Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

Document
D-94-07

Eine Übersicht über Information Retrieval (IR) und NLP-Verfahren zur Klassifikation von Texten

Claudia Wenzel, Rainer Hoch

Juni 1994

Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

Postfach 20 80
67608 Kaiserslautern
Tel.: (+49 631) 205-3211/13
Fax: (+49 631) 205-3210

Stuhlsatzenhausweg 3
66123 Saarbrücken
Tel.: (+49 681) 302-5252
Fax: (+49 681) 302-5341

Deutsches Forschungszentrum für Künstliche Intelligenz

The German Research Center for Artificial Intelligence (Deutsches Forschungszentrum für Künstliche Intelligenz, DFKI) with sites in Kaiserslautern and Saarbrücken is a non-profit organization which was founded in 1988. The shareholder companies are Atlas Elektronik, Daimler-Benz, Fraunhofer Gesellschaft, GMD, IBM, Insiders, Mannesmann-Kienzle, SEMA Group, and Siemens. Research projects conducted at the DFKI are funded by the German Ministry for Research and Technology, by the shareholder companies, or by other industrial contracts.

The DFKI conducts application-oriented basic research in the field of artificial intelligence and other related subfields of computer science. The overall goal is to construct *systems with technical knowledge and common sense* which - by using AI methods - implement a problem solution for a selected application area. Currently, there are the following research areas at the DFKI:

- Intelligent Engineering Systems
- Intelligent User Interfaces
- Computer Linguistics
- Programming Systems
- Deduction and Multiagent Systems
- Document Analysis and Office Automation.

The DFKI strives at making its research results available to the scientific community. There exist many contacts to domestic and foreign research institutions, both in academy and industry. The DFKI hosts technology transfer workshops for shareholders and other interested groups in order to inform about the current state of research.

From its beginning, the DFKI has provided an attractive working environment for AI researchers from Germany and from all over the world. The goal is to have a staff of about 100 researchers at the end of the building-up phase.

Dr. Dr. D. Ruland
Director

Eine Übersicht über Information Retrieval (IR) und NLP-Verfahren zur Klassifikation von Texten

Claudia Wenzel, Rainer Hoch

DFKI-D-94-07

This work has been supported by a grant from The Federal Ministry for Research and Technology (FKZ ITW-9401).

ISSN 0946-0098

Eine Übersicht über Information Retrieval (IR) und NLP-Verfahren zur Klassifikation von Texten

Claudia Wenzel, Rainer Hoch

German Research Center for Artificial Intelligence (DFKI)
P.O. Box 20 80, D - 67608 Kaiserslautern, Germany
Phone: (++49) 631-205-3584, Fax: (++49) 631-205-3210
{wenzel, hoch}@dfki.uni-kl.de

ABSTRACT: Die vorliegende Arbeit soll einen kurzen Überblick über gängige Ansätze aus dem Information Retrieval (IR) und der Natürlichsprachlichen Verarbeitung (NLP) zur Informationsextraktion geben. Diese Untersuchung wurde primär mit dem Ziel durchgeführt, statistische und wissensbasierte Techniken auf ihre Einsetzbarkeit zur Klassifikation von Texten zu evaluieren. Wir unterscheiden zwischen statistischen, regelbasierten, konzeptbasierten, probabilistischen sowie konnektionistischen Verfahren und stellen exemplarisch hierfür bekannte Systeme vor.

Sowohl Information Retrieval- als auch NLP-Systeme gehen von korrekten ASCII-Texten als Eingabe aus. Diese Voraussetzung gilt jedoch in der Dokumentanalyse nicht. Nach dem optischen Abtasten eines Dokuments, der Strukturanalyse und der nachfolgenden Texterkennung treten Wortalternativen mit Erkennungswahrscheinlichkeiten auf, die bei der partiellen inhaltlichen Analyse, d. h. der Informationsextraktion aus Texten, berücksichtigt werden müssen. Deshalb gehen wir am Schluß der Arbeit darauf ein, inwieweit die oben genannten Verfahren prinzipiell auf die Dokumentanalyse übertragbar sind.

Vorab soll betont werden, daß die vorliegende Studie zwei im Rahmen des ALV-Projektes am DFKI entwickelte Prototypen zur inhaltsbasierten Klassifikation von Dokumenten motiviert: einer verwendet statistische Methoden zur automatischen Indexierung; der andere beruht auf einem Regelinterpreter, der die bewerteten Worthypothesen als Evidenzen für Konzepte durch ein hierarchisches Netzwerk propagiert.

KEYWORDS: Information Retrieval, Natürlichsprachliche Verarbeitung, Dokumentanalyse, Textklassifikation, Robustheit

Inhaltsverzeichnis

1 Einleitung

Die Dokumentanalyse befaßt sich mit der elektronischen Verarbeitung eingescannter Dokumente. Dazu wird zunächst die Struktur eines Dokumentes analysiert, anschließend werden Zeichen und darauf aufbauend Wörter erkannt, damit dann innerhalb der partiellen Textanalyse Schlüsse über den Inhalt des Dokumentes gezogen werden können.

Das Forschungsgebiet der Dokumentanalyse existiert schon seit vielen Jahren, selten wurde damit jedoch eine Klassifikationsaufgabe in Verbindung gebracht. Aus diesem Grund existiert zu diesem Spezialthema kaum Literatur.

