf die Stammseite der Zeitung gehören ebenfalls nicht zu den Daten eines Artikels, die in der zweiten Stufe extrahiert werden sollen.

Die Tageszeitungen produzieren ihre Internetausgaben parallel zu der konventionellen Papierausgabe. Da das Internetangebot bisher eher einen zusätzlichen Service für die Leser als eine Einnahmequelle für die Zeitungsverlage darstellt, muß die Erstellung der HTML-Seiten kostengünstig ohne viel Aufwand erfolgen. Heutzutage werden Zeitungen nur noch mit Computersatzsystemen produziert, so daß die Artikel schon in elektronischer Form vorliegen. Die Verlage benutzen die Exportmöglichkeiten der Satzsysteme, um nicht nur Ausgaben für Druckmaschinen, sondern auch automatisch HTML-Seiten zu erzeugen. Dazu werden allgemeine Layoutvorlagen mit den Daten eines konkreten Artikels ausgefüllt. Aufgrund der Automatisierung haben die HTML-Seiten einer Zeitung eine gewisse Struktur, die zur Extraktion der Artikeldaten ausgenutzt werden kann.

Im *PZ*-System gibt es für jede Zeitung eine spezialisierte Menge von Regeln in Form von regulären Ausdrücken. Diese schneiden aus den HTML-Seiten die Artikeldaten heraus. Wie schon die Regeln zur Klassifikation der HTML-Seiten sind auch diese Regeln durch die Analyse vieler Seiten entstanden. Bei einer Layoutänderung von Seiten der Zeitungsverlage sind die Regeln – mit beträchtlichen Aufwand – von Hand neu zu erstellen. Hier böte sich in weiteren Versionen des *PZ*-Systems der Einsatz eines Lernverfahrens an, das auf der Grundlage einiger

Beispiel-HTML-Seiten, bei denen die Artikeldaten markiert sind, eine Syntax der Struktur der Seiten lernt. Während der Anwendung der Syntax zum Parsen neuer Seiten kann die Performanz der Extraktion überwacht werden. Steigt die Fehlerrate stark an, wird eine Fehlermeldung ausgeben und anhand der Markierungen eines menschlichen Experten für die fehlerhaft extrahierten Dokumente kann neu gelernt werden.

Bis auf den Artikeltext werden die Daten aller Artikel eines Tages in einer *Artikelliste* gespeichert, die im Blackboard abgelegt wird. Mit den Artikeldaten werden ebenfalls schon vorformatierte HTML-Seitenteile für jeden Artikel erzeugt. Die Anzeigekomponente des *PZ*-Systems ergänzt diese Teile anschließend um benutzerspezifische Abschnitte (z.B. Listen von Verweisen auf weitere gefilterte Artikel) ud stellt sie für den Benutzer dar.

Die Textdaten eines Artikels sind Schlagzeile, Kopfzeile, Unterzeile und Artikeltext. Nur diese werden für das Filtern zur Repräsentation des Inhalts benutzt. Das *PZ*-System verwendet Wörter als Indexierungsterme und speichert diese in der Vektorraumdarstellung als geordnete Listen von (Term,Termgewicht) Paaren ab.

Um den Einsatz von Lernverfahren zu vereinfachen, sollte der Termraum möglichst klein gehalten werden. Da es sich bei den zu filternden Dokumenten um deutsche Tageszeitungsartikel handelt, wurden bei der Auswahl von Dimensionsreduktionsverfahren (siehe Abschnitt 2.3.1.3) die Besonderheiten dieser Domäne beachtet (siehe auch [Schewe, 1997]). Zum einen haben Zeitungsberichte einen eigenen Stil, der als *Pressesprache* bezeichnet wird, zum anderen sind die Artikel in Deutsch verfaßt.

Die Syntax der Pressesprache zeichnet sich u.a. durch einen ausgeprägten *Nominalstil* und die Bevorzugung von Einfachsätzen vor komplizierten Satzgefügen aus. Der Wortschatz besteht aus vielen Ad-hoc-Zusammensetzungen (z.B. Euro-Vorbereitungen oder Viertelfinal-Aus), fachsprachlichen Ausdrücken, Fremdwörtern und aktuellen Bezeichnungen (z.B. Steuerreform oder Rinderseuche). Adjektive und Substantive repräsentieren durch die lexikalischen und syntaktischen Merkmale der Pressesprache den Inhalt eines Artikel in weitaus höherem Maße als z.B. Verben, Adverbien oder Artikel.

Kennzeichnend für die deutsche Sprache ist u.a. eine eine große Vielfalt an Flexionen. Dies unterscheidet Deutsch z.B. von der englischen Sprache, die die Dokumentensprache bei den meisten IR- und IF-Systemen ist.

Diese beiden Charakteristika der Domäne legen die Termauswahl nach den Wortarten Substantiv (hierunter werden auch Orts- und Eigennamen verstanden) und Adjektiven und die Reparametrisierung des Termraumes durch eine Grundformreduktion.

Der nächste Schritt der Indexierung ist daher eine *morphologische Analyse* der Textdaten aller Artikel. Dafür wird das Programm GERTWOL der finnischen Firma Lingsoft, Inc. benutzt. GERTWOL ist ein System zur automatischen Wortformerkennung deutscher Wörter [Haapalainen und Majorin, 1994]. Es besitzt ein umfangreiches Lexikon mit etwa 85.000 Wortformen, eine umfassende Derivationsmorphologie und einen vollständigen Mechanismus zur Kompositabildung und -zerlegung. In Tests des Herstellers auf verschiedenen Textkorpora erzielte GERTWOL eine Datenabdeckung von über 98% auf unbeschränkten Texten. GERTWOL gibt zu jedem eingegebenen Wort alle möglichen Lesungen an, die jeweils aus der Grundform des Wortes und morphologischen Daten zur Wortform (u.a. der Wortart) bestehen. Das Wort „Systeme" wird z.B. folgendermaßen zerlegt:

```
"<*systeme>"
        "*system" S NEUTR PL NOM
        "*system" S NEUTR PL AKK
        "*system" S NEUTR PL GEN
```

Die morphologische Analyse wird von der *Termgenerierung* zur Dimensionsreduktion benutzt. Es werden nur diejenigen Grundformen der Wörter in die Artikelindices aufgenommen, die die Wortart Substantiv (GERTWOL-Kennzeichnung `S`), Eigenname (`EIGEN`) oder Adjektiv (`A`) besitzen. Im gleichen Verarbeitungsschritt werden die Terme gezählt und so die Termfrequenz ermittelt. Diese wird als als vorläufiges Termgewicht im Artikelindex verwendet. Da alle Zeitungsartikel hintereinander bearbeitet werden, kann auch die Dokumentfrequenz eines jeden Terms der Kollektion bestimmt werden. Diese Information wird ebenfalls in Vektordarstellung als Kollektionsindex an die nächste Verarbeitungsstufe weitergereicht. Die *TFIDF-Gewichtung* errechnet nun die TFIDF-Gewichte nach der auf Seite 16 angegebenen Formel und speichert die gewichteten Indices ab.

Abbildung 5.2 zeigt abschließend den Aufbau der Indexierungskomponente des *PZ*-Systems und den Fluß der Daten durch die einzelnen Verarbeitungsschritte.

5.2.2 Clustering

Das *PZ*-System verarbeitet die Artikel mehrerer Tageszeitungen. Zu einem Nachrichtenthema gelangen daher oft mehrere Artikel in das System. Für den Benutzer ist es hilfreich, wenn diese Gruppen von Artikeln zu einem Thema gemeinsam angezeigt werden. So kann er zu einer Nachricht schnell einen Überblick über die verschiedenen Meinungen der Zeitungen erhalten.

Die Artikelkollektion besteht zu Anfang aber nur aus einer Menge von einzelnen Artikeln. *Clustering* (auch *Clusteranalyse* genannt) ist ein Verfahren des Maschinellen Lernens, das in einer Menge von ungeordneten Objekten Gruppen ähnlicher Objekte (*Cluster*) identifiziert. Dazu ist eine formale Definition der Ähnlichkeit zweier Objekte nötig. Beim Clustering der Zeitungsartikel im *PZ*-System kann dies mit Hilfe der Darstellung der Artikel im Vektorraummodell geschehen. Ein bewährtes Ähnlichkeitsmaß $\mathrm{sim}_d(d_i, d_j)$ für zwei Artikel d_i und d_j mit den

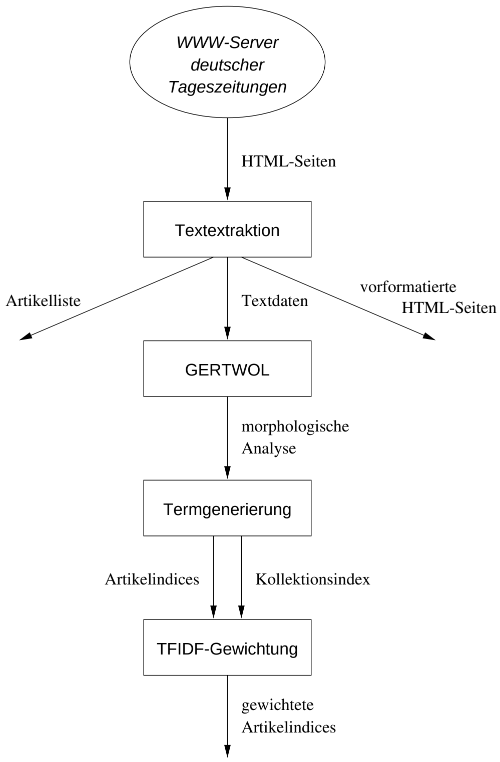


Abbildung 5.2: Aufbau der Indexierungskomponente.

Termvektoren $\vec{d_i}$ und $\vec{d_j}$ ist der Kosinus des Winkels, der von den beiden Vektoren eingeschlossen wird (Kosinus-Maß). Die Ähnlichkeit kann als Skalarprodukt der beiden normierten Vektoren errechnet werden:

$$\mathrm{sim}_d(d_i, d_j) = \cos(\angle(\vec{d_i}, \vec{d_j})) = \frac{\vec{d_i}}{|\vec{d_i}|} \cdot \frac{\vec{d_j}}{|\vec{d_j}|}$$

Nicht-hierarchische Cluster können als Mengen von Artikeln dargestellt werden. Zwischen Clustern muß ebenfalls ein Ähnlichkeitsmaß definiert werden. Man kann z.B. die Centroiden de Termvektoren der im Cluster enthaltenen Artikel bestimmen und diese Vektoren mit dem obigem Ähnlichkeitsmaß vergleichen. Die im *PZ*-System verwendete Methode berechnet die Ähnlichkeit zweier Cluster c_k und c_l als den Durchschnitt der Ähnlichkeiten der Artikel des einen Clusters zu allen Artikeln des anderen Clusters (*group average method*):

$$\mathrm{sim}_c(c_k, c_l) = \frac{\sum_{d_i \in c_k} \sum_{d_j \in c_l} \mathrm{sim}_d(d_i, d_j)}{|c_k| \cdot |c_l|}$$

Der Algorithmus zum Clustern der Artikelkollektion $K = \{d_1, \ldots, d_n\}$ lautet folgendermaßen:

1. Initialisiere die Menge C der Cluster mit $\{c_1, \ldots, c_n\}$, wobei $c_k = \{d_k\}$ für $k = 1, \ldots, n$ ist.
2. Suche das Clusterpaar (c_k, c_l) mit $c_k \neq c_l$, das die maximale Clusterähnlichkeit aufweist:
$$(c_k, c_l) = \operatorname*{argmax}_{c_k, c_l \in C} \mathrm{sim}_c(c_k, c_l)$$
3. Falls $\mathrm{sim}_c(c_k, c_l)$ kleiner als der Schwellwert θ_{Cl} ist, gebe alle Elemente c_k der Menge C aus, die $|c_k| > 2$ erfüllen, und beende den Algorithmus.
4. Andernfalls erzeuge einen neuer Cluster c' als Vereinigung der Cluster c_k und c_l:
$$c' = c_k \bigcup c_l$$
5. Entferne die Cluster c_k und c_l aus der Clustermenge C und füge den Cluster c' hinzu:
$$C' = (C \setminus \{c_k, c_l\}) \bigcup \{c'\}$$
6. Verwende C' als die aktuelle Clustermenge C und fahre bei 2. fort.

Der Algorithmus hat eine Laufzeit von $O(n^2)$, da beim ersten Durchlauf des Punktes 2. die Ähnlichkeiten aller n Artikel zueinander berechnet werden müssen. Die Ergebnisse können in einer Matrix abgespeichert werden und stehen somit in folgenden Zyklen für die Berechnung von $\mathrm{sim}_c(c_k, c_l)$ ohne Neukalkulation zur Verfügung.

Der Schwellwert θ_{Cl} beschreibt die *Intracluster-Ähnlichkeit*, die die Cluster mindestens besitzen müssen und ist im *PZ*-System nach einigen Testläufen auf 0,12 festgelegt worden. Die so erzeugten Cluster bestehen durchweg aus Artikeln, die sich thematisch stark ähneln.

Im *PZ*-System werden Cluster nur weiter verwendet, wenn sie mehr als drei Artikel umfassen. Die Clusterbeschreibungen werden zu den Ergebnissen der Indexierung in das Blackboard geschrieben.

5.2.3 Filteragenten

Das Multiagentensystem besteht aus einer Population von Filteragenten, die jeweils Zeitungsartikel mit unterschiedlichen Verfahren bewerten. Jeder Agent hat gleichermaßen Zugriff auf die Artikelindices, den Kollektionsindex, die Artikelliste und die Clusterbeschreibungen im Blackboard. Die Agenten benutzen diese Daten, um Empfehlungen für Artikel zu erstellen. Eine Empfehlung wird immer von einem Konfidenzwert des Agenten für die Empfehlung begleitet, so daß eine

Rangordnung (*Ranking*) unter den Empfehlungen entsteht. Artikel, bei denen ein Agent sich seiner Empfehlung sehr sicher ist, die Konfidenz also einen hohen Wert hat, sollten dem Benutzer als besonders interessant angeboten werden. Vertraut der Agent seiner Empfehlung nicht so sehr, wird der Artikel erst später angezeigt.

Im *PZ*-System existieren vier Arten von Filteragenten: Clusteragenten, Globalagenten, Ressortagenten und Themenagenten. Die letzen drei Arten bestimmen ihre Empfehlungen für Artikel durch eine Modellierung des Benutzerinteresses auf *thematischer* Ebene, während ersterer das Interesse des Benutzers an *aktueller* Information abbildet und darauf basierend Empfehlungen ausspricht. Im folgenden wird der Empfehlungsmechanismus jeder Agentenart beschrieben.

Clusteragent (CA)**:** Wie schon in Kapitel 4 erwähnt, kann die Anzahl der Artikel zu einem Thema als Indikator für die Wichtigkeit dieses Themas aufgefaßt werden. Wichtigkeit bestimmt nicht die Relevanz des Artikel für den Benutzer, sondern mißt die Bedeutsamkeit des Artikelthemas aufgrund der Nachrichtenlage aus der globaleren Perspektive der Zeitungsherausgeber.

Das Clustering gruppiert nun gerade ähnliche Dokumente der Artikelkollektion zusammen. Die Größe eines Clusters kann also als Wichtigkeitsmaß verwendet werden. Dieses Maß ermöglicht es dem Clusteragenten, die wichtigsten Cluster herauszufinden. Anstatt dann Empfehlungen für alle Artikel dieser Cluster auszusprechen, empfiehlt der Clusteragent jedoch nur den innersten Artikel jedes Clusters als Reräsentanten für das Thema des Clusters. Der innerste Artikel ist derjenige Artikel, der die höchste Ähnlichkeit zu dem Centroiden des Clusters aufweist. Der Centroid eines Clusters wird durch den Termvektor repräsentiert, der durch das Aufsummieren der Termvektoren aller im Cluster enthaltenen Artikel und anschließende Division durch die Anzahl der Artikel berechnet wird. Der Konfidenzwert zu einem empfohlenen Artikel ist proportional zu der Größe des Clusters, aus dem er stammt.

Die Agentenpopulation enthält genau einen Clusteragenten.

Globalagent (GA)**:** Dieser Agent (wie die beiden folgenden) benutzt ein Profil in Form eines Termvektors als Repräsentation der Interessen des Benutzers. Die Erstellung des Profiles wird in Abschnitt 5.3.1 erläutert, für die weitere Beschreibung hier wird von der Existenz eines solchen Profils ausgegangen. Der Globalagent berechnet nun die Ähnlichkeit seines Profils p_{GA} mit den Artikeln $d_1, \ldots, d_n$ der Artikelkollektion K mit Hilfe des Kosinusmaßes. Ist die Ähnlichkeit größer als der Schwellwert θ_{GA}, so wird der entsprechende Artikel vom Globalagenten empfohlen. Die Konfidenz der Empfehlung ist der Wert der Ähnlichkeit zwischen Profil und Artikel.

Es gibt nur einen Globalagenten in der Agentenpopulation eines Benutzers.

Ressortagent (RA_r): Der Globalagent betrachtet alle Artikel in der Kollektion. Die Ressortagenten arbeiten ähnlich, beschränken aber die Berechnung der Ähnlichkeit zum Profil p_{RA_r} auf die Artikel des Ressorts r und benützen den Schwellwert θ_{RA_r}

Zu jedem Ressort r existiert höchstens ein Ressortagent.

Themenagent (TA_i): Diese Agenten vergleichen wieder alle Artikel mit ihrem Profil p_{TA_i} und empfehlen diejenigen, deren Ähnlichkeit zum Profil über dem Schwellwert θ_{TA_i} liegt. Sie unterscheiden sich aber von Globalagenten in der Weise, wie das Profil aufgebaut wird (siehe 5.3.1).

Die Anzahl der Themenagenten in der Agentenpopulation eines Benutzers ist nicht von vorneherein festgelegt, sondern ist Ergebnis des Lernens, und kann – wie die Zahl der Ressortagenten – über der Zeit variieren.

5.2.4 Zusammenfassung der Empfehlungen

Die Agenten schreiben ihre Empfehlungen mit den dazugehörigen Konfidenzwerten in das Blackboard. Zu einem Artikel existieren also u.U. mehrere Empfehlungen von verschiedenen Agenten mit jeweils unterschiedlicher Konfidenz. Damit *eine* Rangordnung der gefilterten Artikel entsteht, sollte das Ergebnis der Filterung aber *eine* Empfehlung mit *einem* Konfidenzwert für jeden Artikel sein. Dazu müssen die Ergebnisse der Filteragenten zusammengefaßt werden.

Jeder von einem Agenten empfohlene Artikel gelangt in den Pressespiegel. Der Konfidenzwert der Empfehlung des Artikel für den Pressespiegel berechnet sich als gewichtete Summe der Konfidenzwerte der Agenten. Dabei wird nicht für jeden Agenten ein eigenes Gewicht vergeben, sondern jede Agentenart hat ein ihr assoziiertes Gewicht ($g_{CA}, g_{GA}, g_{RA}, g_{TA}$). Bei Cluster-, Global- und Ressortagenten bedeutet dies keine Einschränkung, da zu einem Artikel höchstens eine Empfehlung der jeweiligen Agentenart vorliegen kann. Wenn es aber mehrere, sich sehr ähnelnde Themenagenten gibt, kann es vorkommen, daß mehrere Themenagenten Empfehlungen für einen Artikel geben. In diesem Fall wird die Empfehlung mit dem höchsten Konfidenzwert als Empfehlung der Agentenart verwendet (siehe auch [de Kroon et al., 1996]). Die Gewichte der Zusammenfassungskomponente drücken eine Art Vertrauen aus, daß Empfehlungen einer Agentenart das Interesse des Benutzers treffen.

5.3 Lernen durch Feedback

Für das *PZ*-System wurde u.a. das Ziel formuliert, daß sich das System selbständig an das Interesse der Benutzer anpasst. Außerdem sollte die Veränderung des Benutzerinteresses mit der Zeit verfolgt werden können. Um diese Ziele zu erreichen, benutzt das *PZ*-System die Reaktionen der Benutzer auf die erstellten Pressespiegel. Während der Benutzung des Pressespiegels nimmt die Anzeigekomponente implizites (der Benutzer liest einen Artikel bzw. er überspringt ihn) und explizites (der Benutzer betätigt einen der Feedbackhyperlinks im Anzeigefenster) Feedback des Benutzers auf. Über das Blackboard kann das Multiagentensystem auf diese Daten zugreifen. Die Reaktionen des Benutzers auf einen Artikel d werden im PZ-System wie folgt gewertet:

Benutzerreaktion	Bewertung $FB(d)$
Artikel nicht gelesen	0
Artikel gelesen oder im Kiosk bewertet	1
Artikel als interessant bewertet	2
Artikel als uninteressant bewertet	-1

Zur anfänglichen Erfassung und zur ständigen Verfolgung des Benutzerinteresses werden im PZ-System eine Reihe von Lernverfahren auf zwei verschiedenen Ebenen eingesetzt.

Die einzelnen Agenten lernen jeweils durch den *Rocchio-Algorithmus* unterschiedliche Aspekte des Benutzerinteresses. Sie verwenden einen *Schwellwertlerner*, um zu entscheiden, welche Artikel empfohlen werden und welche nicht.

Im gesamten Agentensystem findet die Anpassung an den Benutzer zum einen durch die *Erzeugung neuer Agenten* für neue Interessensbereiche statt. Zweitens werden durch ein *Belohnungs/Bestrafungssystem* Agenten, die einen Teilbereich des Interesses gut abbilden, gefördert und Agenten mit schlechten Modellen des Interesses zurückgedrängt. Schließlich erfolgt eine *Anpassung der Gewichte* der Zusammenfassungskomponente gemäß der Filterleistung der Agentenarten.

Diese Lernmechanismen werden im folgenden einzeln beschrieben.

5.3.1 Lernen der einzelnen Agenten

Jeder Agent soll einen Aspekt des Benutzerinteresses abbilden. Bei Clusteragenten, die das Interesse an aktuellen Nachrichten repräsentieren, ist die Anpassung an den Benutzer alleine durch die im Abschnitt 5.3.2 beschriebenen Mechanismen ausreichend. Die Verfahren in diesem Abschnitt werden also nur von Global-, Ressort- und Themenagenten angewendet.

5.3.1.1 Der Rocchio-Algorithmus

Bei der Entwicklung von SMART, einem experimentellen Information Retrieval System [Salton, 1971], wurde bemerkt, daß Benutzer ihre initiale Anfrage, die aus wenigen Termen bestand, nach der Anzeige einiger Dokumente oft um weitere Terme ergänzten und das System eine neue Antwort generieren liessen. Durch die zusätzlichen Terme wurde die Suche nach Dokumenten weiter fokussiert und der Anteil relevanter Dokumente in der Antwort des IR-Systems stieg. Der Zyklus

Anfrage/Anzeige von Dokumenten/Reformulierung der Anfrage wurde solange durchlaufen, bis der Informationsbedarf des Benutzers gedeckt war.

Das *Relevance Feedback* bietet für den Benutzer eine Vereinfachung des Prozesses. Anstatt den Informationsbedarf charakterisierende Terme anzugeben, muß er nur die Relevanz einiger Dokumente bezüglich seines Interesses markieren. Wird das Vektorraummodell für Dokumente d und die Anfrage q_n benutzt, gibt der Rocchio-Algorithmus an, wie eine neue Anfrage aufgrund der Relevanzmarkierungen für die Dokumente erzeugt wird [Rocchio, 1971]. Sei P die Menge der als relevant markierten Dokumente aus der Antwortmenge des Systems für die Anfrage q_n. Die nicht relevanten Dokumente bilden die Menge N. Dann kann ein Vektor für eine neue Anfrage q_{n+1} wie folgt berechnet werden[3]:

$$\vec{q}_{n+1} = \vec{q}_n + \frac{1}{|P|} \sum_{d \in P} \vec{d} - \frac{1}{|N|} \sum_{d \in N} \vec{d}$$

Durch die Iteration des Prozesses nähert sich die Anfrage q_n immer mehr der für das Benutzerinteresse optimalen Anfrage q_{opt}.

Im Information Filtering kann der Rocchio-Algorithmus zur Erstellung von Benutzerprofilen verwendet werden. Für jede vom Benutzer vorgegebene Klasse C der Dokumente kann mittels eines Schrittes des Rocchio-Algorithmus ein Profil p_C in Termvektordarstellung berechnet werden (Die Mengen P_C und N_C enthalten die positiven bzw. negativen Beispiele der Klasse C):

$$\vec{p}_C = \frac{1}{|P_C|} \sum_{d \in P_C} \vec{d} - \frac{1}{|N_C|} \sum_{d \in N_C} \vec{d}$$

Die Klassifikation eines neuen Dokuments erfolgt nun durch die Berechnung der Ähnlichkeit des Dokuments d mit den Profilen p_C mit Hilfe des Kosinusmaßes $\mathrm{sim}_d(d, p_C)$. Das Dokument wird der Klasse zugeordnet, mit dessen Profil es die höchste Ähnlichkeit hat ($\mathcal{C}$ ist die Menge aller Klassen):

$$H_{Rocchio}(d) = \operatorname*{argmax}_{C \in \mathcal{C}} \mathrm{sim}_d(d, p_C)$$

Ist die Klassifikation der Dokumente binär + oder − (z.B. relevant/nicht relevant) wird nur ein Profil p berechnet und ein Schwellwert θ benutzt, um die Klassen zu trennen:

$$H_{Rocchio_{bin}}(d) = \begin{cases} + & \text{für} \mathrm{sim}_d(d, p) > \theta \\ - & \text{sonst} \end{cases}$$

Im *PZ*-System wird der Rocchio-Algorithmus nicht nur zur einmaligen Berechnung von initialen Profilen benutzt. Falls der Benutzer Artikel aus seinem

[3] Hier und im folgenden wird angenommen, daß jeder Anfrage q_n ein Termvektor $\vec{q}_n$ zugeordnet ist. Die Notation von Dokumenten und Profilen und zugehörigen Termvektoren erfolgt gleichermaßen.

Pressespiegel gelesen hat, existiert Feedback für diese Artikel im Blackboard. Dies kann zur Verbesserung des Profils während der fortgesetzten Benutzung des Systems verwendet werden. Die einzelnen Agentenarten gehen dabei unterschiedlich vor:

Globalagent: Das Profil des Globalagenten p_{GA} soll das allgemeine Interesse des Benutzers abbilden. Daher benutzt der Globalagent alles Feedback im Blackboard. Zur Modifikation des Profils werden die Termvektoren der f Artikel, die gelesen oder bewertet wurden, gewichtet mit der Bewertung FB des Benutzers zu dem bisherigen Profil summiert (K ist die Artikelkollektion eines Tages):

$$\vec{p}_{GA}{}' = \vec{p}_{GA} + \frac{1}{f} \sum_{d \in K} FB(d) \cdot \vec{d}$$

Das Profil des Globalagenten wächst jeden Tag, da täglich neue Terme in der Artikelkollektion auftauchen. Daher wird für jeden Term im Profil gespeichert, wann er zuletzt bei einer Ähnlichkeitsberechnung benutzt wurde. Liegt dies mehr als τ_1 Tage zurück, wird der Term aus dem Profil entfernt. Ein Term wird ebenfalls als unwichtig betrachtet und gelöscht, wenn er vor mehr als τ_2 Tagen zuletzt benutzt worden ist und ein Gewicht kleiner γ besitzt. So wird das Wachstum des Profils gering gehalten und alte Interessen, die nicht mehr aktuell sind, verschwinden aus dem Profil.

Ressortagent: Die Profile dieser Agenten spiegeln das Interesse eines Benutzers an Themen eines Ressorts. Auch die Verarbeitung von Feedback beschränken Ressortagenten also auf das ihnen zugeordnete Ressort r. Ansonsten verläuft die Modifikation des Profils wie bei Globalagenten.

Themenagenten: Diese Agenten verfolgen einzelne Themen, an denen der Benutzer besonderes Interesse gezeigt hat. Ihr Profil ändern Themenagenten daher nur, falls Feedback zu Artikeln existiert, die dem Profil ähneln. Während des Filterprozesses merkt sich jeder Themenagent solche Artikel. Im Lernschritt wird zu jedem der gespeicherten Artikel im Blackboard die Bewertung abgefragt und der Termvektor des Artikels (entsprechend der Bewertung FB gewichtet) zum Profil p_{TA_i} hinzuaddiert.

5.3.1.2 Schwellwertlernen

Die Agenten benutzen jeweils einen Schwellwert θ_A ($A \in \{CA, GA, RA_r, TA_i\}$), um zu entscheiden, ob ein Artikel empfohlen wird. Bei Themenagenten hat sich der feste Schwellwert $\theta_{TA_i} = 0,12$ (analog zum Schwellwert des Clusterings) bewährt. Themenagenten beschränken sich auf ein sehr spezielles Thema und haben kleine, wenig streuende Profile. Globalagenten dagegen decken weite Interessenbereiche ab und besitzen ein breit gestreutes Profil. Deswegen wird der Schwellwert des Globalagenten anhand des Benutzerfeedbacks ständig angepaßt.

Im nächsten Abschnitt wird das ökomische System der Agentenpopulation ausführlich vorgestellt. Hier sei nur erwähnt, daß es Kosten K_{Empf} für das Empfehlen eines Artikels und Belohnungen bzw. Bestrafungen B_{Fb} für Feedback gibt.

Der Globalagent benutzt zur Bestimmung eines Empfehlungsschwellwertes θ_{GA} eine Rangordnung aller Artikel nach ihrer Ähnlichkeit mit dem Profil, die im Filterprozeß erstellt wird. Vom Artikel mit dem höchsten Ähnlichkeitswert abwärts wird der *Break-Even-Point* von Empfehlungskosten und Belohnungen berechnet. Hätte der Globalagent im zurückliegenden Pressespiegel alle Artikel oberhalb des Artikels am Break-Even-Point empfohlen, d.h. wäre der Ähnlichkeitswert dieses Artikel der Schwellwert θ_{GA} gewesen, so hätten sich Kosten und Belohnung die Waage gehalten. Es wird nun angenommen, daß der Ähnlichkeitswert des Artikels am Break-Even-Point sich auch als Schwellwert für die Bestimmung der Empfehlungen für den nächsten Pressespiegel eignet. Um Schwankungen des Benutzerverhaltens (und damit der Feedbackverteilung) an einzelnen Tagen auszugleichen, wird der Mittelwert der Ähnlichkeit des Artikels am Break-Even-Point und des alten Schwellwert als neuer Schwellwert verwendet.

5.3.2 Lernen im gesamten Agentensystem

Die Population der Agenten im *PZ*-System führt eine Art *konkurrierendes Lernen* durch. Jeder Agent lernt lokal, einen Teilbereich des Benutzerinteresses immer besser abzubilden. Im Rahmen des gesamten Multiagentensystem muß er sich auch gegen die anderen Agenten durchsetzen.

5.3.2.1 Erzeugung neuer Agenten

Falls ein Agent das Feedback zu einem Artikel beim Lernen benutzt, vermerkt er dies im Blackboard. Nachdem alle Agenten der Population gelernt haben, fragt die Steuerung im Blackboard den Bearbeitungsstatus der Feedbackdaten ab. Gibt es Artikel, deren Thema der Benutzer als interessant bewertet hat, die aber nicht von einem Themenagenten empfohlen wurden, wird ein neuer Themenagent gestartet. Das Profil dieses Agenten besteht anfänglich nur aus dem Termvektor des interessanten Artikels. Falls der Benutzer einen Artikel liest, der vom Global- oder von einem Themenagenten empfohlen wurde, oder im Kiosk sein Interesse am Thema eines Artikels ausdrückt, und für das Ressort dieses Artikel momentan kein Ressortagent existiert, wird ein Ressortagent für dieses Ressort erzeugt. Hier entsteht das initale Profil aus den allen Artikeln, die im aktuellen Pressespiegel aus diesem Ressort vom Benutzer bewertet wurden.

5.3.2.2 Belohnungs-/Bestrafungsmodell

Wie schon erwähnt, agieren die Agenten in einem ökonomischen System. Jeder Agent besitzt einen Etat E_A ($A \in \{CA, GA, RA_r, TA_i\}$), der abhängig von seiner Nützlichkeit für den Benutzer ist. Es wird angenommen, daß es für den Benutzer Aufwand bedeutet, die Schlagzeile und den Untertitel jedes Artikels auf den Ressortüberblicksseiten des Pressespiegels zu lesen und zu entscheiden, ob er den ganzen Artikel liest. Der Aufwand des Benutzers wird dem Agenten in Form von Kosten K_{Empf} für eine Empfehlung in Rechnung gestellt. Andererseits erzeugt der Agent auch Nutzen für den Benutzer, indem er interessante Artikel aus der großen Artikelkollektion herausfiltert. Diesem Nutzen entsprechen die Belohnungen B^{+}_{Fb} und B^{++}_{Fb}, die der Agent erhält, falls der Benutzer einen von ihm empfohlenen Artikel liest bzw. sogar als interessant bewertet. Dementsprechend ist B^{-}_{Fb} die Bestrafung für das Empfehlen eines Artikels, den der Benutzer als uninteressant bewertet.

Jeder Agent wertet das Feedback im Blackboard in Hinblick auf Belohnungen und Bestrafungen aus. Sein Etat für den nächsten Pressespiegel berechnet sich aus dem alten Etat abzüglich der Kosten für das Empfehlen von Artikel für den aktuellen Pressespiegel und der Bestrafungen für empfohlene, aber nicht als relevant befundene Artikel, zuzüglich der Belohnungen für vom Benutzer als relevant bewertete, von ihm empfohlene Artikel.

Der Etat bestimmt für jeden Agenten, wie viele Artikel er für einen Pressespiegel empfehlen darf. Die Kosten für eine Anzahl n^A_{Empf} Empfehlungen dürfen den zur Verfügung stehenden Etat E_A des Agenten A ($A \in \{CA, GA, RA_r, TA_i\}$) nicht überschreiten. Die Bedingung

$$E_A > n^A_{Empf} \cdot K_{Empf}$$

muß immer erfüllt bleiben. Reicht der Etat eines Agenten nicht mehr dazu, Artikel zu empfehlen, wird der Agent aus der Population entfernt.

Nur über diesen Belohnung-/Bestrafungsmechanismus lernt der Clusteragent, wieviele Empfehlungen er aussprechen darf. Zeigt der Benutzter Interesse an wichtigen Nachrichten, erhält der Agent Belohnungen und ist in der Lage, auch am nächsten Tag eine gewisse Anzahl Artikel vorzuschlagen. Interessieren den Benutzer eher spezifische Themen als die allgemeine Nachrichtenlage, wird der Clusteragent einen kleinen Etat haben und wenig vorschlagen.