Ein ähnliches Forschungsgebiet, das *Information Retrieval*, befaßt sich mit der elektronischen Verarbeitung und Speicherung einer großen Menge von Textdateien und der Beantwortung von Suchanfragen über diesen Texten. Man benötigt Methoden, um auf die gewünschten Informationen zuzugreifen oder nach ihnen zu suchen. Dies gestaltet sich so, daß der Benutzer Suchanfragen stellt und das System ihm daraufhin die zu dieser Anfrage passenden Dokumente bereitstellt. Die Ähnlichkeiten, die zum Vergleich zwischen Suchanfragen und Dokumenten herangezogen werden, berechnen sich durch Vergleiche von Text- und Anfrageattributen (meist Schlüsselwörter).

Dokumentanalyse und Information Retrieval basieren auf unterschiedlichen Grundlagen. Im Information Retrieval arbeitet man mit korrekten ASCII-Texten, die entweder manuell eingegeben oder eingescannt und manuell überarbeitet werden. Die Dokumentanalyse beinhaltet eine eigene Texterkennung, die ein eingescanntes Dokument bearbeitet und Worthypothesen produziert. Diese Worthypothesen bilden die Eingangsdaten für die Textanalyse und sind häufig fehlerbehaftet.

Daher unterscheiden sich die Anforderungen, die an Systeme für die Dokumentanalyse gestellt werden, grundlegend von solchen für IR-Systeme. Trotzdem ist es interessant, erst einen Überblick über verschiedene existierende IR-Verfahren zu geben, um dann zu entscheiden, ob die Methoden oder Teile davon in den Bereich der Textanalyse übertragen werden können.

2 Statistische Verfahren

2.1 Klassischer Ansatz

2.1.1 Boolesche Schlüsselwortsuche

Die Boolesche Schlüsselwortsuche [Salton 83] basiert darauf, daß Anfragen als eine Boolesche Verknüpfung von Schlüsselwörtern dargestellt werden. Schlüsselwörter sind die Wörter aus den Texten, die übrigbleiben, wenn sog. *Stoppwörter* (Wörter mit sehr hoher Frequenz z. B. Konjunktionen, Artikel, Pronomen ...) eliminiert werden. Besteht eine Anfrage etwa aus der Disjunktion zweier Schlüsselwörter, werden alle Dokumente ausgegeben, die mindestens eines der Schlüsselwörter enthalten.

Damit die Dokumente nicht einzeln und wortweise auf das Vorkommen von Schlüsselwörtern untersucht werden müssen, werden *invertierte Indexdateien* angelegt. Dies sind

Listen, in denen für jedes Schlüsselwort eingetragen ist, in welchen Dokumenten es vorkommt. Dadurch ist eine schnelle Verarbeitung der Anfrage gesichert.

Da die Methode auf einer einfachen Technik beruht, ist sie einfach zu implementieren, schnell und sehr robust. Nachteile sind jedoch zahlreich:

Es werden nur die Texte gefunden, die das Schlüsselwort selbst enthalten, so daß Synonyme nicht erkannt werden. Steht z. B. in einer Anfrage das Wort 'Computer', im Text aber wird das Wort 'Rechner' verwendet, so wird dieser Text nicht als relevant erkannt.

Zudem kann das Schlüsselwort in einem völlig anderen als dem intendierten Kontext stehen, so daß ein Text ausgegeben wird, der für den Benutzer irrelevant ist. Außerdem werden die Texte ohne Reihenfolge ausgegeben, d. h. es gibt keinen *besten* Text. Der Benutzer muß unter Umständen viele für ihn irrelevante Dokumente durcharbeiten, was eigentlich vermieden werden sollte.

Aus diesen Nachteilen resultieren schlechte Genauigkeitswerte. Genauigkeit wird im Information Retrieval häufig an den beiden Indikatoren *Recall* und *Precision* gemessen. Der Recall gibt an, wieviele der relevanten Dokumente tatsächlich auf eine Anfrage hin gefunden wurden. Die Precision drückt aus, wieviele der gefundenen Dokumente relevant sind. Die Schlüsselwortsuche erreicht nur geringe Recall- und Precisionwerte.

Zu diesem Ansatz gibt es viele Erweiterungen:

- Man erlaubt in der Anfrage neue Parameter. Wenn A und B Schlüsselwörter sind und C ein Satz ist, dann bedeutet die Anfrage *A within sentence C*, daß A im Satz C auftreten muß, damit das Dokument ausgegeben wird. Die Anfrage *A adjacent B* verlangt, daß A und B hintereinander stehen sollen. Dieser Operator ermöglicht die Suche nach Mehrwortbegriffen.
- Die Schlüsselwörter werden mit Gewichten versehen, die die Relevanz des Wortes angeben. Der Benutzer erhält dadurch eine nach Wichtigkeit der Texte geordnete Ausgabe der Dokumente.
- In der Anfrage kann man zu einzelnen Schlüsselwörtern Synonyme angeben, die dann genauso wie die Schlüsselwörter behandelt werden.
- Man setzt einen Thesaurus ein, in dem zu Schlüsselwörtern jeweils Synonyme und ein Thesaurus-Klassen-Indikator stehen. In der internen Repräsentation der Texte und der Anfrage werden die Schlüsselwörter durch ihren Indikator ersetzt. Das System findet dann zu einer Anfrage alle Dokumente, die das Schlüsselwort selbst oder eines der Synonyme enthalten.
- Um die Effizienz zu erhöhen und den Speicherplatz zu verringern, ersetzt man Schlüsselwörter in der internen Repräsentation durch ihre Stammformen.