Agenten, die nie etwas vorschlagen (und deshalb auch nie eine Belohnung bekommen), bieten keinen Nutzen für den Benutzer, können aber beliebig lange in der Agentenpopulation bleiben. Um dies zu verhindern, wurden zusätzlich die Kosten K_{Lesen} für das „Lesen“ der Artikelkollektion durch einen Agenten definiert. An jedem Tag verringert sich der Etat eines Agenten automatisch um diesen Wert. So werden Agenten, die nicht mehr aktuelle Themen- und Interessensbereiche abbilden und daher keine relevanten Artikel empfehlen können, langsam aus der Population entfernt.

5.3.2.3 Anpassung der Gewichte der Zusammenfassungskomponente

Die unterschiedlichen Konfidenzwerte zu Empfehlungen einzelner Agentenarten werden von der Zusammenfassungskomponente zu einem Konfidenzwert kombiniert. Jeder Agentenart zugeordnete Gewichte drücken den Stellenwert von Empfehlungen dieser Agentenart für die Rangordnung des Pressespiegels aus.

Im *PZ*-System wird ein sehr einfaches *Perzeptron* [Mitchell, 1997] für die Durchführung der Zusammenfassung und deren Anpassung an die Interessen des Benutzers benutzt. Das Perzeptron besteht aus fünf Eingabeeinheiten $x_1, \ldots, x_5$. Die letzten vier werden mit den Konfidenzwerte der vier Agentenarten belegt. Die erste Einheit ist die Schwellwerteinheit, an der immer der Wert 1 anliegt. Die Eingabeeinheiten sind mit einer einzelnen Ausgabeeinheit verbunden. Jede Verbindung ist mit einem Gewicht $g_1, \ldots, g_n$ versehen. Die Ausgabeeinheit summiert die mit dem entsprechenden Gewicht multiplizierten Werte der Eingabeeinheiten und normalisiert das Ergebnis mit Hilfe einer Ausgabefunktion $S(y) = \frac{1}{1+e^{-y}}$:

$$o = S\left(\sum_{n=1}^{5} g_n x_n\right)$$

Der Wert o wird als Konfidenzwert der zusammengefaßten Empfehlung verwendet (siehe Abbildung 5.3).

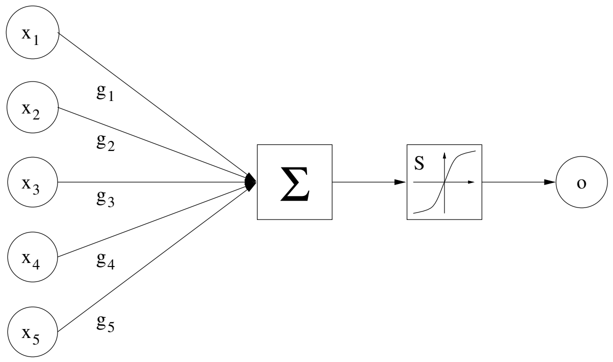


Abbildung 5.3: Das von der Zusammenfassungskomponente verwendete Perzeptron.

Zum Lernen wird zunächst ebenfalls der Ausgangswert o für die Konfidenzwerte der Empfehlungen zu einem Artikel d berechnet. Dieser wird mit der tatsächlichen Bewertung des Benutzers verglichen und der Fehler berechnet[4]:

[4] Da die Ausgabeeinheit des Perzeptrons nur Werte im Intervall (0..1) liefert, wird $FB_{Perzeptron}(d)$ wie folgt von $FB(d)$ abgeleitet:

$$FB_{Perzeptron}(d) = \begin{cases} 0,9 & \text{für} \quad FB(d) > 0 \\ 0,1 & sonst \end{cases}$$

$$\delta = o - FB_{Perzepton}(d)$$

Nun werden die Gewichte der Eingangseinheiten verändert, um den Fehler zu minimieren:

$$g_i' = g_i + \eta x_i \delta \quad \text{für} \quad i = 1, \ldots, 5$$

η ist eine kleine positive Konstante, mit der die Lernrate des Perzeptrons bestimmt werden kann.

Nacheinander wird dieser Prozeß für jeden Artikel durchlaufen. Diese Veränderung der Gewichte über alle Beispiele kann solange wiederholt werden, bis die Summe der Fehler für die Artikel minimal ist.

Experimente im *PZ*-System haben gezeigt, daß die optimale Anpassung der Gewichte an das Artikelfeedback nicht wünschenswert ist, da die Verteilung des Feedbacks auf die Agentenarten je nach Tag sehr unterschiedlich ist. Eine Minimierung des Perzeptronfehlers würde deshalb zu starken täglichen Schwankungen der Gewichte führen. Insbesondere besteht die Gefahr, daß das Gewicht einer Agentenart negativ wird, falls der Benutzer an einem Tag nur wenige Artikel liest, die von dieser Agentenart empfohlen wurden. Ein negatives Gewicht bedeutet die Verkehrung einer Empfehlung: je sicherer die Empfehlung für den Agenten ist, desto weniger relevant wird der Artikel für den Benutzer sein. Dies entspricht nicht dem generellen Ziel eines jeden Agenten, ein positives Abbild der Interessen des Benutzers darzustellen.

Aufgrund dieser Beobachtung werden im *PZ*-System nur eine festgelegte Anzahl Lernschritte für das Perzeptron durchgeführt. Das Perzeptron paßt sich zwar langsamer, dafür aber kontinuierlich an die Lesegewohnheiten des Benutzers an.

5.4 Vergleich mit anderen Informationsfiltern

In Kapitel 2 wurden bereits einige Informationsfilter mit verschiedenen Lernverfahren vorgestellt. In diesem Abschnitt sollen insbesondere Informationsfilter beschrieben werden, die als Multiagentensysteme aufgebaut sind. Die Methoden dieser Filter werden mit den Ansätzen, die im *PZ*-System verwirklicht sind, verglichen.

5.4.1 Informationsfilter von Baclace

Die Idee von miteinander konkurrierenden Agenten innerhalb eines Informationsfilters wurde zum ersten Mal von Paul Baclace formuliert [Baclace, 1992]. Sein Multiagentensystem besteht aus homogenen, nicht miteinander kommunizierenden Agenten. Das Profil eines Agenten besteht aus einer festen Klassifikationsregel, einem Vorhersagewert für die Bewertung der Dokumente, die durch die Regel positiv klassifiziert werden, und einem Etat. Die Repräsentation der Dokumente erfolgt durch Dokumentattribute wie Autor, Thema oder Schlüsselwörter. Die Klassifikationsregeln bestehen aus Konjunktionen von Attributwerten.

Gilt eine Regel für ein Dokument, so gibt der zugehörige Agent den Vorhersagewert als Bewertung des Dokuments aus. Für diese Bewertung wird dem Agenten ein kleiner Betrag von seinem Etat abgezogen. Die Bewertungen aller Agenten zu einem Dokument werden summiert und durch die Anzahl der Bewertungen geteilt. Für einen Block von Dokumenten wird so jeweils der Durchschnitt der Agentenbewertungen für jedes Dokument berechnet. Anschließend wird eine Rangordnung der Dokumente anhand ihres Ergebnisses erstellt, die dann dem Benutzer präsentiert wird.

Jede Bewertung des Benutzers für ein Dokument ergibt eine Belohnung in festgelegter Höhe. Die Agenten, deren Vorhersage nahe an der tatsächlichen Bewertung lag, teilen sich diese Belohnung. Die restlichen Agenten, die stark abweichende Bewertungen vorhersagten, müssen eine Strafe zahlen, die dann ebenfalls unter den gut vorhersagenden Agenten verteilt wird. Dieser Mechanismus führt dazu, daß nicht zu viele Agenten Vorhersagen zu einem Dokument erstellen. In diesem Fall würde die feste Belohnung so stark aufgeteilt werden, daß die Kosten für eine Bewertung höher wären als die Belohnung. Die Agenten bezahlen außerdem eine Art Miete, um in der Agentenpopulation zu bleiben. Sinkt der Etat eines Agenten auf null, wird er aus der Population gelöscht.

Die Agent lernen lokal nicht durch die Anpassung der Regel, sondern durch Justierung des Vorhersagewertes für die Bewertung in Richtung der tatsächlichen Bewertung des Dokuments. Agenten mit neuen Regeln werden durch eine Steuerkomponente erzeugt: Reagiert keiner der vorhandenen Agenten auf ein Dokument, werden neue Agenten mit Regeln für die Attributwerte des Dokuments erzeugt.

Baclace veröffentlichte keine Ergebnisse über die Leistungsfähigkeit seines Systems.

Viele Ideen aus diesem System sind in die Gestaltung nachfolgender Informationsfilter eingeflossen. Im *PZ*-System ist jeder Agent ebenfalls mit einem Etat ausgestattet, zahlt einen Betrag für das Empfehlen von Artikeln und das Verbleiben in der Population. Die Verteilung von Belohnungen ist jedoch unterschiedlich geregelt. Die größten Unterschiede der beiden Systeme liegen aber bei den verwendeten Filter- und Profillernverfahren. Hier benutzt das *PZ*-System das Vektorraummodell mit dem Rocchio-Algorithmus, während Baclaces System starre Regeln und die Anpassung des Vorhersagewertes anwendet.

5.4.2 Newt

Im News-Filteringsystem NewT [Sheth, 1994] erzeugt der Benutzer selber für einzelne Teilbereiche seines Interesses jeweils einen Agenten (z.B. einen Sport-, einen Wirtschaft- und einen Politikagenten). Der Benutzer kann dem Agenten interessante Artikel oder Artikelpassagen zeigen (*Programming by Demonstration*), oder den internen Zustand des Agenten direkt manipulieren.

Jeder Agent besteht aus einer Population von Profilen (typischerweise 20). Diese sind im Sinne von genetischen Algorithmen Klassifikationshypothesen für die Dokumente. Jedes Profil hat eine Reihe von Feldern (z.B. News-Gruppe, Autor, Text), die jeweils einen Termvektor mit $TFIDF$-Gewichtung beinhalten. Die Dokumente werden auf die gleiche Weise repräsentiert, so daß die Ähnlichkeit zwischen einem Profil und einem Dokument als Summe der Ähnlichkeiten der Termvektoren (nach dem Kosinusmaß) der verschiedenen Felder berechnet werden kann. Jedes Profil besitzt außerdem eine Gewichtung für die Felder, die ebenfalls in die Ähnlichkeitsberechnung einfließt. Die Anzahl der Dokumente, die ein Profil empfiehlt, ist abhängig von der Fitness des Profils. Die errechnete Ähnlichkeit skaliert mit der Fitness ist der Konfidenzwert für ein Dokument. Der Agent ordnet die Empfehlungen aller Profile zu einem Ranking und präsentiert dem Benutzer die obersten n Dokumente.

Das Feedback des Benutzers für die angezeigten Dokumente wird zum einen mittels des Rocchio-Algorithmus zur Veränderung der Termvektoren der Profilfelder benutzt. Andererseits beinflußt das Feedback die Fitness der Profile. Gute Bewertungen erhöhen die Fitness, schlechte senken sie.

Die Fitness entscheidet, ob ein Profil weiter in der Population bleibt oder entfernt wird. In jeder Generation werden die r Profile mit der höchsten Fitness bewahrt und der Rest gelöscht. Den freigewordenen Platz in der Population nehmen c durch Crossover und m durch Mutation erzeugte Profile ein. Beim Crossover tauschen zwei Profile Termvektoren aus einer Auswahl ihrer Felder aus und erzeugen so zwei neue Profile. Ein Profil erzeugt durch Mutation ein neues Profil, indem im News-Gruppen-Feld des Profils eine dort vorhandene News-Gruppe durch eine zufällig gewählte andere ersetzt wird. Alle Parameter des Genetischen Algorithmus (r, c, m und die Populationsgröße) lassen sich vom Benutzer einstellen.

Mit NewT wurden Tests mit fiktiven, simulierten Benutzern durchgeführt und das System über mehrere Wochen von mehreren Anwendern benutzt. Das System konnte seine Fähigkeit zur Anpassung an neue Benutzer unter Beweis stellen. Veränderungen des Interesses mit der Zeit konnten ebenfalls verfolgt werden.

NewT legt viele Steuerungsmöglichkeiten in die Hände des Benutzers. Das *PZ*-System hingegen versucht, als *Learning Apprentice* die Interessen des Benutzers schon der Anwendung des Systems zu erkennen, und erfordert keinen großen zusätzlichen Bedienaufwand von Seiten des Benutzers.

Die Profile des NewT-Systems entsprechen den Ressortagenten des *PZ*-Systems. Beide beschränken sich auf Teile der Dokumentkollektion. Eine Entsprechung zum Globalagent ist bei NewT praktisch nicht möglich, da die Dokumentkollektion (alle UseNet-News-Artikel eines Tages) sehr groß ist. Themenagenten und Clusteragent besitzen ebenfalls keine Pendants in NewT.

Lernen geschieht in NewT durch einen Genetischen Algorithmus. Dieser hat den Nachteil, durch die zufallsgesteuerte Erzeugung neuer Profile mittels Crossover und Mutation oft sehr schlechte Profile zu erzeugen, die der Benutzer über negatives Feedback wieder aus der Population entfernen muß. Das *PZ*-System geht zielgerichteter vor wenn es neue Ressort- und Themenagenten erzeugt.

5.4.3 Amalthea

Dieses System ist – genau wie NewT – am MIT Media Lab entstanden. Der Ansatz von NewT, eine Population von Agenten einer Evolution zu unterwerfen, wird in Amalthea weiterverfolgt und erweitert [Moukas, 1996],[Moukas und Zacharia, 1997].

Es gibt zwei Arten von Agenten: Informationsfilter und Informationsentdecker. Während erstere den Agenten im *PZ*-System entsprechen und ähnliche Verfahren zur Filtering anwenden, dienen letztere dazu, das System ständig mit neuen Dokumenten zu versorgen.

Die zwei Agentenarten arbeiten eng zusammen. Informationsfilteragenten senden Anfragen in Form von Termvektoren an die Entdeckeragenten. Ein solcher Agent entscheidet aufgrund seiner bisherigen Interaktionserfahrung mit dem Sender der Anfrage, ob er diese bearbeitet. Bei einer positiven Entscheidung stellt er eine Anfrage an eine WWW-Suchmaschine, die aus einigen Wörtern des übergebenen Termvektors besteht. Die Ergebnisse der Suche werden anschließend dem Filteragenten präsentiert. Dieser wählt gemäß seinem Profilvektor das beste Dokument aus und empfiehlt dies mit einem Konfidenzwert, der proportional zu der Ähnlichkeit des Dokuments mit seinem Profil und proportional zu seiner Fitness ist. Von den Empfehlungen allerr Agenten werden dem Benutzer nur eine einstellbare Zahl präsentiert.

Feedback des Benutzers wirkt gewichtet mit der Konfidenz des Agenten für seine Empfehlung als Belohnung oder Bestrafung für einen Filteragenten. Dieser gibt einen Teil davon weiter an den Informationsentdecker, der das bewertete Dokument beschafft hat. Empfehlen zwei Agenten das gleiche Dokument, erhalten sie eine Bestrafung. So bleibt die Vielfalt der Agentenpopulation gewährleistet.

Auch in Amalthea werden die genetischen Operatoren Crossover und Mutation benutzt. Die Evolution der zwei Agentenarten verläuft dabei getrennt. Das Feedback des Benutzers kontrolliert die Lernrate des Genetischen Algorithmus: Steigt das negative Feedback stark an, werden pro Generation mehr Agenten aus dem System gelöscht, damit die Anpassung an das (wahrscheinliche geänderte) Benutzerinteresse schneller erfolgt. Ist das Feedback überwiegend positiv, erniedrigt das System die Zahl der zu entfernenden Agenten.

Bei Testläufen mit simulierten Benutzerprofilen stellte sich heraus, daß über 50 Generationen des GA nötig waren, um die Leistung des Systems zu optimieren. In jeder Generation mußten 20 Dokumente bewertet werden. Amalthea gelang dann die Anpassung an sich langsam verändernde Interessen sehr gut. Nach abrupteren Veränderungen des Interesses sank die Filterleistung des Systems für einige Generationen stark ab, stieg aber wieder schnell auf das vorherige Niveau.

Im Unterschied zum *PZ*-System behandelt Amalthea auch die Informationssuche durch Agenten. Es gibt aber keine Filteragenten mit verschiedenen Filterstrategien wie sie im *PZ*-System existieren. Die Benutzung eines GA als Lernverfahren führt – wie schon bei NewT – dazu, daß eine Reihe von Generationen durchlaufen werden muß, bis das System sich an den Benutzer angepaßt hat. Dies bedeutet für den Benutzer, daß er sehr viele Dokumente bewerten muß, bis die Anpassung an seine Interessen vollzogen ist.