2.1.2 Das Vektorraummodell

Anfrage und Dokumente werden als Vektoren der in ihnen enthaltenen Terme dargestellt. Dabei sind die Werte der Vektorkomponenten meist binär. Wenn der Term in Anfrage bzw. Dokument enthalten ist, ist der Wert der entsprechenden Komponente 1, sonst 0.

Diese Vektoren spannen einen Vektorraum auf, dessen Dimension abhängig ist von der Anzahl der Terme. Anfrage und Dokumente werden nun über die Vektoren verglichen, etwa durch Benutzung des Kosinus-Ähnlichkeitsmaßes [Salton 89]. Zu einer Anfrage findet man so die ähnlichsten Vektoren. Dies bedeutet, daß die Dokumente gefunden werden, die viele Wörter mit der Anfrage gemeinsam haben. Auch hier gibt es eine Verbesserung. Man gewichtet die Terme, um eine Bedeutung der Wörter für die Dokumente einzuführen.

Für die Gewichtung existieren folgende Funktionen:

- Nimmt man die *Frequenz* des Termes in der Dokumentbasis als Gewicht, hat man ein einfaches, aber nicht normiertes Maß.
- Die *inverse Dokumenthäufigkeit* liefert einen hohen Wert, wenn der Term häufig in einem speziellen Dokument auftritt, aber in Relation dazu selten in allen Dokumenten.
- Der *Informationswert* eines Termes ist hoch, wenn die relative Wahrscheinlichkeit des Auftretens des Termes gering ist.
- Der *Diskriminanzwert* eines Termes gibt an, wie sehr der Begriff geeignet ist, um zwischen Dokumenten zu unterscheiden.

Eine gute Beschreibung des Vektorraummodells findet sich in [Salton 89].

2.1.3 Das probabilistische Modell

Dem probabilistischen Modell [Salton 89] liegt die Idee zugrunde, zu einer Anfrage diejenigen Dokumente auszugeben, die mit der höchsten Wahrscheinlichkeit relevant für den Benutzer sind. Das Modell berücksichtigt, daß sowohl die Textrepräsentation als auch die Anfragenformulierung (als Ausdruck des Informationsbedarfs) unsicher sind und infolgedessen die Relevanzbeziehung zwischen ihnen ebenfalls.

Als Quellen für die Berechnung der Wahrscheinlichkeit benutzt man die statistische Verteilung der Terme in der Dokumentbasis und die Trennung der Dokumente in relevante und nicht relevante Dokumente.

Da die Auftrittswahrscheinlichkeiten der Terme in relevanten und nicht relevanten Dokumenten aber nur geschätzt werden können, ist diese Methode für die Praxis nicht so gut geeignet. Sie liefert allerdings eine theoretische Begründung für den Einsatz von Gewichtungsfunktionen und hilft, Retrieval-Prozesse besser zu charakterisieren.

2.1.4 Clustering

Unter *Clustering* [Salton 89] versteht man die Einteilung vieler gleichartiger Elemente in mehrere Gruppen. Im Information-Retrieval werden die Dokumente in Cluster eingeteilt, so daß jedes Cluster ähnliche Dokumente enthält. Dafür wird eine Vektordarstellung der Dokumente benötigt, auf die man dann ein Ähnlichkeitsmaß anwendet.

Zur Definition eines Clusters dient sein *Zentroid.* Dieser Zentroid besteht aus einem künstlichen Dokument, das genau in der Mitte seines Clusters liegt. Die Zentroiden aller Cluster können wieder in Gruppen eingeteilt werden, die dann ihrerseits durch sogenannte *Hyperzentroiden* identifiziert werden können. Dieser Vorgang kann solange iteriert werden, bis die Anzahl der Zentroiden der gewünschten Größenordnung entspricht. Man erhält dann eine hierarchische Anordnung von Clustern, wobei die Dokumente auf der untersten Ebene zu finden sind.

Die Beantwortung der Anfrage reduziert sich durch diese Struktur auf Vergleiche der Anfrage mit den Zentroiden und erst zuletzt mit Dokumenten. Zuerst erfolgt ein Vergleich mit den Zentroiden der höchsten Ebene. Die ähnlichsten Zentroiden werden zur Weiterverarbeitung ausgewählt, d. h. anschließend wird die Anfrage mit allen Elementen der Cluster der ähnlichsten Zentroiden verglichen usw. Der Vorgang stoppt, wenn die Anfrage auf der untersten Ebene mit den Dokumenten selbst verglichen wurde. Der Benutzer erhält die zur Anfrage ähnlichsten Dokumente nach ihrer Ähnlichkeit geordnet.