5.4.4 SIGMA

Das Multiagentensystem in SIGMA [Karakoulas und Ferguson, 1995] [Ferguson und Karakoulas, 1996] hat den komplexesten Aufbau aller hier beschriebenen Systeme. Es gibt Agenten zur Dokumentbeschaffung (UW), zur Dokumentspeicherung (RM) und zur Anzeige des Filterergebnisses für den Benutzer (IM). Die wichtigsten Agentenarten jedoch sind die FE, PG, PS und BM Agenten. FE-Agenten (*Feature Extractor*) setzen Dokumente in die Termvektordarstellung um. PG-Agenten (*Profile Generator*) pflegen ein Profil in Termvektordarstellung, daß anfangs aus einer vom Benutzer eingegebenen kurzen Beschreibung seines Interesses besteht. Jeder PG-Agent sucht über FE-Agenten nach Artikeln, die seinem Profil ähneln, und bietet diese anderen Agenten an. PS-Agenten (*Profile Selector*) repräsentieren ein bestimmtes Interesse eines Benutzers. Jeder PS-Agent besitzt ein Budget, mit dem er Dokumente von PG-Agenten kaufen kann. Die Preise der Dokumente werden vom BM-Agenten (*Bid Manager*) je nach Angebot und Nachfrage im Markt der PG- und PS-Agenten festgelegt. Die vom PS-Agenten erworbenen Dokumente werden dem Benutzer angezeigt.

Benutzerfeedback beinflußt in SIGMA die PS- und PG-Agenten. Mit Reinforcement Learning lernen PS-Agenten mit der Zeit, von welchen PG-Agenten sie die interessantesten Dokumente kaufen können. Die PG-Agenten lernen mit dem Rocchio-Algorithmus ein verbessertes Profil. Die Bewertungen des Benutzers bestimmen ebenfalls die Belohnung, die der PS-Agent erhält. Über den Kauf von Dokumenten gelangt die Belohnung bei der nächsten Transaktion weiter zu den PG-Agenten.

Es sind noch keine Testergebnisse für SIGMA bekannt.

Das *PZ*-System besitzt nicht mehrere FE-Agenten, sondern stellt allen Agenten die Daten der gesamten Artikelkollektion im Blackboard zur Verfügung. Anders als SIGMA fragt das *PZ*-System vom Benutzer keine Beschreibung seines Interesses ab, um Agenten zu initialisieren, sondern beobachtet die Reaktion auf einen benutzerunabhängig zusammengestellten Anfangspressespiegel. Im *PZ*-System ist kein Mechanismus zur dynamischen Preisbestimmung vorhanden, die Kosten für das Empfehlen eines Artikels sind festgelegt.

Kapitel 6

Evaluation des *PZ*-Systems

In diesem Kapitel soll ausgewertet werden, inwieweit das *PZ*-System die Ziele

- Fokussierung der Benutzeraufmerksamkeit auf interessante Artikel,
- selbständige Anpassung an die Interessen des Benutzers,
- Abbildung und zeitliches Verfolgen der verschiedenen Bereiche des Benutzerinteresses und
- leichte Bedienbarkeit

erfüllt, die in Kapitel 5 definiert wurden.

Die Auswertung erwies sich als schwierig, da die Ziele sehr informal definiert sind. Es ergeben sich keine einfachen quantitativen Maße, die den Grad der Zielerreichung angeben, direkt aus der Definition der Ziele. Obwohl im folgenden einige Gütemaße eingeführt werden, bleiben viele Aussagen zur Evaluation sehr qualitativ.

Eine weitere Schwierigkeit für die Auswertung besteht darin, daß das individuelle Verhalten der Anwender bei der Benutzung des Systems sehr unterschiedlich ist. Um also allgemeine Aussagen über das *PZ*-System machen zu können, wären Tests mit vielen Benutzern notwendig. Dies war im Rahmen der vorliegenden Arbeit nicht möglich. Die vorgestellten Ergebnisse des Systembetriebs sind also nur als Beispiele für das Systemverhalten zu betrachten.

Aus der Verwendung einer Vielzahl an Lernverfahren im *PZ*-System ergibt sich eine dritte Schwierigkeit. Jedes Verfahren führt neue Parameter ein, die das Verhalten des Systems beeinflussen. Um die optimalen Einstellungen zu bestimmen, wären viele Testläufe mit verschiedenen Parametersets nötig. Ist eine große Datenbasis vorhanden und existiert ein einfach berechenbares Qualitätsmaß für das System, kann diese Optimierung automatisch erfolgen (siehe [Boyan et al., 1996]). Im *PZ*-System sind diese beiden Vorausetzungen jedoch nicht erfüllt. Die Einstellung der Parameter erfolgte daher heuristisch auf der Grundlage einiger

Testläufe und einer dem Systembetrieb vorhergehenden Pilotphase. Die Erfahrungen aus dem Systembetrieb zeigen denn auch, daß hier weitere Anpassungen nötig sind.

Schwerpunkt der Evaluation ist die Leistung der Filterkomponente. Zu ihrer Bewertung wurden verschiedene Methoden angewendet.

Zum einen wurden Tests mit *fiktiven Benutzern* durchgeführt. Diese Benutzer besitzen fixierte Interessen, die durch eine Menge von relevanten Artikeln festgelegt sind. Die Definition von Relevanz für die gesamte Kollektion ist notwendig, um quantitative Aussagen über die Filterleistung machen zu können.

Zum zweiten wurde der *Systembetrieb* über 3 Wochen beobachtet. Sieben Benutzer ließen sich während dieser Zeit einen persönlichen Pressespiegel vom *PZ*-System zusammenstellen. Leider wurden die Pressespiegel nicht von allen Benutzern regelmäßig gelesen, so daß nur hier nur für drei Benutzer Ergebnisse präsentiert werden können.

Die geringe Anzahl Daten aus dem Systembetrieb verhinderte eine Evaluation der Benutzungsoberfläche. Aus Gesprächen mit den Benutzern ist jedoch zu schließen, daß diese gut mit dem System zurecht kamen. Die Navigation innerhalb des Pressespiegels und des Kiosk sollte noch geringfügig verbessert werden. Dem Benutzer sollte deutlicher angezeigt werden, welche Artikel er bereits gelesen hat. Bisher geschieht dies alleine durch den WWW-Browser des Benutzers, der schon besuchte Hyperlinks farblich markiert. Bei den Seiten des *PZ*-Systems, die durch CGI-Skripte dynamisch erzeugt werden, führt dies u.U. aber nicht zu den gewünschten Ergebnissen. Diese Funktionalität sollte in weiteren Versionen des *PZ*-Systems in die Anzeigekomponente eingebaut werden.

Nicht unerheblich für eine Informationsfiltersystem ist die Frage der Laufzeit. Geschieht das Filtern sehr schnell, kann das System zum Online-Filtern eingesetzt werden, andernfalls läßt sich das System nur offline im Batchbetrieb verwenden.

Das *PZ*-System ist als Offline-Filter gestaltet und wird jeden Tag nachts um 2 Uhr automatisch gestartet. Der gesamte Prozeß vom Holen der Seiten über das Internet bis zur Erstellung der Pressespiegel dauert durchschnittlich 45 Minuten. Mit knapp 20 Minuten den größten Anteil daran hat das Holen der HTML-Seiten von den WWW-Servern der Tageszeitungen. Die Textextraktion benötigt im Durchschnitt eine Minute. Der zweite große Zeitanteil gehört der morphologischen Analyse und der Termgenerierung. Dies wird in 15 Minuten erledigt. Die anschließende $TFIDF$-Gewichtung geschieht in weiteren vier Minuten. Der Lern- und Filterprozeß innerhalb des Multiagentensystems inklusive des Clusterings, das im selben Programm berechnet wird, benötigt etwa 5 Minuten. Die Zeiten wurden auf einer SparcStation IPX füer die Erstellung von Pressespiegeln für 14 Benutzer gemessen. Die Artikelkollektionen enthalten täglich etwa 200 Artikel.

Der Rest des Kapitels beschreibt die Ergebnisse der Tests mit fiktiven Benutzern (Abschnitt 6.1) und des Systembetriebs (Abschnit 6.2). In beiden Abschnitten werden Pressespiegel aus den Artikeln von vier Tageszeitungen erstellt. Im einzelnen sind dies:

1. Passauer Neue Presse (URL `http://www.vgp.de`)
2. Schweriner Volkszeitung (URL `http://www.svz.de`)
3. taz, die Tageszeitung (URL `http://www.taz.de`)
4. DIE WELT (URL `http://www.welt.de`)

Da im Testzeitraum (22. 4. - 14. 5. 1997) zwei Feiertage lagen, an denen drei Zeitungen nicht erschienen, eine Zeitung zum 1. 5. 1997 das Format ihrer HTML-Seiten änderte (was im automatischen Betrieb erst zu spät bemerkt wurde) und zeitweilig Probleme mit den WWW-Servern der Zeitungen auftraten, sind nicht in jeder Artikelkollektion alle Zeitungen vertreten.

Die oben schon erwähnten Gütemaße zur Beurteilung der Filterqualität sind aus dem Information Retrieval entlehnt. Jedes Dokument der Kollektion muß dafür gemäß dem Benutzerinteresse als relevant oder nicht relevant klassifiziert sein. Das IR- bzw. IF-System führt seinerseits eine Klassifikation der Dokumentkollektion durch. Nun kann folgende Tabelle aufgestellt werden:

		Benutzerklassifikation	
		Relevant	Nicht Relevant
Systemklassifikation	Relevant	A	B
	Nicht Relevant	C	D

A, B, C und D sind jeweils Mengen von Dokumenten. Die Maße *Precision* und *Recall* lassen sich jetzt folgendermaßen berechnen:

$$Precision = \frac{|A|}{|A| + |B|}$$

$$Recall = \frac{|A|}{|A| + |C|}$$

Precision ist also der Anteil der für den Benutzer interessanten Dokumente an den Dokumenten, die das System als relevant beurteilt. Das Verhältnis zwischen den vom System gefundenen und allen für den Benutzer relevanten Dokumenten wird als Recall bezeichnet.

Die Berechnung beider Maße für die Filterergebnisse bei fiktiven Benutzern ist möglich, da dort eine Relevanzbeurteilung für alle Dokumente vorliegt. Aus dem Betrieb liegt eine solche Beurteilung für die gesamte Kollektion nicht vor, deshalb ist in diesem Fall nur die Precision berechenbar. Dieses Maß ist dort jedoch mit

Vorsicht zu bewerten. In einem Pressespiegel sind oft mehrere Berichte zu einem Thema (also aus einem Cluster) vorhanden. Die Benutzer lesen aber meist nicht alle dieser Berichte. Durch implizites Feedback wird also die Relevanz der Artikel für den Benutzer nicht korrekt wiedergegeben. Das Feedback bildet aber eine untere Grenze für die Relevanz. Die in Abschnitt 6.2 berechneten Precision-Werte sind also als untere Abschätzung des wirklichen Wertes zu betrachten.

6.1 Tests mit fiktiven Benutzern

Zur Durchführung von nachvollziehbaren und vollständig auswertbaren Tests muß das Benutzerinteresse festgelegt werden. Dieses fixierte Interesse definiert die fiktiven Benutzer. Für diese Benutzer wird das Lesen des Pressespiegels simuliert, indem Feedback füer die relevanten Artikel erzeugt wird. Diese Technik wurde bereits mit Erfolg zur Evaluation eines Informationsfilters eingesetzt [Sheth, 1994].

Da die Erstellung einer kompletten Relevanzbeurteilung für die Testdaten (2751 Artikel aus dem Zeitraum vom 22. 4. 1997 bis zum 9. 5. 1997) durch einen menschlichen Experten wegen des zu großen Aufwands nicht möglich war, wurden über eine einfache Schlüsselwortsuche[1] Mengen von relevanten Dokumenten für sechs unterschiedliche Themenbereiche bestimmt. Tabelle 6.1 zeigt die verwendeten Schlüsselwörter und die Anzahl relevanter Dokumente. Einige Artikel sind in mehreren Themenbereichen vertreten, da die Mengen sich überschneiden.

Themenbereich	Schlüsselwort	Anzahl Dokumente
T1	steuer	213
T2	euro	71
T3	kohl	135
T4	spd	202
T5	zaire	57
T6	korea	62

Tabelle 6.1: Schlüsselwörter und Grö:se der Beispielinteressen.

Zwei verschiedene Testreihen mit verschiedenen Zielen wurden durchgeführt:

A Das Benutzerinteresse setzt sich über den gesamten Testzeitraum aus allen sechs Themen zusammen. Durch diese Testreihe kann die Anpassungsfähigkeit des Systems an unterschiedliche Interessensbereiche festgestellt werden.

B Das Benutzerinteresse besteht in der ersten Hälfte des Testzeitraums aus nur drei Themen (T1, T3, T5). Zu Beginn der zweiten Hälfte wechselt das

[1] Es wurde das UNIX-Kommando `grep` auf die Artikelkollektionen angewandt.

Benutzerinteresse abrupt auf die restlichen Themen (T2, T4, T6). Diese Tests sollen die Fähigkeit des Systems zur Verfolgung wechselnder Interessen testen.

6.1.1 Testreihe A

Zwei verschiedene Tests wurden durchgeführt. Im ersten Test wurde nur das einfache Lesen aller relevanten Artikel simuliert. Für den zweiten Test wurde für jedes Thema am 22. 4. der Artikel mit der größten Vorkommenshäufigkeit als„besonders interessant“ bewertet. Dadurch wird im Multiagentensystem für jeden Themenbereich ein Themenagent gestartet. Im folgenden werden die Ergebnisse des zweiten Tests vorgestellt und anschließend kurz mit den Ergebnissen des ersten Tests verglichen.

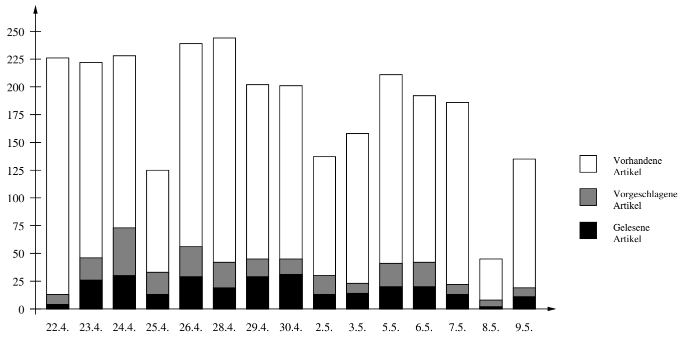


Abbildung 6.1: Anzahl vorhandener/empfohlener/gelesener Artikel pro Tag (Testreihe A)

Abbildung 6.1 zeigt im Balkendiagramm die Anzahl der vorhandenen Artikel pro Tag. Der graue Balken repräsentiert die Anzahl der vom *PZ*-System empfohlenen Artikel, während die Größe des schwarzen Balkens der Anzahl der davon gelesenen Artikel entspricht. Es ist deutlich zu sehen, daß das System einen Großteil der Artikel ausfiltert.

Wie gut ist nun der erzeugte Pressespiegel? Abbildung 6.2 zeigt das Verhältnis von empfohlenen (durchgezogene Linie) und gelesenen (gestrichelte Linie) Artikeln zur Größe der Artikelkollektion. An den ersten zwei Tagen steigt die Kurve steil an, das System führt die Anpassung durch und bekommt viel Feedback. Die Agenten merken nun, daß der Pressespiegel zu groß wird, und verringern die Anzahl ihre Empfehlungen wieder. Zu dieser Entwicklung trägt auch bei, daß die Anzahl der relevanten Dokumente in der Kollektion zum Ende des Testzeitraumes geringer wird. Der Prozentsatz der gelesenen Artikel spiegelt diese Verringerung ebenfalls wieder.

In Abbildung 6.3 ist die zeitliche Entwicklung der beiden Maße Precision und Recall (jeweils die durchgezogene Linie) gezeigt. Schon fast von Beginn an

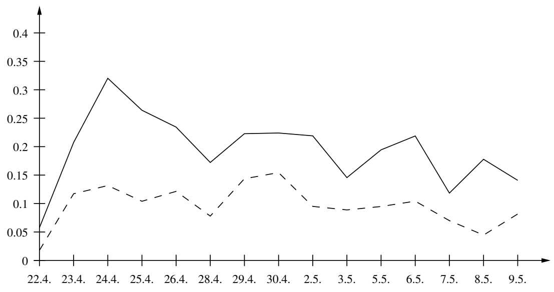


Abbildung 6.2: Verhältnis zwischen der Anzahl empfohlener bzw. gelesener Artikel und der Größe der Artikelkollektion (Testreihe A)

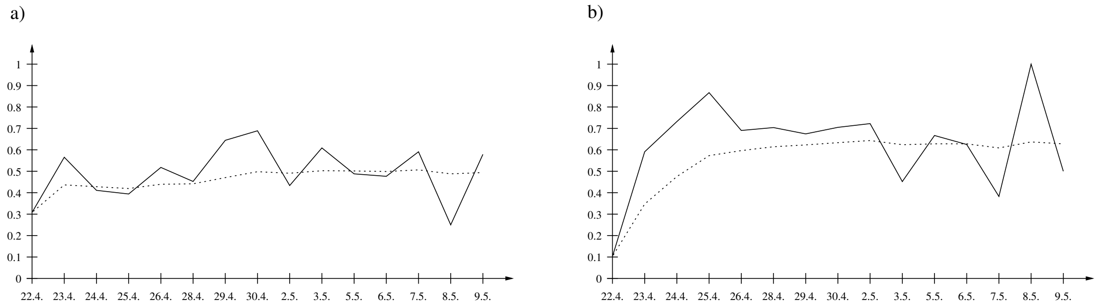


Abbildung 6.3: a) Precision und b) Recall für Testreihe A

ist etwas weniger als die Hälfte der empfohlenen Dokumente für den Benutzer relevant (Abbildung6.3a). Der Anfangswert der Precision von 0,31 zeigt, daß die Interessen des Benutzers momentan wichtige Themen in allen Zeitungen sind, da der erste Pressespiegel ja ohne Wissen über den Benutzer erstellt wird. Der Mittelwert der Precision (gestrichelte Linie) ist am Ende des Testzeitraumes 0,49.