2.1.5 Relevanz-Feedback

Relevanz-Feedback gibt dem Benutzer die Möglichkeit, die Ergebnisse des Retrieval-Prozesses durch Rückkopplung zu verbessern. Dies geschieht zumeist dadurch, daß der Benutzer nach Ausgabe der gefundenen Texte angibt, welche der Dokumente relevant und welche weniger relevant für ihn sind. Mit Hilfe diese zusätzlichen Wissens startet dann eine neue Suche.

Bei einer einfachen Booleschen Schlüsselwortsuche nutzt das System diese zusätzliche Information, um der ursprünglichen Anfrage Schlüsselwörter aus den relevanten Dokumenten hinzuzufügen. Dies entspricht einer Spezialisierung der Anfrage, so daß eine erneute Suche zu verbesserten Ergebnissen führen kann.

Welche Terme der Anfrage hinzugefügt werden, kann entweder durch den Algorithmus oder durch den Benutzer bestimmt werden. Die Bestimmung durch den Benutzer hat den Vorteil, daß die zusätzlichen Terme relevant sind und den Informationswunsch des Benutzers widerspiegeln. Mehr über Relevanz-Feedback ist in [Salton 89] zu finden.

2.1.6 Das SMART-System

Das wohl bekannteste klassische System heißt SMART [Salton 71] und ist schon über zwanzig Jahre alt. Das SMART-System basiert auf den obigen Methoden wie Relevanz-Feedback, Clustering und Wortstamm-Reduktion. Es enthält mehrere statistische Ähnlichkeitsfunktionen und verschiedene Wörterbücher.

Ein Wörterbuch enthält die Liste der Stoppwörter, ein anderes realisiert einen Thesaurus von Wortstämmen mit Konzeptnummern für eine Klasse von Synonymen. Schließlich existiert noch ein Phrasenwörterbuch mit Mehrwortbegriffen und eine Sammlung von Termen und Konzepten in hierarchischer Anordnung.

Die Dokumente werden um mehrere Zentroiden gruppiert. Daher vergleicht das System eine Anfrage zuerst mit allen Zentroiden, um dann bestimmte Zentroide auszuwählen, deren Dokumente danach durch Schlüsselwortsuche untersucht werden.

Obwohl das SMART-System schnell brauchbare Ergebnisse liefert, beinhaltet es doch die Nachteile seiner Methoden. Die Anfrage ist eingeschränkt durch Verwendung von Schlüsselwörtern und deren Boolesche Verknüpfung. Manchmal kennt der Benutzer die genaue Terminologie des Gebietes, über das er sich informieren möchte, gar nicht und erhält dann durch eine schlechte Formulierung der Anfrage mangelhafte Information.

Daher werden die Recall-Werte eines klassischen Systems schlecht, wenn in der Dokumentbasis viele Dokumente enthalten sind, die zwar gleiche Themen zum Inhalt haben, aber unterschiedliches Vokabular benutzen. Setzt man einen Thesaurus für Synonyme ein, kann dieses Problem teilweise behoben werden. Jedoch geschieht die Konstruktion eines Thesaurus meist manuell und kostet viel Zeit. Da klassische Systeme wortbasiert sind, können sie keine Kontexte eruieren, so daß einige der gefundenen Dokumente zwar ein Schlüsselwort enthalten, welches aber in einem anderen Sinnzusammenhang steht[1].

2.2 Der Eigenvektoren-Ansatz

Als latentes semantisches Indexieren (kurz LSI) [Lochbaum 87] wird ein statistischer Ansatz bezeichnet, der auf einer Zerlegung von Merkmalsmatrizen in Eigenvektoren (*singular value decomposition—SVD*) basiert.
Man zerlegt eine Term×Dokument-Matrix, die zu jedem Dokument die Häufigkeit der in ihm auftretenden Terme enthält, in viele, typischerweise 50–150 orthogonale Faktoren. Durch eine Linearkombination dieser Vektoren kann dann die ursprüngliche Matrix angenähert werden. Diese Annäherung filtert Störungen der ursprünglichen Matrix heraus.

Dokumente und Terme ordnet man als Punkte in einem z. B. 100-dimensionalen Raum an. Es werden nur solche Terme benutzt, die in mehr als einem Text vorkommen, da sie die Ähnlichkeiten zwischen Dokumenten widerspiegeln sollen. Daher liegen nun einerseits Terme nahe beieinander, die in gleichen Kontexten verwendet wurden (was ein Indiz dafür ist, daß sie Synonyme sind) und andererseits liegen Dokumente nahe beieinander, die viele Wörter gemeinsam haben (was ein Indiz dafür ist, daß sie gleiche Themen behandeln). So schafft man es, auf rein statistischer Basis auch semantische Korrelationen zwischen Termen darzustellen, die in der Dokumentbasis nicht ersichtlich waren.

Die Beantwortung von Benutzeranfragen erfolgt, indem sie selbst im Dokumentraum eingetragen und dann über ein Ähnlichkeitsmaß mit den Dokumenten verglichen werden. Die nächsten Nachbarn der Anfrage, die die meisten ähnlichen Terme verwenden, werden zurückgeliefert.

Das folgende Beispiel soll diese Ausführungen verdeutlichen. Hat man zwei Gruppen von Dokumenten, die zwei verschiedene Sachgebiete behandeln, so findet sich diese Zweiteilung auch im Dokumentraum wieder, unabhängig davon, welches Vokabular benutzt wurde. Auch ähnliche Terme wie z. B. Mensch und Benutzer liegen nahe zusammen. Deshalb kann eine Anfrage Synonyme und überdies thematisch ähnliche Texte finden.