Der Recallgraph (Abbildung6.3b) zeigt die steigende Fähigkeit des Systems, die relevanten Dokumente aus der Kollektion herauszufiltern. Innerhalb von drei Tagen ist ein Recall-Niveau von über 0.6 erreicht. Der Ausbrecher der Kurve am 8. 5. ist darauf zurückzuführen, daß die Kollektion an diesem Tag nur zwei relevante Dokumente enthielt und nur eines empfohlen wurden. Dies ergibt einen Recall von 0.5. Der Mittelwert des Recall (gestrichelte Linie) ist am Ende des Testzeitraumes 0.63.

Der erste Test ergab im Vergleich dazu einen Mittelwert der Precision von 0,49 und einen Mittelwert des Recall von 0,59. Der Anteil an relevanten Dokumenten im ohne Themenagenten erstellten Pressespiegel ist also genauso hoch wie im zweiten Test. Durch den Einsatz der Themenagenten steigt jedoch die Fähigkeit des Systems, die relevanten Dokumente aus der Kollektion herauszufinden.

6.1.2 Testreihe B

Auch hier wurden zwei Tests (ohne und mit Berücksichtigung von Themenagenten) durchgeführt. Die Änderung der Interessen erfolgte am 1. 5. 1997 .Wieder ergibt sich eine gleichhohe Precision und ein geringfügig kleinerer Recall des ersten Testlaufs zum zweiten Testlauf, der im folgenden ausführlich beschrieben wird.

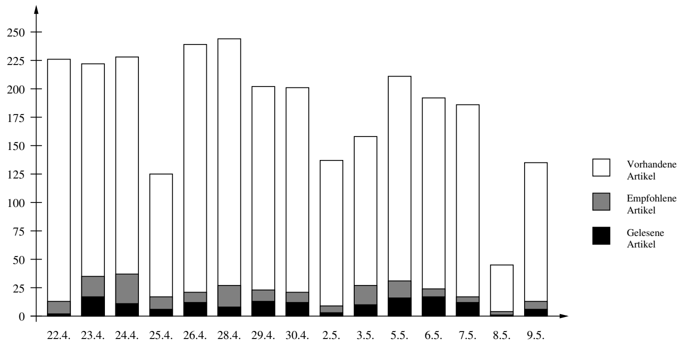


Abbildung 6.4: Anzahl vorhandener/empfohlener/gelesener Artikel pro Tag (Testreihe B)

Das Balkendiagramm (Abbildung 6.4) zeigt, daß entsprechend der geringeren Zahl relevanter Dokumente (im Verhältnis zu Testreihe A etwa 2/3) weniger Dokumente empfohlen werden. Die Precision- und Recall-Graphen in Abbildung 6.5

veranschaulichen das Systemverhalten genauer. Obwohl die Interessen des fiktiven Benutzers enger gefaßt sind, ist die Precision (Abbildung 6.5a) im Durchschnitt (Endwert 0,44) geringer als in Testreihe A. Die Schwankungen der Werte sind ebenfalls deutlich stärker.

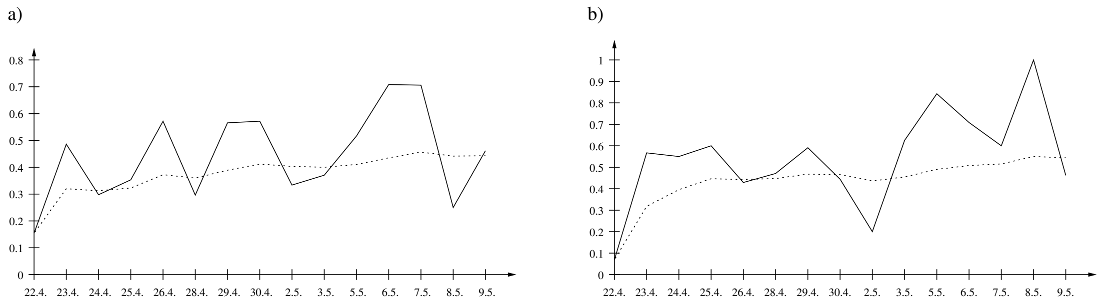


Abbildung 6.5: a) Precision und b) Recall für Testreihe B

Der Einbruch der Filterleistung nach der Veränderung der Interessen ist klar am Recall-Graphen zu erkennen (Abbildung 6.5b). Durch den Start neuer Themenagenten kann das System aber sofort das Filterergebnis wieder verbessern. Der Recall in der zweiten Hälfte liegt durchweg höher als in der ersten Hälfte. Der Mittelwert des Recall liegt aber mit 0,54 (0,49 ohne den Einsatz von Themenagenten) deutlich unter dem von Testlauf A.

6.1.3 Auswertung

In den beiden Testläufen mit fiktiven Benutzern konnte nachgewiesen werden, daß das System innerhalb kurzer Zeit in der Lage ist, die Interessen des Benutzers zu erfassen und entsprechende Artikel aus der Kollektion zu empfehlen. Der Einsatz von Themenagenten erhöht den Anteil der relevanten Artikel, die gefunden werden. Die Abbildung eines breiter gestreuten Interesses in Testlauf A gelang besser, als die Fokussierung auf eine sehr kleine Anzahl Themen. Auch ein abrupter Wechsel des Benutzerinteresses konnte vom System ausgeglichen werden.

6.2 Betrieb des Systems

Nach den ermutigenden Ergebnissen der Tests mit fiktiven Benutzern sollen in diesem Abschnitt die Erfahrungen aus dem Systembetrieb beschrieben werden.

Für drei Benutzer lagen genügend Daten vor, um eine sinnvolle Bewertung der Filterleistung durchzuführen. Die Benutzer unterscheiden sich in ihrem Leseverhalten. Dementsprechend ist auch das resultierende Verhalten des Systems verschieden.

6.2.1 Benutzer 1

Benutzer 1 (siehe Tabelle 6.2) liest den Pressespiegel, ohne viel explizites Feedback zu geben. Zu Beginn des Testzeitraumes wies er das *PZ*-System im Kiosk auf einige seiner Interessen hin, verringerte aber diese Art von *Programming by Demonstration* im weiteren Verlauf.

	Pressespiegel			Kiosk
Tag	# gelesen	# interessant	# uninteressant	# Hinweise
23.4.	5	0	0	3
24.4.	11	0	0	8
25.4.	6	0	0	1
26.4.	5	0	0	3
28.4.	9	0	0	0
29.4.	5	0	0	1
30.4.	5	1	0	3
2.5.	12	1	1	0
3.5.	9	1	0	2
5.5.	9	1	0	1
6.5.	9	1	0	0
8.5.	2	1	1	0
9.5.	10	0	0	0

Tabelle 6.2: Aufteilung des Benutzerfeedbacks für Benutzer 1.

In Abbildung 6.6 ist zu erkennen, daß der Pressespiegel für Benutzer 1 täglich aus etwa 40 Artikeln besteht. Für Benutzer 1 ist keine Anfangslernphase zu erkennen, da dieser Benutzer schon vor dem Testzeitraum in einer Pilotphase das *PZ*-System angewendet hat.

Die Anzahl der empfohlenen Artikel steht in einem stabilen Verhältnis zu der Größe der Artikelkollektion (siehe Abbildung 6.7a), durchgezogene Linie). Der leichte Anstieg zum Ende des Testzeitraumes hin erklärt sich aus dem ebenfalls angestiegenen Verhältnis an relevanten Artikeln in der Kollektion (Abbildung 6.7a) gestrichelte Linie).

Der Precision-Graph in Abbildung 6.7b) zeigt, daß im Durchschnitt nur ein Fünftel des Pressespiegels gelesen wird (Mittelwert der Precision: 0,21) Dieser Anteil bleibt aber über den ganzen Zeitraum – bis auf die Abweichung am 2. 5. –relativ konstant.

Anhand der Testdaten von Benutzer 1 wurde untersucht, inwieweit das implizite Feedback die korrekte Berechnung der Precision erlaubt. Dazu wurden nicht nur die wirklich gelesenen Artikel als relevant betrachtet, sondern auch Artikel aus Clustern, in denen sich gelesene Artikel befinden. Wird die Precision über

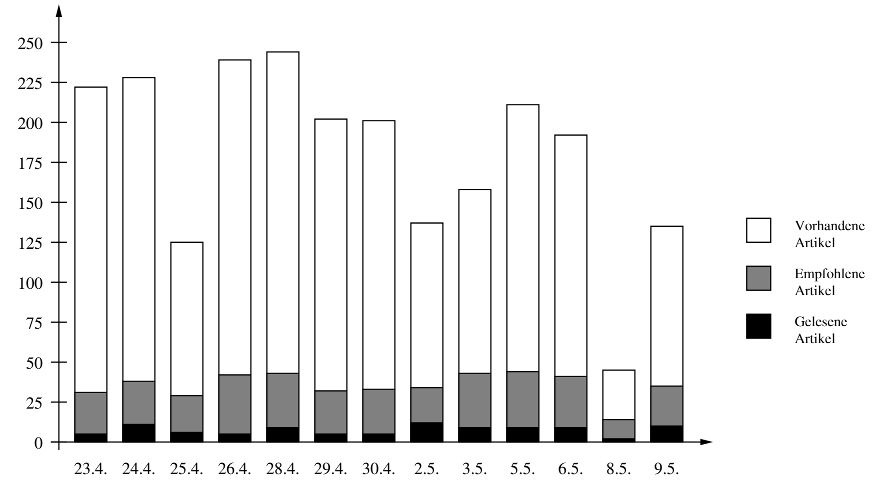


Abbildung 6.6: Anzahl vorhandener/empfohlener/gelesener Artikel pro Tag (Benutzer 1)

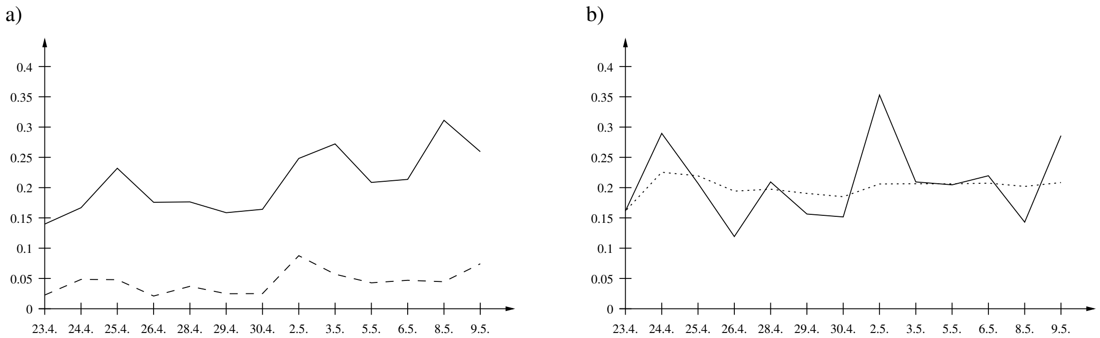


Abbildung 6.7: a) Verhältnis zwischen der Anzahl empfohlener bzw. gelesener Artikel und der Größe der Artikelkollektion b) Precision (Benutzer 1)

diesen angepaßten Feedbackdaten berechnet, ergibt sich ein Mittelwert von 0,34. Dies zeigt, daß die alleine durch das implizite Feedback berechnete Precision nur eine sehr grobe untere Abschätzung bietet.

6.2.2 Benutzer 2

Benutzer 2 nutzte viel stärker als Benutzer 1 die Möglichkeiten des expliziten Feedbacks (siehe Tabelle 6.3). Zu Beginn der Benutzung wurden viele Artikel im Kiosk gelesen und bewertet, um das System auf Interessen hinzuweisen. Fast immer wurden die gelesenen Artikel auch als „besonders interessant" markiert.

	Pressespiegel			Kiosk
Tag	# gelesen	# interessant	# uninteressant	# Hinweise
6.5.	1	0	0	0
9.5.	2	2	0	17
10.5.	3	2	1	7
12.5.	10	10	0	8
13.5.	6	3	0	0
14.5.	8	8	0	4

Tabelle 6.3: Aufteilung des Benutzerfeedbacks für Benutzer 2.

An den Feedbackdaten von Benutzer 2 kann man die Schwierigkeiten in der Anfangsphase der Benutzung des *PZ*-Systems erkennen. Im Balkendiagramm (Abbildung 6.8) sieht man, daß das System anfangs nur langsam die Anzahl seiner Empfehlungen erhöht. Das starke positive Feedback am 12. 5. erhöht jedoch die Etats der Agenten sehr stark, so daß am 13. 5. ein deutlicher Anstieg der Anzahl empfohlener Artikel zu beobachten ist. Wie in Abbildung 6.9b) zu erkennen ist, führt dies aber zu einem Absinken der Precision, da zu viele irrelevante Artikel in den Pressespiegel gelangen. Am nächsten Tag sinkt die Anzahl der empfohlenen Artikel und die Precision steigt wieder. Der Mittelwert der Precision liegt bei 0,21, es ist aber fraglich, welche Bedeutung diese Zahl angesichts der starken Schwankungen hat.

6.2.3 Benutzer 3

Bei Benutzer 3 zeigt sich, daß die Parametereinstellungen im PZ-System nicht optimal sind. Dieser Benutzer gibt stetig positives und negatives explizites Feedback für den Pressespiegel. Außerdem gibt er im Kiosk viele Hinweise zu interessanten Artikeln. Die hohe Anzahl gelesener Artikel am Ende des Testzeitraums und die gesunkene Anzahl der im Kiosk bewerteten Artikel weisen erst einmal darauf hin, daß das *PZ*-System sich dann an die Interessen des Benutzers angepaßt hat.

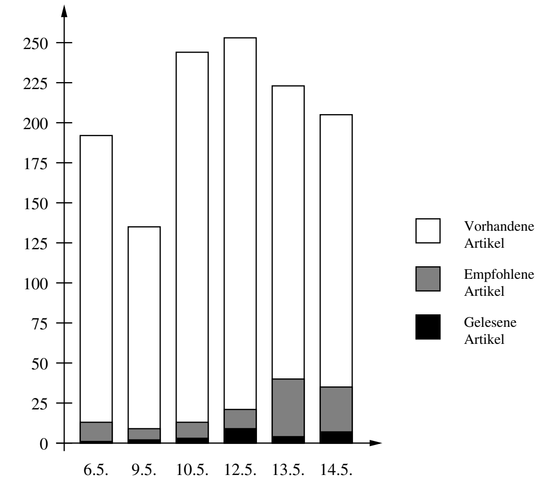


Abbildung 6.8: Anzahl vorhandener/empfohlener/gelesener Artikel pro Tag (Benutzer 2)

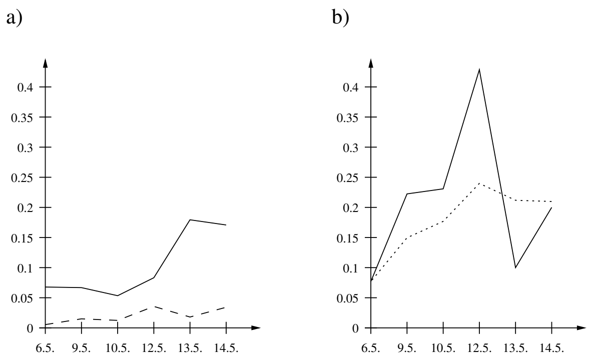


Abbildung 6.9: a) Verhältnis zwischen der Anzahl empfohlener bzw. gelesener Artikel und der Größe der Artikelkollektion b) Precision (Benutzer 2)

Tag	Pressespiegel			Kiosk
	# gelesen	# interessant	# uninteressant	# Hinweise
24.4.	7	3	1	12
25.4.	11	2	2	5
28.4.	10	1	1	5
29.4.	12	2	1	3
30.4.	7	0	0	8
2.5.	8	2	2	13
5.5.	11	0	0	1
6.5.	10	1	1	10
7.5.	6	2	1	11
9.5.	2	0	0	3
12.5.	17	0	0	0
13.5.	21	3	2	0

Tabelle 6.4: Aufteilung des Benutzerfeedbacks für Benutzer 3.

Allerdings ist der Pressespiegel für diesen Benutzer sehr groß. Wie in den Abbildungen 6.10 und 6.11a) zu sehen ist, werden fast die Hälfte der Artikel in der Kollektion empfohlen. Die Analyse der Population des Multiagentensystems zeigt, daß der Globalagent dieses Benutzers einen sehr großen Etat hat und daher viele Artikel empfehlen kann. Am 13. 5. stammen von 99 Empfehlungen nur fünf nicht vom Globalagenten. Die Ressort- und Themenagenten der Population empfahlen dagegen 27 Artikel. Von diesen wurden 17 gelesen, das entspricht einer Precision von 0,63. Die Precision des Pressespiegels der gesamten Agentenpopulation ist aus Abbildung 6.11b) abzulesen und beträgt an diesem Tag nur 0,21.

Die Gewichte der Zusammenfassungskomponente spiegeln ebenfalls die schlechte Leistung des Globalagenten: Während die Themenagenten ein Gewicht von 0,59 zugewiesen bekommen haben und die Ressortagenten ein Gewicht von 0,43 besitzen, werden die Empfehlungen des Globalagenten nur mit 0,25 gewichtet. Da die Zusammenfassungskomponente aus den Empfehlungen nur eine Reihenfolge berechnet und keine zusätzliche Filterung durchführt, gelangen die schlechten Empfehlungen des Globalagenten dennoch in den Pressespiegel.

6.2.4 Auswertung

Der Betrieb des *PZ*-Systems konnte die guten Ergebnisse der Tests mit fikiven Benutzern nicht auf voller Linie bestätigen. Während für Benutzer 1 ein Pressespiegel von konstanter Größe und Güte zusammengestellt wurde, erwiesen sich die heuristisch gewählten Parametereinstellungen für Benutzer 3 als ungeeignet. Besonders die Höhe der Belohnungen B^{+}_{FB} und B^{++}_{FB} scheint noch nicht optimal

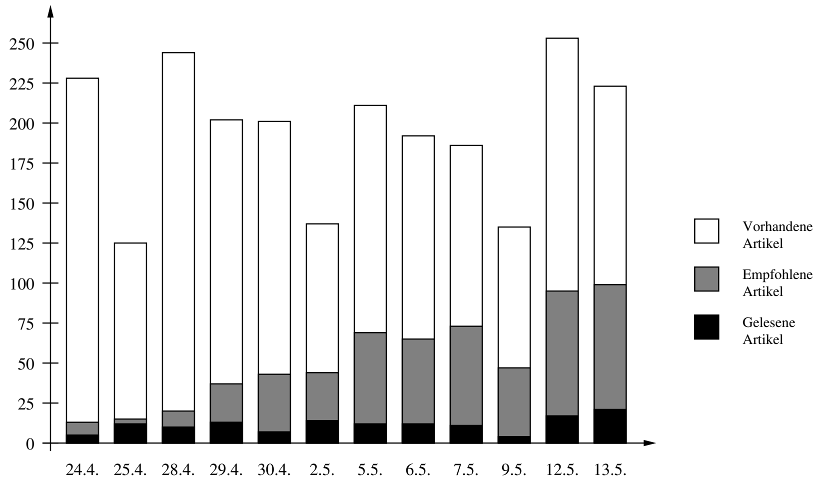


Abbildung 6.10: Anzahl vorhandener/empfohlener/gelesener Artikel pro Tag (Benutzer 3)

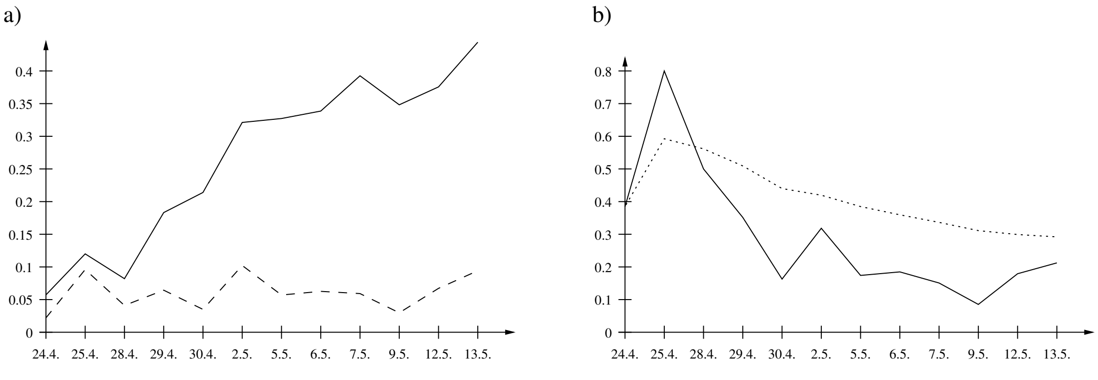


Abbildung 6.11: a) Verhältnis zwischen der Anzahl empfohlener bzw. gelesener Artikel und der Größe der Artikelkollektion b) Precision (Benutzer 3)

gewählt. Ein Globalagent mit sehr großem Etat empfiehlt viele Artikel, von denen immer eine Reihe als relevant bewertet werden und durch die dafür ausgeschüttete Belohnung den Etat weiter erhöhen. Der Benutzer muß in einer solchen Situation verstärkt explizites negatives Feedback für unerwünschte Artikel geben.

Kapitel 7

Zusammenfassung

Angesichts der steigenden Informationsflut, die in Papierform als Buch, Zeitschrift oder Zeitung und über elektronischen Kanäle wie Fernsehen oder Internet auf jedermann zuströmt, wächst die Notwendigkeit, Methoden und Werkzeuge zur Bewältigung des Datenstroms zu entwickeln.

Für die vorliegende Arbeit wurde ein Ausschnitt aus dem großen Informationsangebot des Internet als Anwendungsgebiet ausgewählt, der bisher kaum Beachtung gefunden hat. Mehr und mehr deutsche Tageszeitungen bieten ihre aktuelle Ausgabe in Ausschnitten oder vollständig im WWW an. Für die Internetanwender ergibt sich die Möglichkeit, zu einem Nachrichtenthema die Berichte verschiedener Zeitungen zu lesen und sich so ausgewogen und umfassend zu informieren. Die Suche nach den interessanten Artikeln erfordert jedoch einen sehr großen Zeitaufwand, der die vollständige Nutzung des Angebots verhindert.

Diese Arbeit untersucht, ob der Einsatz eines Computerprogramms den Benutzern Hilfe bieten kann. Dazu soll das Programm täglich einen auf das Interesse des Benutzers abgestimmten Pressespiegel erstellen. Dabei ist zu berücksichtigen, daß das Interesse nicht fest auf ein Thema fixiert ist, sondern sich auf mehrere u.U. sehr verschiedenartige Themen verteilt. Zudem verändern sich Interessen des Benutzers mit der Zeit.

Das im Rahmen dieser Arbeit entwickelte *PZ*-System kombiniert Verfahren aus dem Bereich der *Agentensysteme*, dem *Information Filtering* und dem *Maschinellen Lernen*, um diese Anforderungen zu erfüllen. Das *PZ*-System ist als *Interface-Agent* entworfen, der dem Benutzer die Informationssuche nicht vollständig abnimmt, sondern ihn dabei begleitet und unterstützt. Die Architektur des *PZ*-Systems ist ein ein *Multiagentensystem* mit einer Reihe von konkurrierenden Filteragenten. Die Agenten benutzen unterschiedliche Strategien, um Teilbereiche des Benutzerinteresses abzubilden, und lernen mittels des *Rocchio-Algorithmus* aus Beobachtungen des Leseverhaltens Benutzerprofile. Die Koordination der gesamten Agentenpopulation erfolgt durch die leistungsbezogene Vergabe von Belohnungen oder Bestrafungen für die Agenten. Leistungsfähige Agenten bleiben im System, schwache hingegen werden entfernt.

Die Akzeptanz eines Informationsfilters bei Anwendern hängt von zwei Faktoren ab:

1. Die Benutzung des Filters muß gegenüber dem direkten Zugang zu den Informationsquellen die Suche nach Information deutlich erleichtern und verbessern.

2. Die Anpassung des Filters an die individuellen Informationsbedürfnisse der Benutzer darf keinen großen Aufwand für den Benutzer bedeuten.

In Tests mit fiktiven Benutzern stellte das *PZ*-System seine Leistungsfähigkeit unter Beweis. Nach einer kurzen Lernphase zu Beginn der Benutzung waren die Agenten in der Lage, einen Pressespiegel mit mehr als der Hälfte der relevanten Artikel zusammenzustellen. Veränderungen des Interesses, die in den Tests sehr abrupt vorgenommen wurden, wurden ebenfalls sehr gut verfolgt.

Der Systembetrieb zeigte, daß der Benutzer dem System anfangs einige Hinweise auf seine Interessen geben muß, bis die Agenten sich daran angepaßt haben. Die Filterleistung bleibt bei der weiteren Benutzung alleine durch das Lernen anhand der Beobachtungen konstant.

Der Systembetrieb deckte auch einen Schwachpunkt des momentanen Systems auf: das Verhalten des Systems ist abhängig von den Einstellungen einiger Parameter, die bisher heuristisch und noch nicht optimal gewählt wurden. Weitere Tests sind nötig, um diese Parameter optimal einzustellen. Um ein besseres Verständnis für die Lerndynamik des Multiagentensystems und den Einfluß einzelner Agentenarten auf den Filterprozeß zu erhalten, ist eine weitergehende Analyse des Verhaltens der einzelnen Agenten erforderlich.

Zur Ergänzung der Funktionalität und zur Verbesserung der Filterqualität sind einige Erweiterungen des aktuellen Systems denkbar. Das Auswählen und Empfehlen der Artikel geschieht momentan aufgrund einer Repräsentation des Inhalts der Artikel. Selektionsverfahren, die Attribute wie Artikellänge, Autor oder Zeitungsverlag berücksichtigen, wären eine Alternative.

Das *PZ*-System stellt schon jetzt Pressespiegel für mehrere Benutzer zusammen. Agenten könnten aber nicht nur für einen Benutzer filtern, sondern innerhalb eines Agentensystems für alle Benutzer Artikel für gemeinsame Interessen verschiedener Benutzer empfehlen. Die Anwendung von Methoden des kollaborativen Filterns ist ebenfalls möglich.

Die vorliegende Arbeit zeigt, daß auch auf deutschen Texten der Einsatz von Methoden des Information Filtering möglich ist. Durch die Ausnutzung von Charakteristika der Tageszeitungsdomäne wurde eine gute Filterleistung erreicht.

Das *PZ*-System ist nicht als Ersatz für eine normale Tageszeitung gedacht, obwohl der Name **P**ersönliche **Z**eitung dies nahelegt. Dazu läßt sich der Pressespiegel viel zu schlecht am Frühstückstisch lesen. Als Werkzeug zur Informationsuche zu bestimmten Themen im großen Angebot der Tageszeitungen jedoch hat das *PZ*-System durchaus seine Bedeutung.

Anhang A

Implementation

Die Implemenation des *PZ*-Systems besteht aus drei großen Teilen:

1. Indexierungskomponente
2. Filterkomponente
3. Anzeigekomponente

Da erstere und letzere Komponente sehr viele Ein- und Ausgabeoperationen für Zeichenketten benötigen, wurde zur Implementation Perl5 benützt, das auf einfache Art diese Funktionalität zur Verfügung stellt. Die laufzeitintensivere Filterkomponente wurde in C++ implementiert.

Die Komponenten werden auf unterschiedliche Weise aufgerufen. Die Anzeigekomponente muß auf Anforderung des Benutzers HTML-Seiten generieren und über einen HTTP-Server an den Benutzer übermitteln. Die einzelnen Teilprogramme der Anzeige werden also vom Benutzer aufgerufen. Die Indexierungs- und die Filterkomponente werden jede Nacht automatisch aufgerufen und erledigen ihre Aufgaben. Zur Vereinfachung des Aufrufs existiert ein Perl5-Skript `steuerPZ2.pl`, das nicht nur die Programme in der richtigen Reihenfolge startet, sondern zusätzlich die erforderlichen Verzeichnisse anlegt und mit `webcopy` die HTML-Seiten der Zeitungen für einen Tag aus dem WWW auf die lokale Festplatte lädt.

Der Datenaustausch zwischen den Komponenten erfolgt über Dateien, die im *PZ*-Verzeichnisbaum abgelegt werden. Der Verzeichnisbaum teilt sich in benutzerunspezifische (Verzeichnis `Artikel/`) und benutzerspezifische (Verzeichnis `Benutzer/`) Daten:

```
<PZ-Verzeichnis>/
<PZ-Verzeichnis>/Artikel/
<PZ-Verzeichnis>/Artikel/<Tag>/
<PZ-Verzeichnis>/Artikel/<Tag>/<Artikelnummer>.ind
```

```
<PZ-Verzeichnis>/Artikel/<Tag>/<Artikelnummer>.html
<PZ-Verzeichnis>/Artikel/<Tag>/gesamt.ind
<PZ-Verzeichnis>/Artikel/<Tag>/artikel.nlist
<PZ-Verzeichnis>/Artikel/<Tag>/clustering

<PZ-Verzeichnis>/Benutzer/
<PZ-Verzeichnis>/Benutzer/benutzer
<PZ-Verzeichnis>/Benutzer/<Benutzername>/
<PZ-Verzeichnis>/Benutzer/<Benutzername>/<Tag>.dat
<PZ-Verzeichnis>/Benutzer/<Benutzername>/<Tag>.nlist
<PZ-Verzeichnis>/Benutzer/<Benutzername>/<Tag>.nfb
```

Für jeden Tag ist ein Unterverzeichnis im Verzeichnisbaum `Artikel/` vorhanden. Dort sind die Artikelindices in den Dateien `<Artikelnummer>.ind` und die vorformatierten HTML-Seiten in den Dateien `<Artikelnummer>.html` gespeichert. `gesamt.ind` enthält den Kollektionsindex, während `clustering` die Clusterbeschreibungen eines Tages enthält. In `artikel.nlist` schließlich ist die Artikelliste abgelegt.

Die Datei `benutzer` enthält die Namen und Paßwörter aller im *PZ*- System angemeldeten Benutzer. Für jeden Benutzer existiert ein eigenes Unterverzeichnis. Dort ist zu jedem Tag der Zustand der Agentenpopulation des Benutzers in `<Tag>.dat` abgespeichert. Die Datei `<Tag>.nlist` enthält den Pressespiegel für diesen Tag und die Datei `<Tag>.nfb` das Feedback des Benutzers für diesen Pressespiegel.

A.1 Indexierungskomponente

Die Indexierungskomponente besteht aus fünf Programmen:

`html2gertwol.pl` liest die HTML-Seiten von der Festplatte und extrahiert die Artikeldaten daraus. Diese werden (bis auf den Artikeltext) in Form einer Artikelliste gesichert. Für jeden Artikel wird außerdem eine vorformatierte HTML-Seite erzeugt und abgespeichert. Eine temporäre Datei `zeitung.art` enthält die Artikeldaten als Eingabe für GERTWOL. `htmt2gertwol.pl` benutzt zur Extraktion Subroutinen aus `extract.pl`.

GERTWOL führt die morphologische Analyse durch.

`gertwol2index.pl` ist durch eine UNIX-Pipe mit der Ausgabe von GERTWOL gekoppelt. Hier erfolgt die Termgenerierung und die Dimensionsreduktion durch Wortartauswahl. Es entstehen vorläufige, mit der Termfrequenz gewichtete Artikelindices und der Kollektionsindex, die jeweils in die entsprechenden Dateien gespeichert werden. Die Indices sind lexikalisch aufsteigend sortiert.

`index2tfidf.pl` führt die Berechnung der *TFIDF*-Gewichtung der Artikelindices mit Hilfe des Kollektionsindices durch. Die Vektoren werden auf die Länge 1 normiert, um die Ähnlichkeitsberechnung zu vereinfachen.

`sortindex` sortiert die Artikelindices und den Kollektionsindex nach C++ Konvention. Dieser Schritt ist notwendig, da die Sortierreihenfolge von Perl5 Zeichenketten und des C++ Standarddatentyps `string` für nicht-ASCII-Zeichen (z.B. Umlaute und Buchstaben mit Akzenten) nicht übereinstimmt.

A.2 Filterkomponente

Die Filterkomponente realisiert nicht nur das Multiagentensystem sondern auch das Clustering. Das Filtern für alle Benutzer des Systems wird durch einen Aufruf des Programms `pzfilter` gestartet.

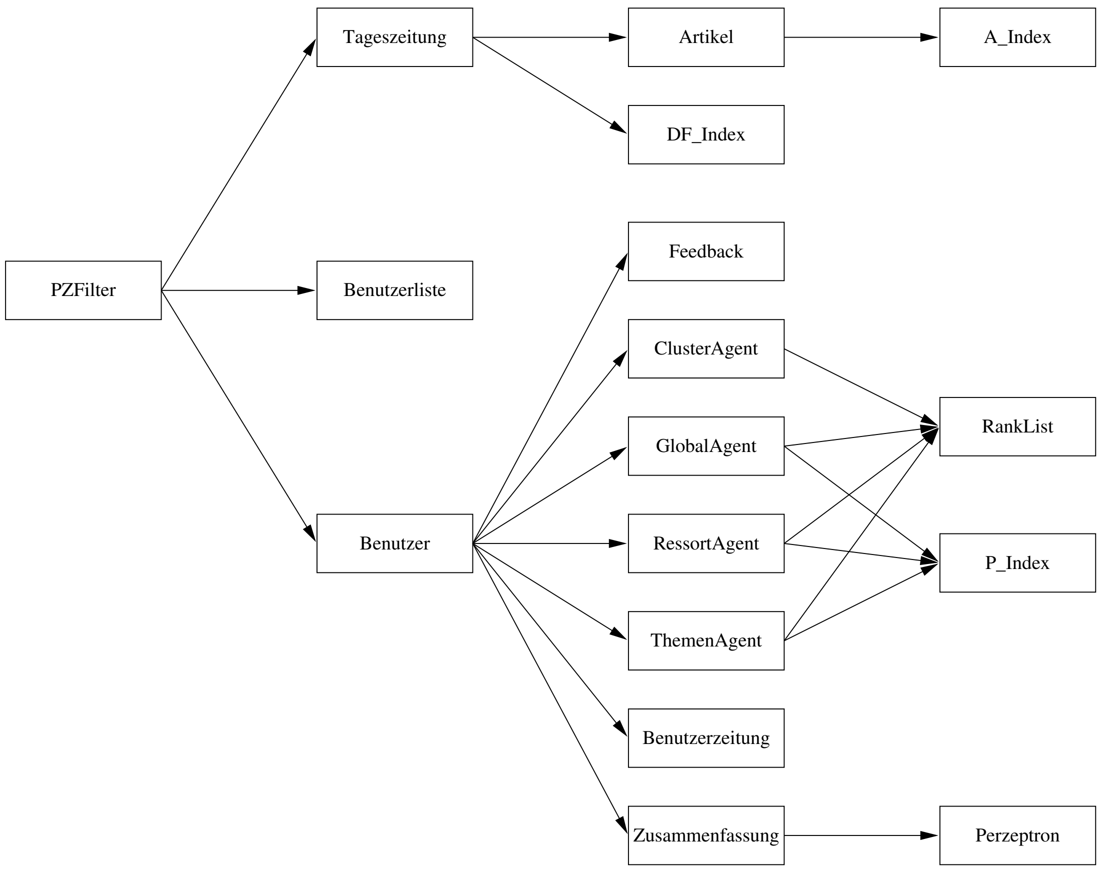


Abbildung A.1: Klassenstruktur des Programms `pzfilter`.