Ergebnisverbesserungen erzielt man durch

- Formulierung von Anfragen, die aus vielen Termen bestehen und daher eine genauere Plazierung der Anfrage im Dokumentraum ermöglichen;

[1] Beispielsweise existieren Mehrdeutigkeiten bei Bank, Strauß,...

- Erkennung von Mehrwortbegriffen in Dokument und Anfrage;
- Benutzung gewichteter Terme in der ursprünglichen Matrix, wobei sich Entropie und inverse Dokumenthäufigkeit als beste Gewichtsfunktionen erwiesen haben;
- Umwandlung der Vollformen in Stammformen.

Eine Verschlechterung ergibt sich, wenn auch Terme, die nur in einem bestimmten Dokument auftreten, in den Dokumentraum einbezogen werden. Sie können keine Relationen zwischen Dokumenten aufdecken und bewirken überdies eine stärkere Streuung der Dokumente im Vektorraum.

Probleme treten auf in der Behandlung von Polysemie, da in der SVD-Repräsentation ein Wort, das in verschiedenen Kontexten benutzt wird, als Zentroid seiner verschiedenen Bedeutungen dargestellt wird, was nur einen schädlichen Einfluß hat. Normalerweise kann ein Wort im Vektorraum die räumliche Trennung zweier Sachgebiete bewirken. Tritt es aber in verschiedenen Kontexten auf, werden diese nicht getrennt, sondern nähern sich an.

Ein weiterer Mangel der LSI-Methode ist die Nichteinbeziehung syntaktischer Strukturen. Dadurch wird unter Umständen die eigentliche Semantik eines Wortes nicht gefunden, wenn sie erst durch den Kontext festgelegt wird.

Positiv überraschend ist folgende Tatsache: Obwohl der LSI-Ansatz bereichsunabhängig und daher portabel ist, löst er das Problem der Synonymie. Überdies ist der Ansatz robust und schnell aufgrund seiner mathematischen Basis.

Sinnvoll ist die Kombination von SVD und Vektorraummodell. Das Vektorraummodell findet in solchen Fällen die Dokumente zu einer Anfrage, in denen der Benutzer das präzise Vokabular des Gebietes kennt und seine Anfrage korrekt formuliert ist. Ist die Anfrage hingegen 'schlecht' formuliert, benutzt man die SVD-Methode, um Synonyme bei der Suche mit einzubeziehen.

2.3 Erweiterter klassischer Ansatz

Im Vektorraummodell werden Texte als ähnlich bewertet aufgrund globaler Ähnlichkeiten. Da dieser Ansatz den Nachteil hat, daß keine Ähnlichkeiten zwischen kleineren Texteinheiten — gleich welcher Größe — berücksichtigt werden, wird der Kontext von Wörtern vernachlässigt.

Der erweiterte klassische Ansatz [Salton 91, Salton 92] versucht, diesen Mangel durch Einführung lokaler Ähnlichkeiten zu beseitigen. Dazu erfolgt eine Einteilung aller Dokumente in Textfragmente, die Kontexte widerspiegeln sollen. Die Größe der Fragmente ist unterschiedlich, sie können einige Wörter, einen Satz oder einen Paragraphen umfassen.

Die Terme werden nach Entfernung von Stoppwörtern wie vorher durch die inverse Dokumenthäufigkeit indexiert. Dann stellt man Dokumente, Textfragmente und die Anfrage — analog zum Vektorraummodell — als Vektoren der in ihnen enthaltenen Terme dar. Die Beantwortung geschieht dadurch, daß Ähnlichkeiten zwischen Vektoren berechnet und die Dokumente zurückgeliefert werden, deren Vektoren am ähnlichsten zur Anfrage sind.

Neu ist aber, daß ein Text nur dann ausgegeben wird, wenn sowohl globale als auch lokale Ähnlichkeitswerte bestimmte Schwellwerte übersteigen. Die lokalen Ähnlichkeitswerte berechnen sich durch Vergleich der Textfragmente mit der Anfrage.

Allerdings muß man die Gewichtsfunktion für die Textfragmente abändern. Die inverse Dokumenthäufigkeit enthält eine Normalisierung über die Dokumentlänge, um allen Texten gleiche Chancen zu ermöglichen. Dadurch jedoch können auf lokaler Ebene Textstücke mit wenigen Deskriptoren (z. B. Redewendungen) den Textvergleich kontrollieren. Da dieser Effekt unerwünscht ist, setzt man eine nicht normalisierende Gewichtsfunktion ein.

Auch diese Methode kann durch Retrieval-Feedback verbessert werden, indem die im ersten Schritt gefundenen Dokumente als neue Anfragen benutzt werden. Durch die Einführung lokaler Ähnlichkeiten ist die Verwendung eines Thesaurus überflüssig geworden. Ein System, das auf dieser Methode basiert, ist daher bereichsunabhängig und hat eine hohe Portabilität. Es kann große Textmengen verschiedener Länge bearbeiten und berücksichtigt Kontexte beim Retrieval-Prozeß. Da der Kern des Systems aber immer noch wortbasiert ist, bleiben syntaktische oder semantische Textstrukturen unerkannt.