Die Klassenstruktur von `pzfilter` ist in Abbildung A.1 gezeigt. Im folgenden werden die einzelnen Klassen erläutert.

PZFilter ist das Hauptprogramm. Es initialisiert die Tageszeitung des aktuellen Datums und des vergangenen Tages und startet für jeden Benutzer von der Benutzerliste das Lernen und Filtern.

Tageszeitung stellt benutzerunabhängig sämtliche Artikeldaten, den Kollektionsindex und die Clusterbeschreibungen zur Verfügung. Das Clustering wird von Tageszeitung selbst berechnet und in der Datei `clustering` abgespeichert.

Artikel kapselt die Artikeldaten Zeitung, Ressort, Schlagzeile, Unterzeile und Kurzzusammenfassung und enthält den Artikelindex.

A_Index speichert einen Artikelindex und bietet Funktionen zum Addieren, Normalisieren und Multiplizieren von Artikelindices.

DF_Index speichert einen Kollektionsindex.

Benutzerliste enthält die Namen aller Benutzer.

Benutzer bildet das Multiagentensystem für einen Benutzer ab. Benutzer stellt das Blackboard über die Klassen Feedback und Zusammenfassung zur Verfügung und kontrolliert das Lernen und Filtern der Agenten.

ClusterAgent, GlobalAgent, RessortAgent und ThemenAgent erstellen jeweils Empfehlungen für Artikel und speichern diese in der Benutzerzeitung ab.

P_Index speichert den Profilindex für Global-, Ressort- und Themenagenten.

RankList dient der lokalen Speicherung eines Rankings für die Agenten und der endgültigen Artikelreihenfolge für einen Benutzer.

Benutzerzeitung sammelt die Empfehlungen aller Agenten.

Zusammenfassung kombiniert die in Benutzerzeitung gespeicherten Empfehlungen mit Hilfe eines Perzeptrons zu einem Ranking.

Perzeptron ist eine allgemeine Implementation eines Perzeptrons.

A.3 Anzeigekomponente

Die Anzeigekomponente besteht aus zwei Teilen: der Anzeige des Pressespiegels und der Anzeige des Kiosk. Beide sind ähnlich aufgebaut. Abbildung A.2 zeigt für den Pressespiegel die Aufteilung des Browserfensters in einzelne Frames und die dazugehörigen CGI-Skripte, die HTML-Code für die Frames produzieren. Auf der

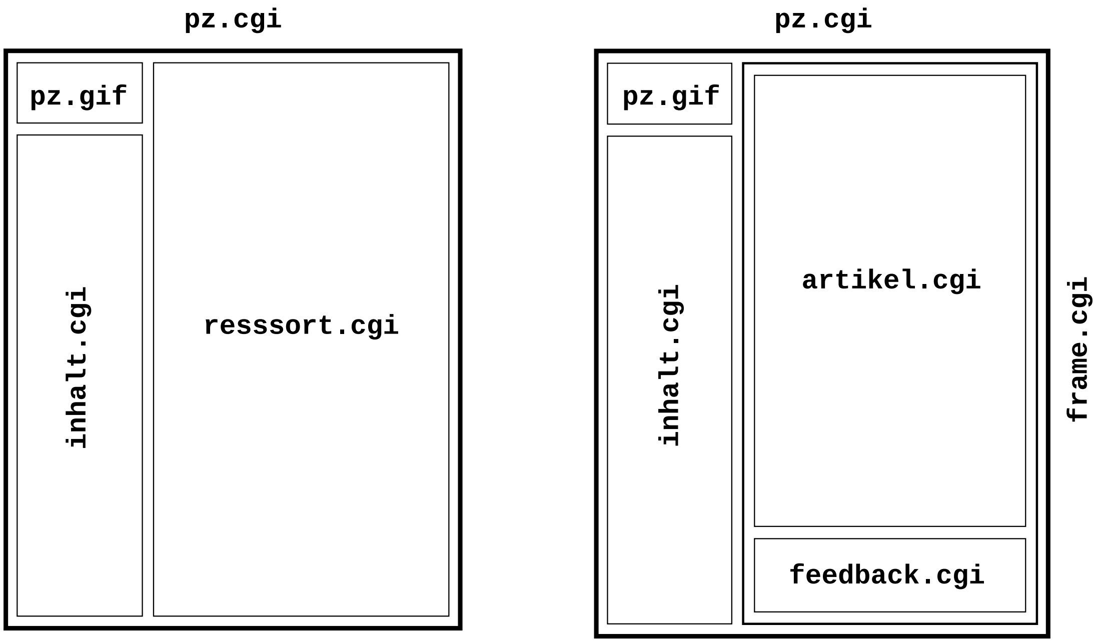


Abbildung A.2: Aufteilung des Browserfensters in Frames.

linken Seite ist die Situation bei der Anzeige von Ressortüberblicken gezeigt, auf der linken Seite sieht man die Aufteilung für die Anzeige eines einzelnen Artikels.

Die CGI-Skripte lesen die benötigten Daten (Benutzernamen und Paßwörter, Benutzerpressespiegel, Clusterbeschreibungen und vorformatierte HTML-Seiten) aus den entsprechenden Dateien im *PZ*-Verzeichnisbaum. Einige Skripte schreiben in die Feedbackdatei, das Hauptskript auch in die Namen- und Paßwortdatei. Im folgenden werden die einzelnen Skripte beschrieben.

`pz.cgi` ist das Hauptskript. Ist kein Cookie für das *PZ*-System auf dem Rechner des Benutzers vorhanden, wird die Eingangsseite ausgegeben. Gibt der Benutzer Name und Paßwort ein, werden die Daten wieder an `pz.cgi` übertragen. Stellt das Skript fest, daß ein neuer Benutzer sich anmeldet, wird ein Standardpressespiegel für den Benutzer generiert und eine Anfangsagentenpopulation erzeugt. Dann wird ein Cookie auf dem Rechner des Benutzers gesetzt und das Skript ruft sich selber erneut auf. Nun ist ein Cookie gesetzt und das äußere Frameset (Abbildung A.2 links) kann ausgegeben werden.

`inhalt.cgi` zeigt eine Liste der Namen der im Pressespiegel verfügbaren Ressorts an. Außerdem werden Verweise auf den Cartoon der taz, eine Seite mit Informationen zum *PZ*-System und den Kiosk erzeugt.

`ressort.cgi` gibt eine Liste der Kopf- und Schlagzeilen, Untertiteln und Kurzzusammenfassungen aller Artikel eines Ressorts aus.

`frame.cgi` erzeugt das innere Frameset, das in Abbildung A.2 rechts abgebildet ist.

`artikel.cgi` zeigt den ganzen Text eines Artikels mit Hilfe der vorformatierten HTML-Seite zum Artikel und einen Überblick über das Ressort an. Beim Aufruf von `artikel.cgi` wird ein Eintrag für den Artikel in die Feedbackdatei geschrieben.

`feedback.cgi` erzeugt die beiden Links für positives und negatives explizites Feedback. Falls einer der Links betätigt wird, wird dies in der Feedbackdatei entsprechend vermerkt.

Literaturverzeichnis

[Altavista, 1997] Altavista (1997).
URL: http://altavista.digital.com

[Angelaccio et al., 1996] Angelaccio, M., Zamburru, L. und Genovese, D. (1996). BOTH: Cooperative Automatic Web Navigation and Hierarchical Filtering. In *Proceedings Second Australian WorldWideWeb Conference (AUSWEB96)*.
URL: http://elmo.scu.edu.au/sponsored/ausweb/ausweb96/tech/angelaccio/paper.html

[Apté et al., 1994] Apté, C., Damerau, F. und Weiss, S. M. (1994). Automated Learning of Decision Rules for Text Categorization. *ACM Transactions on Information Systems*, 12(3):233–251.

[Armstrong et al., 1995] Armstrong, R., Freitag, D., Joachims, T. und Mitchell, T. (1995). WebWatcher: A Learning Apprentice for the World Wide Web. In *Proceedings AAAI Spring Symposium on Information Gathering from Distributed, Heterogeneous Environments*, Stanford, CA.
URL: http://www.isi.edu/sims/knoblock/sss95/papers/mitchell.ps

[Baclace, 1992] Baclace, P. E. (1992). Competitive Agents for Information Filtering. *Communications of the ACM*, 35(12):50.

[Balabanovic und Shoham, 1995] Balabanovic, M. und Shoham, Y. (1995). Learning Information Retrieval Agents: Experiments with Automated Web Browsing. In *Proceedings AAAI Spring Symposium on Information Gathering from Distributed, Heterogeneous Environments*, Stanford, CA.
URL: http://www.isi.edu/sims/knoblock/sss95/papers/balabanovic.ps

[Bates, 1994] Bates, J. (1994). The role of emotion in believable agents. *Communications of the ACM*, 37(7):122–125.

[Beale und Wood, 1994] Beale, R. und Wood, A. (1994). Agent-Based Interaction. In *Proceedings of HCI'94*, Seiten 239–245, Glasgow, UK.
URL: ftp://ftp.cs.bham.ac.uk/pub/dist/hci/papers/agent-based_interaction.ps.Z

[Belkin und Croft, 1992] Belkin, N. J. und Croft, W. (1992). Information Filtering and Information Retrieval: Two Sides of the Same Coin? *Communications of the ACM*, 35(12):29–38.

[Biberman, 1995] Biberman, Y. (1995). The Role of Prototypicality in Exemplar-Based Learning. In *Proceedings of 8th European Conference on Machine Learning (ECML95)*, Seiten 77–91.

[Boyan et al., 1996] Boyan, J., Freitag, D. und Joachims, T. (1996). A machine learning architecture for optimizing Web search engines. In *Proceedings AAAI-96 Workshop on Internet-based Information System*, Portland, OR.
URL: `http://www.cs.cmu.edu/~reinf/papers/boyan-laser.ps`

[Brachman und Schmolz, 1985] Brachman, R. und Schmolz, J. (1985). An Overview of the KL-ONE Knowledge Representation System. *Cognitive Science*, 9(2):171–216.

[Brewer und Johnson, 1994] Brewer, R. S. und Johnson, P. M. (1994). Toward Collaborative Knowledge Management within Large, Dynamically Structured Information Systems. Technical Report ICS-TR-94-02, University of Hawaii, Honolulu, HA.
URL: `ftp://ftp.ics.hawaii.edu/pub/tr/ics-tr-94-02.ps.Z`

[Brooks, 1986] Brooks, R. A. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2(1):14–23.

[Chimera und Shneiderman, 1994] Chimera, R. und Shneiderman, B. (1994). An Exploratory Evaluation of Three Interfaces for Browsing Large Hierarchical Tables of Contents. *ACM Transactions on Information Systems*, 12(4):383–406.
URL: `ftp://ftp.cs.umd.edu/pub/papers/papers/2620/2620.ps.Z`

[Clark und Niblett, 1989] Clark, P. und Niblett, T. (1989). The CN2 induction algorithm. *Machine Learning*, 3:261–283.

[Cohen, 1996] Cohen, W. W. (1996). Learning Rules that Classify E-Mail. In *Proceedings AAAI Spring Symposium on Machine Learning in Information Access*, Stanford, CA.
URL: `http://www.parc.xerox.com/istl/projects/mlia/papers/cohen.ps`

[Corkill, 1991] Corkill, D. D. (1991). Blackboard Systems. *AI Expert*, 6(6):40–47.

[Cutting et al., 1992] Cutting, D. R., Karger, D. R., Pedersen, J. O. und Tukey, J. W. (1992). Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections. In *Proceedings International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '92)*.

[Cypher, 1993] Cypher, A. (1993). *Watch what I do: Programming by Demonstration.* MIT Press, Cambridge, MA.

[de Kroon et al., 1996] de Kroon, H. C. M., Mitchell, T. und Kerckhoffs, E. J. H. (1996). Improving Learning Accuracy in Information Filtering. In *Proceedings International Conference on Machine Learning - Workshop Machine Learning Meets HCI (ICML-96)*, Bari, I.
URL: `http://www.cs.cmu.edu/afs/cs/project/theo-4/text-learning/www/publications/ICML96.ps`

[Deerwester et al., 1990] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K. und Harshman, R. (1990). Indexing by Latent Semantic Indexing. *Journal of the American Society for Information Science*, 42(6):391–407.

[Edwards et al., 1996] Edwards, P., Bayer, D., Green, C. L. und Payne, T. R. (1996). Experience with Learning Agents which Manage Internet-Based Information. In *Proceedings AAAI Spring Symposium on Machine Learning in Information Access*, Stanford, CA.
URL: `http://www.parc.xerox.com/istl/projects/mlia/papers/edwards.ps`

[Etzioni und Weld, 1994] Etzioni, O. und Weld, D. (1994). A Softbot-Based Interface to the Internet. *Communications of the ACM*, 37(7):72–76.
URL: `ftp://ftp.cs.washington.edu/pub/etzioni/softbots/cacm.ps.Z`

[Etzioni und Weld, 1995] Etzioni, O. und Weld, D. S. (1995). Intelligent Agents on the Internet - Fact, Fiction, and Forecast. *IEEE Expert - Intelligent Internet Services*, 10(4):44–49.
URL: `ftp://ftp.cs.washington.edu/pub/ai/ieee-expert.ps.Z`

[Ferguson, 1992] Ferguson, I. A. (1992). *TouringMachines: An Architecture for Dynamic, Rational, Mobile Agents.* PhD thesis, Clare Hall, University of Cambridge, UK.
URL: `http://coli.lili.uni-bielefeld.de/~milde/seminar.s95/papers/TR273-iaf-thesis.ps.gz`

[Ferguson und Karakoulas, 1996] Ferguson, I. A. und Karakoulas, G. J. (1996). Multiagent Learning and Adaptation in an Information Filtering Market. In *Proceedings AAAI Spring Symposium on Adaptation, Coevolution and Learning in Multiagent Systems*, Stanford, CA.

[Firefly Network Inc., 1997] Firefly Network Inc. (1997). Firefly.
URL: `http://www.firefly.com`

[Fischer und Stevens, 1991] Fischer, G. und Stevens, C. (1991). Information Access in Complex, Poorly Structured Information Spaces. In *Proceedings of Conference on Human Factors in Computing Systems CHI'91*, Seiten 63–70, New Orleans, LA.

[Foloutsos, 1992] Foloutsos, C. (1992). Signature Files. In Frakes, W. B. und Baeza-Yates, R., Herausgeber, *Information Retrieval - Data Structures & Algorithms*, Kapitel: 4, Seiten 44–65. Prentice Hall, Engelwood Cliffs, NJ.

[Foltz und Dumais, 1992] Foltz, P. W. und Dumais, S. T. (1992). Personalized Information Delivery: An Analysis of Information Filtering Methods. *Communications of the ACM*, 35(12):51–60.

[Foner, 1993] Foner, L. N. (1993). What's and Agent, Anyway? A Sociological Case Study. Agents Memo 93-01, MIT Media Lab, Cambridge, MA.
URL: `ftp://ftp.media.mit.edu/Foner/Papers/Julia/Agents--Julia.ps`

[Foner, 1996] Foner, L. N. (1996). A Multi-Agent Referral System for Matchmaking. In *Proceedings of the First International Conference on the Practical Applications of Intelligent Agents and Multi-Agent Technology PAAM96*, London, UK.
URL: `http://foner.www.media.mit.edu/people/foner/Reports/PAAM-96/PAAM.ps`

[Franklin und Graeser, 1996] Franklin, S. und Graeser, A. (1996). Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents. In *Proceedings of the Third International Workshop on Agent Theories, Architectures and Languages*. Springer-Verlag.
URL: `http://www.msci.memphis.edu/~franklin/AgentProg.html`

[Fuhr, 1997] Fuhr, N. (1997). Skriptum zur Vorlesung Information Retrieval Sommersemester 97.
URL: `http://ls6-www.informatik.uni-dortmund.de/ir/doc/courses/ir/irskall.ps.gz`

[Genesereth und Ketchpel, 1994] Genesereth, M. R. und Ketchpel, S. P. (1994). Software Agents. *Communications of the ACM*, 37(7):48–53.

[Goldberg et al., 1992] Goldberg, D., Nicholas, D., Oki, B. M. und Terry, D. (1992). Using Collaborative Filtering to Weave an Information Tapestry. *Communications of the ACM*, 35(12):61–70.

[Haapalainen und Majorin, 1994] Haapalainen, M. und Majorin, A. (1994). GERTWOL: Ein System zur automatischen Wortformerkennung deutscher Wörter.

[Harmann et al., 1992] Harmann, D., Baeza-Yates, R., Fox, E. und Lee, W. (1992). Inverted Files. In Frakes, W. B. und Baeza-Yates, R., Herausgeber, *Information Retrieval - Data Structures & Algorithms*, Kapitel: 3, Seiten 28–43. Prentice Hall, Engelwood Cliffs, NJ.

[Hermans, 1996] Hermans, B. (1996). *Intelligent Software Agents on the Internet: an inventory of currently offered functionality in the information society & a prediction of (near-)future developments.* PhD thesis, Universiteit Tilburg, Tilburg, NL.
URL: `http://www.hermans.org/agents`

[Holland, 1975] Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI.

[HotBot, 1997] HotBot (1997).
URL: `http://www.hotbot.com`

[Hull, 1995] Hull, D. A. (1995). *Information Retrieval using statistical Classification*. PhD thesis, Stanford University.

[Jennings und Higuchi, 1992] Jennings, A. und Higuchi, H. (1992). A Personal News Service based on a User Model Neural Network. *IEICE Transactions on Information and Systems*, 75(2):198–209.

[Joachims, 1996a] Joachims, T. (1996). Einsatz eines intelligenten, lernenden Agenten für das World Wide Web. Diplomarbeit, Universität Dortmund.
URL: `ftp://ftp-ai.informatik.uni-dortmund.de/pub/Diplomarbeiten/joachims.ps.Z`

[Joachims, 1996b] Joachims, T. (1996). A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. Technical Report CMU-CS-96-118, Carnegie Mellon University, Pittsburgh, PA.
URL: `ftp://reports.adm.cs.cmu.edu/usr/anon/1996/CMU-CS-96-118.ps`

[Kaelbling et al., 1996] Kaelbling, L. P., Littmann, M. L. und Moore, A. W. (1996). Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research*, 4:237–285.
URL: `http://www.cs.brown.edu/people/lpk/rl-survey.ps`

[Karakoulas und Ferguson, 1995] Karakoulas, G. J. und Ferguson, I. A. (1995). A Computational Market for Information Filtering in Multi-Dimensional Spaces. In *Proceedings AAAI Fall Symposium on AI Applications in Knowledge Navigation and Retrieval*, Seiten 78–83, Cambridge, MA.

[Konstan et al., 1997] Konstan, J. A., Miller, B. N., Maltz, D., Herlocker, J. L., Gordon, L. R. und Riedl, J. (1997). GroupLens: Applying Collaborative Filtering to Usenet News. *Communications of the ACM*, 40(3):77–87.

[Lang, 1995] Lang, K. (1995). NewsWeeder: Learning to Filter Netnews. In *Proceedings of the 12th International Machine Learning Conference (ICML95)*, Seiten 331–339.

[Lewis, 1992] Lewis, D. D. (1992). An Evaluation of Phrasal and Clustered Representations on a Text Categorization Task. In *Proceedings International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '92)*, Seiten 37–50.
URL: `http://www.research.att.com/~lewis/papers/lewis92b.ps`

[Liebermann, 1995] Liebermann, H. (1995). Letizia: An Agent That Assists Web Browsing. In *Proceedings International Joint Conference on Artificial Intelligence (IJCAI-95)*, Montreal, Kanada.
URL: http://lieber.www.media.mit.edu/people/lieber/Lieberary/Letizia/Letizia-AAAI/Letizia.ps

[Loeb, 1992] Loeb, S. (1992). Architecting Personalized Delivery of Multimedia Information. *Communications of the ACM*, 35(12):39–48.

[Luhn, 1958] Luhn, H. P. (1958). A Business Intelligence System. *IBM Journal of Research and Development*, 2(4):314–319.

[Lycos, 1997] Lycos (1997).
URL: http://www.lycos.com

[Mackensen, 1986] Mackensen, L. (1986). *Deutsches Wörterbuch*. Südwest Verlag, 11. Auflage.

[Maes, 1991] Maes, P. (1991). The agent network architecture. *SIGART Bulletin*, 2(4):115–120.

[Maes, 1994a] Maes, P. (1994). Agents that Reduce Work and Information Overload. *Communications of the ACM*, 37(7):31–40.
URL: http://pattie.www.media.mit.edu/people/pattie/CACM-94/CACM-94.p1.html

[Maes, 1994b] Maes, P. (1994). Modeling Adaptive Autonomous Agents. *Artificial Life Journal*, 1(1 & 2).
URL: http://pattie.www.media.mit.edu/people/pattie/alife-journal.ps

[Maes und Kozierok, 1993] Maes, P. und Kozierok, R. (1993). Learning Interface Agents. In *Proceedings of the Eleventh National Conference on Artificial Intelligence AAAI-93*, Washington, DC.

[Malone et al., 1992] Malone, T. W., Grant, K. R., Turbak, F. A. und Brobst, S. A. (1992). Intelligent Information Sharing Systems. *Communications of the ACM*, 30(5):390–402.

[Maltz, 1994] Maltz, D. A. (1994). Distributing Information for Collaborative Filtering on Usenet Net News. Master's thesis, MIT Department of Electrical Engineering and Computer Science, Cambridge, MA.
URL: http://www.cs.cmu.edu/afs/cs.cmu.edu/user/maltz/Doc/mit-thesis.ps.gz

[Marchionini, 1995] Marchionini, G. (1995). *Information Seeking in Electronic Environments*. Cambridge University Press, Cambridge, MA.

[Miller, 1995] Miller, G. A. (1995). WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41.

[Minsky, 1975] Minsky, M. (1975). A Framework for Representing Knowledge. In Winston, P. H., Herausgeber, *The Psychology of Computer Vision*, Seiten 211–227. McGraw-Hill, New York, NY.

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

[Mitchell et al., 1994] Mitchell, T., Caruana, R., Freitag, D., McDermott, J. und Zabowski, D. (1994). Experience with a Learning Personal Assistant. *Communications of the ACM*, 37(7):81–91.

[Mladenic, 1996] Mladenic, D. (1996). Personal WebWatcher: Implementation and Design. Technical Report IJS-DP-7472, Carnegie Mellon University, Pittsburgh, PA.
URL: `http://www.cs.cmu.edu/afs/cs/project/theo-4/text-learning/www/publications/pwwTR.ps`

[Mock, 1996] Mock, K. J. (1996). *Intelligent Information Filtering via Hybrid Techniques: Hill Climbing, Case-Based Reasoning, Index Patterns, and Genetic Algorithms*. PhD thesis, University of California, Davis, CA.
URL: `http://phobos.cs.ucdavis.edu:8001/~mock/infos/aaai96.ps.gz`

[Mock und Vemuri, 1994] Mock, K. J. und Vemuri, V. R. (1994). Adaptive User Models for Intelligent Information Filtering. In *Proceedings of the Third Golden West International Conference on Intelligent Systems*, Las Vegas, NV.
URL: `http://phobos.cs.ucdavis.edu:8001/papers/gwic.ps.gz`

[Morik, 1993] Morik, K. (1993). Maschinelles Lernen. In Görz, G., Herausgeber, *Einführung in die Künstliche Intelligenz*, Kapitel: 3. Addison-Wesley, Bonn, D.

[Moukas, 1996] Moukas, A. (1996). Amalthea: Information Discovery and Filtering using a Multiagent Evolving Ecosystem. In *Proceedings First International Conference on Practical Application of Intelligent Agents and Multi-Agent Technology*, Seiten 421–436, London.
URL: `http://moux.www.media.mit.edu/people/moux/papers/PAAM96.ps`

[Moukas und Zacharia, 1997] Moukas, A. und Zacharia, G. (1997). Evolving a Multi-agent Information Filtering Solution in Amalthea. In *Proceedings First International Conference on Autonomous Agents*, Marina del Rey, CA.
URL: `ftp://ftp.media.mit.edu/pub/agents/moux/aa97.ps`

[MUC3, 1991] MUC3 (1991). *Proceedings of the Third Message Understanding Conference (MUC-3)*, San Mateo, CA. Morgan Kaufmann.

[MUC4, 1992] MUC4 (1992). *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, San Mateo, CA. Morgan Kaufmann.

[MUC5, 1993] MUC5 (1993). *Proceedings of the Fifth Message Understanding Conference (MUC-5)*, San Mateo, CA. Morgan Kaufmann.

[Mukhopadhyay et al., 1996] Mukhopadhyay, S., Mostafa, J., Olakal, M., Lam, W., Xue, L. und Hudli, A. (1996). An Adaptive Multi-Level Information Filtering System. In *Proceedings Fifth International Conference on User Modelling.*
URL: ftp://ftp.informatik.uni-essen.de/pub/UMUAI/um_96/ information_filtering_system.ps.gz

[Müller et al., 1995] Müller, J. P., Pishel, M. und Thiel, M. (1995). Modelling Reactive Behaviour in Vertically Layered Agent Architectures. In Wooldridge, M. und Jennings, N. R., Herausgeber, *Intelligent Agents*, Nummer 890 in Lecture Notes in Artificial Intelligence, Seiten 261–276. Springer Verlag, Heidelberg.

[Newell, 1980] Newell, A. (1980). Physical Symbol Systems. *Cognitive Science*, (4).

[Nwana, 1996] Nwana, H. S. (1996). Software Agents: An Overview. *Knowledge Engineering Review*, 11(3):205–244.
URL: http://www.cs.umbc.edu/agents/introduction/ao.ps

[Oard, 1996] Oard, D. W. (1996). *Adaptive Vector Space Text Filtering for Monolingual and Cross-Language Applications.* PhD thesis, University of Maryland, Central Park, MD.
URL: http://www.clis.umd.edu/dlrg/filter/papers/thesis.ps.gz

[Oard und Marchionini, 1996] Oard, D. W. und Marchionini, G. (1996). A Conceptual Framework for Text Filtering. Technical Report CS-TR-3643, University of Maryland, College Park, MD.
URL: http://www.clis.umd.edu/dlrg/filter/papers/filter.ps

[Pannu und Sycara, 1996] Pannu, A. S. und Sycara, K. (1996). A Learning Personal Agent for Text Filtering and Notification. In *Proceedings of International Conference of Knowledge Based Systems (KBCS96).*
URL: http://www.cs.cmu.edu/~softagents/papers/kbcs96.ps.gz

[Payne und Edwards, 1995a] Payne, T. R. und Edwards, P. (1995). Interface Agents that Learn: An Investigation of Learning Issues in a Mail Agent Interface. Technical Report AUCS/TR9508, University of Aberdeen, Aberdeen, UK.
URL: ftp://ftp.csd.abdn.ac.uk/pub/pedwards/aai95.ps

[Payne und Edwards, 1995b] Payne, T. R. und Edwards, P. (1995). Learning Mechanisms for Information Filtering Agents. Technical Report AUCS/TR9509, University of Aberdeen, Aberdeen, UK.
URL: ftp://ftp.csd.abdn.ac.uk/pub/reports/tr9509.ps

[Pazzani et al., 1996] Pazzani, M., Muramatsu, J. und Billsus, D. (1996). Syskill & Webert: Identifying Interesting Web Sites. In *Proceedings AAAI Spring Symposium on Machine Learning in Information Access*, Stanford, CA.
URL: http://www.parc.xerox.com/istl/projects/mlia/papers/pazzani.ps

[Petrie, 1996] Petrie, C. J. (1996). Agent-based Engineering, the Web, and Intelligence. *IEEE Expert*, 11(6).
URL: http://cdr.stanford.edu/NextLink/Expert.html

[Pollock, 1988] Pollock, S. (1988). A Rule-based Message Filtering System. *ACM Transactions on Office Information Systems*, 6(3):232–254.

[Quinlan, 1994] Quinlan, J. R., Herausgeber (1994). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA.

[Rao und Georgeff, 1995] Rao, A. S. und Georgeff, M. (1995). BDI Agents: From Theory to Practice. In *Proceedings of the First International Conference on Multi-Agent Systems ICMAS-95*, Seiten 312–319, San Francisco, CA.

[Resnick et al., 1994] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. und Riedl, J. (1994). GroupLens: An Open Architecture for Collaborative Filtering of NetNews. In *Proceedings of the Conference on Computer Supported Cooperative Work*, Seiten 175–186.

[Resnick und Varian, 1997] Resnick, P. und Varian, H. R. (1997). Recommender Systems. *Communications of the ACM*, 40(3):56–58.

[Riloff und Lehnert, 1994] Riloff, E. und Lehnert, W. (1994). Information Extraction as a Basis for High-Precision Text Classification. *ACM Transactions on Information Systems*, 12(3):296–333.
URL: http://www.cs.utah.edu/~riloff/psfiles/single-acm.ps

[Rocchio, 1971] Rocchio, J. J. (1971). Relevance Feedback in Information Retrieval. In Salton, G., Herausgeber, *The SMART Retrieval System: Experiments in Automatic Document Processing*, Seiten 313–323. Prentice Hall, Engelwood Cliffs, NJ.

[Salton, 1971] Salton, G. (1971). The SMART Retrieval System: Experiments in Automatic Document Processing. In *The SMART Retrieval System: Experiments in Automatic Document Processing*, Seiten 313–323. Prentice Hall, Engelwood Cliffs, NJ.

[Salton und Buckley, 1990] Salton, G. und Buckley, C. (1990). Improving Retrieval Performance by Relevance Feedback. *Journal of the American Society for Information Science*, 41(4):288–296.

[Schewe, 1997] Schewe, S. (1997). Automatische Kategorisierung von Volltexten unter Anwendung von NLP-Techniken. Diplomarbeit, Universität Dortmund.
URL: ftp://ftp-ai.informatik.uni-dortmund.de/pub/Diplomarbeiten/schewe.ps.Z

[Schütze et al., 1995] Schütze, H., Hull, D. A. und Pedersen, J. O. (1995). A Comparison of Classifiers and Document Representations for the Routing Problem. In *Proceedings International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '95)*, Seiten 229–237.

[Sen et al., 1994] Sen, S., Sekaran, M. und Hale, J. (1994). Learning to coordinate without sharing information. In *Proceedings AAAI*, Seiten 426–431, Seattle, WA.
URL: http://euler.mcs.utusla.edu/~hale/learn.ps

[Shardanand und Maes, 1995] Shardanand, U. und Maes, P. (1995). Social Information Filtering: Algorithms for Automating "Word of Mouth". In *Proceedings of Conference on Human Factors in Computing Systems CHI'95*, Seiten 210–217.

[Sheth, 1994] Sheth, B. D. (1994). A Learning Approach to Personalized Information Filtering. Master's thesis, MIT Department of Electrical Engineering and Computer Science, Cambridge, MA.
URL: ftp://ftp.media.mit.edu/pub/agents/interface-agent/news-filter.ps

[Song et al., 1996] Song, H., Franklin, S. und Negatu, A. (1996). SUMPY: A Fuzzy Software Agent. In *Proceedings of the ISCA 5th International Conference on Intelligent Systems*, Seiten 124–129.
URL: http://www.msci.memphis.edu/~songh/publications/sumpy.html

[Sorensen und McElligott, 1995] Sorensen, H. und McElligott, M. (1995). PSUN: A Profiling System for Usenet News (Extended Abstract). In *Proceedings CKIM '95 Workshop on Intelligent Information Agents*, Baltimore, MD.
URL: http://odyssey.uuc.ie/filtering/psun.ps

[Stone und Veloso, 1996] Stone, P. und Veloso, M. (1996). Multiagent Systems: A Survey from a Machine Learning Perspective. *IEEE Transactions on Knowledge and Data Engineering*.
URL: http://www.cs.cmu.edu/afs/cs/usr/pstone/public/papers/96ieee-survey/survey.ps.Z

[Taylor, 1962] Taylor, R. S. (1962). The Process of Asking Questions. *American Documentation*, 13(4):391–336.

[Thomas und Fischer, 1996] Thomas, C. G. und Fischer, G. (1996). Using Agents to Improve the Usability and Usefulness of the World-Wide Web. In *Proceedings Fifth International Conference on User Modelling*, Hawaii, USA.
URL: ftp://ftp.informatik.uni-essen.de/pub/UMUAI/um_96/www_agents.ps.gz

[Web.de, 1997] Web.de (1997).
URL: `http://www.web.de`

[Wooldridge und Jennings, 1995] Wooldridge, M. und Jennings, N. R. (1995). Intelligent Agents: Theory and Practice. *Knowledge Engineering Review*, 10(2):115–152.
URL: `http://www.doc.mmu.ac.uk/STAFF/mike/ker95.ps`

[Yan und Garcia-Molina, 1994] Yan, T. W. und Garcia-Molina, H. (1994). Index Structures for Information Filtering Under the Vector Space Model. In *Proceedings International Conference on Data Engineering*, Seiten 337–347.
URL: `ftp://db.stanford.edu/pub/yan/1993/sdi-vector-model.ps`

[Yan und Garcia-Molina, 1995] Yan, T. W. und Garcia-Molina, H. (1995). SIFT - A Tool for Wide-Area Information Dissemination. In *Proceedings of the 1995 USENIX Technical Conference*, Seiten 177–186.
URL: `ftp://db.stanford.edu/pub/yan/1994/sift.ps`

[Yang und Pedersen, 1997] Yang, Y. und Pedersen, J. O. (1997). A Comparative Study on Feature Selection in Text Categorization. In *Proceedings of the 14th International Machine Learning Conference (ICML97)*. To be published.