3 Regelbasierte Verfahren

Die Benutzung von Regeln im Information Retrieval stellt einen Mittelweg zwischen klassischen Methoden und Verwendung von NLP dar. Durch Verwendung von Regeln wird versucht, den Text auf einer konzeptuellen Ebene und nicht vollständig oder in großen Teilen, wie durch einen Parser, zu analysieren.

Deshalb haben alle regelbasierten Verfahren ähnliche Merkmale. Sie sind schneller als Systeme, die Parsing-Techniken verwenden, und sie sind genauer als klassische Verfahren.

3.1 TCS/CONSTRUE

TCS [Hayes 90] ist ein relativ einfaches, regelbasiertes System. Es bildet eine Shell für Textkategorisierungssysteme. Solche Systeme identifizieren das Thema eines Textes und ordnen es einer oder mehreren Kategorien zu. Das größte der mit TCS gebildeten Kategorisierungssysteme heißt CONSTRUE [Hayes 88, Hayes 92] und klassifiziert Zeitungsmeldungen aus dem finanziellen und wirtschaftlichen Sektor in eine oder mehrere von 674 Kategorien. Diese Kategorien bestehen aus Namen von Ländern, Firmen und Organisationen und aus wirtschaftlichen Kategorien wie Währungen, Waren und Lohnlisten. Die Kategorisierung eines Textes basiert auf der Anwesenheit bestimmter Wörter und Phrasen in bestimmten lexikalischen Kontexten.

Kernstück von CONSTRUE ist die Regelbasis, die Bereichswissen enthält. Da TCS nur die Shell für ein Kategorisierungssystem bildet, ist seine Regelbasis anfänglich leer. Sie kann mit dem TCS-Editor gefüllt bzw. verändert werden und beinhaltet dann Kategorisierungsregeln und Konzeptdefinitionen. Die Kategorisierungsregeln bauen auf Konzepten und Konzeptdefinitionen auf. Zum Aufbau der Regeln für Konzepte existiert eine eigene Patternsprache. Sie besteht aus einer Erweiterung der Booleschen Schlüsselwortsprache um folgende Operatoren:

- Wort- oder Phrasen-Alternativen können angegeben werden (Operator: !)
- optionale Phrasen oder Wörter können angegeben werden (Operator: ?)
- Wörter oder Phrasen, die nicht vorkommen dürfen, können spezifiziert werden (Operator: `&not`)
- Wörter können übersprungen werden, wobei die Anzahl der Wörter und die Richtung des Skips angegeben werden können (Operator: `&skip`)
- Es kann angegeben werden, daß ein Wort ein Nomen ist und daher im Plural auftreten kann (Operator: `+N`)
- Es kann angegeben werden, daß ein Wort ein Verb ist und in konjugierter Form auftreten kann (Operator: `+V`)

Das Pattern `(gold (&not (medal!reserve!jewelry)))` steht z. B. für Gold als Handelsware. Dadurch wird 'Goldmedaille' nicht erfaßt, wohl aber der Begriff 'Goldmine'. Jedes Pattern in einer Konzeptdefinition wird gewichtet (z. B. mit *wahrscheinlich* oder *möglich*), um anzugeben, wie weit es das Vorhandensein des Konzeptes determiniert. Diese Gewichtung kann einfließen bei der Entscheidung, wie oft ein Konzept vorhanden ist, was bei der Festlegung der Kategorie wichtig ist.
Kategorisierungsregeln werden als Wenn-Dann Regeln formuliert und bestehen aus Booleschen Kombinationen von Konzepten oder der Stellenangabe des Auftretens des Konzeptes im Text[2]. Die Definition:

```
(if test: (or (and [gold-concept :scope headline 1]
                   [gold-concept :scope body 1])
              (or [gold-concept :scope body 4])
    action: (assign gold-category) ...)
```

besagt, daß die Gold-Kategorie zugewiesen wird, wenn das Gold-Konzept entweder einmal in der Überschrift und einmal im Text oder viermal im Text vorkommt. Dadurch wird dem Gold-Konzept eine größere Bedeutung zugemessen, wenn es in der Überschrift auftritt.

Gibt man einen neuen Text in das System, versucht das System zunächst, die Konzepte im Text zu finden. Durch die gefundenen Konzepte können dann die Kategorien festgelegt werden können. Die Genauigkeit von CONSTRUE zeigt sich in einem Recall von 94% und einer Precision von 84%. Diese Werte erscheinen zunächst erstaunlich hoch, man muß allerdings bedenken, daß die Kategorisierungsaufgabe mit einigen, fest definierten Kategorien leichter zu erfüllen ist als die Beantwortung einer natürlichsprachlichen Anfrage.

Mit der TCS-Shell können ähnlich gute Werte erreicht werden. Sie sind natürlich abhängig von der Güte der Regeln, mit denen TCS gefüllt wird. Weitere Vorteile von TCS sind Schnelligkeit (da auf ein Parsen der Dokumente verzichtet wird), leichte Portabilität (da bereichsunabhängig) und Robustheit. Nachteilig ist der hohe Zeitbedarf für die manuelle Erstellung der Regeln (für CONSTRUE ungefähr vier Mannjahre). Deshalb arbeitet man an Methoden für eine halbautomatische Patterngenerierung.

[2](z. B. Überschrift, Textrumpf)

3.2 RUBRIC

Ein anderes, wesentlich mächtigeres und umfangreicheres regelbasiertes System heißt RUBRIC [Tong 85, Tong 89]. Die Regelbasis von RUBRIC enthält ebenfalls Konzeptdefinitionen. Diese sind allerdings hierarchisch in einer Baumstruktur angeordnet. Jeder Baum repräsentiert ein Thema durch seine Konzepte und Teilkonzepte.

Texte werden als Evidenzen für Konzepte betrachtet. Da diese Ansicht aber unsicher ist und nicht durch eine syntaktische Textanalyse bestätigt wird, werden die Unsicherheiten numerisch repräsentiert und durch die Baumstruktur propagiert. Die Konzepte werden definiert durch Regeln der Form:

Patternausdruck $\Rightarrow$ Konzept (Wahrscheinlichkeit)

Dabei besteht der Patternausdruck aus einer Konjunktion von Teilkonzepten oder Schlüsselwörtern. Das Konzept ist ein Konzept aus einer höheren Ebene und die Wahrscheinlichkeit gibt an, wie sehr der Benutzer glaubt, daß das Pattern der Regel die Anwesenheit des Konzeptes determiniert.

Man kann sich den Baum als Und/Oder-Baum vorstellen, da einerseits der Patternausdruck aus einer Konjunktion bestehen und andererseits ein Konzept im Rumpf mehrerer Regeln stehen kann. Eine Kante im Baum[3] kann mit einem Gewicht zwischen 0 und 1 versehen werden. Dieses Gewicht drückt z. B. einen partiellen Wahrheitswert für die Regel aus (measure of belief = Verstärkung der Annahme durch den Patternausdruck, daß das Konzept zutrifft). In der untersten Schicht, den Blättern des Baumes, bestehen die Pattern-Ausdrücke lediglich aus einer Verknüpfung von Schlüsselwörtern. Hier treten keine Konzepte auf.

Durch Benennung eines Themas in der Anfrage startet der Benutzer automatisch eine zielorientierte Suche in dem Teil des Baumes, der durch dieses Thema definiert wird. Innerhalb dieses Teilastes erfolgt ein Abstieg zu seinen Blättern. Da diese aus einer Verknüpfung von Schlüsselwörtern bestehen, können sie mit den Dokumenten verglichen werden. Trifft ein Patternausdruck zu, kann die entsprechende Regel feuern und selbst wieder andere Regeln aktivieren.

Die Propagierung der Wahrscheinlichkeit durch das Netz kann auf viele Arten geschehen. Sie ist abhängig von der Wahl des zugrundeliegenden Unsicherheitskalküls. Eine einfache Möglichkeit ist die folgende: Die Gewichte der Kanten werden direkt weitergegeben. Falls jedoch mehrere Werte vorhanden sind, wird im Und-Zweig das Minimum der Werte und im Oder-Zweig das Maximum weitergegeben. Die Suche ist beendet, wenn die Propagierung der Wahrscheinlichkeitswerte wieder bei dem in der Fragestellung benannten Thema angelangt ist. Durch Vergleich mit einem Schwellwert wird entschieden, ob die Wahrscheinlichkeit hoch genug ist, daß der Text das gesuchte Thema beinhaltet und somit ausgegeben werden soll.

Wie schon erwähnt, kann man verschiedene Unsicherheitskalküle wählen, was sich dann in der Güte der Ergebnisse widerspiegelt. Ebenso wichtig ist die Festlegung der Wahrschein-

[3] entspricht dem $\Rightarrow$ der Regel

lichkeiten im Baum, die keine objektiven Wahrscheinlichkeiten, sondern die Einstellungen des Benutzers darstellen.

Ein großer Vorteil dieses Systems ist seine Fähigkeit, unsichere Werte und somit konkrete Relationen zwischen Teilkonzepten und Konzepten miteinzubeziehen. Daher erreicht RUBRIC auch sehr gute Genauigkeitswerte. Außerdem bietet die Benutzung verschiedener Unsicherheitskalküle Spielraum für Vergleiche der Kalküle und es können mehrere Ansichten über probabilistisches Schließen eingebracht werden. Allerdings liegt auch gerade in der Verwendung solcher Kalküle ein Problem, da sich gezeigt hat, daß sie keine Normierung beinhalten. So liefert eine lange Kette von Schlüssen mit Propagierung der Wahrscheinlichkeiten immer schlechtere Werte als kurze Ketten, obwohl dies der Anschauung widerspricht, daß ein komplexeres Regelsystem mit mehreren Hierarchien genauere (aber keine schlechteren) Werte liefern sollte als ein einfaches Regelsystem.

3.3 Andere Systeme

JASPER [Hayes 92] ist ein System zur Extraktion von Fakten aus Dokumenten, d. h. die wesentlichen Informationen eines Textes werden durch dieses System gewonnen. Es extrahiert aus Geschäftsberichten Informationen über Dividenden und Jahresüberschüsse und wandelt diese in Zeitungsberichte um. JASPER kombiniert eine framebasierte Wissensrepräsentation mit objektorientierter Verarbeitung und benutzt dieselbe Art des Pattern-Matching wie CONSTRUE und TCS. Gute Genauigkeitswerte (76% Recall, 92% Precision) und eine schnelle Verarbeitung zeichnen JASPER aus.

Das RUBRIC-System wird innerhalb des CODEX-Systems [Tong 91] als *Profiler*[4] verwendet. CODEX ist ein textextrahierendes System. Das System wird durch einen Controller gesteuert, der nach Texteingabe RUBRIC aufruft, um relevante Konzepte zu erhalten und dadurch die wichtigsten Teile des Textes festzulegen. Diese Textfragmente dienen dann als Eingabe für den CAUCUS-Parser, der aus ihnen die benötigte Information extrahiert. CODEX wurde auf der MUC-3 Konferenz getestet, wobei allerdings der Parser nicht eingesetzt werden konnte. Es wurden keine besonders guten Ergebnisse erreicht, was aufgrund des fehlenden Parsers nicht sehr aussagekräftig ist.

4 Konzeptbasierte Verfahren

Unter diesen Bereich fallen solche Systeme, in denen man versucht, mittels Einsatz eines Parsers die syntaktische und die semantische Struktur des Textes zu erhalten und damit den Text besser zu verstehen und Kontexte zu erfassen.

4.1 FRUMP

Das FRUMP-System *(Fast Reading and Understanding Memory Program)* [DeJong 82] ist ein Programm, das Zeitungsnachrichten überfliegt und zusammenfaßt. Es verwendet dazu

[4] Ein Profiler reduziert den Text auf seine wesentliche Abschnitte.

die Technik des *Skimming*, wobei von einem Text nur soviel gelesen und geparst wird, wie gerade notwendig ist. Ist das Thema eines Textes gefunden und wurden alle wichtigen Informationen dazu extrahiert, bricht der Analysevorgang ab, unabhängig davon, ob der ganze Text bereits bearbeitet wurde oder nicht. Dabei kann es geschehen, daß das Hauptthema eines Textes gar nicht gefunden wird, da die Analyse schon geendet hat.

Skimming ist schneller und einfacher als vollständiges, tiefes Parsen, das für lange Sätze und große Dokumentbasen im Information Retrieval zu langsam ist. Außerdem ist die Skimming-Technik meist ausreichend, um die nötigen Informationen zu ermitteln und grammatikalisch falsche Eingaben führen nicht zum Scheitern des Parses.

Die Vorgehensweise von FRUMP ist erwartungsgesteuert (top-down). FRUMP benutzt sein pragmatisches und syntaktisches Wissen, um allgemeine Ereignisse aufgrund der bisherigen Geschehnisse vorherzusagen. Dann versucht die Textanalyse, Instanzen dieser Hypothesen im Text zu finden. Je nachdem, ob die Hypothesen zutreffen oder nicht, interpretiert FRUMP die Situation neu und erzeugt neue Vorhersagen.

FRUMP benutzt sog. *Sketchy Scripts* zur Kodierung seines Weltwissens. Ein Script ist eine framebasierte Wissensrepräsentation. Es enthält mehrere Frames[5], die einen Sinnzusammenhang bilden. FRUMP enthält Sketchy Scripts für Situationen wie Erdbeben, Streiks, Demonstrationen, militärische Aktionen usw. Zwei Komponenten arbeiten in FRUMP zusammen.

- Der *Predictor* sagt aus dem aktuellen Kontext und der Kenntnis der Konzepte der Welt mit Hilfe der Scripts mögliche Fortsetzungen des Textes voraus.
- Die Sketchy Scripts werden dann dem *Substantiierer* übergeben. Dieser versucht, die Scripts mit Hilfe syntaktischen Wissens aus dem Text anzufüllen.

Diese Suche ist mehr als eine Suche nach Schlüsselwörtern, denn es wird eine Bedeutung innerhalb eines Kontextes gesucht. Als Ausgabe liefert FRUMP eine konzeptualisierte Darstellung des Ereignisses, das im Eingabetext beschrieben wurde.

FRUMP ist direkt an ein Presse-Netz angeschlossen und verarbeitet Kurznachrichten in durchschnittlich 20 Sekunden. Ungefähr 10% der Scripts werden korrekt zugeordnet. In den restlichen 90% der Fälle fehlt entweder Wissen, Aktionen werden falsch gedeutet oder die falschen Scripts werden angewandt. Der FRUMP-Parser ist mehrfach weiterentwickelt worden und taucht in anderen Systemen wieder auf.

4.2 FERRET

Das FERRET-System *(Flexible Expert Retrieval of Relevant English Texts)* [Mauldin 87, Mauldin 91] verwendet den McFrump Parser, um Scripts mit Wissen über den Text anzufüllen. Dieser Parser benutzt einfachere Scripts als FRUMP und beinhaltet neuere Scripts, seine Vorgehensweise ist gleich geblieben. Um die semantische Struktur des Textes zu eruieren, genügt der Parser allein nicht. Dafür benutzt FERRET zusätzlich Wörterbücher:

[5] Menge von Attribut-Wert-Paaren

- ein handcodiertes Lexikon mit Redewendungen, Wortbedeutungen und Konzepteinträgen für über 12000 Frames
- ein Lexikon für Synonyme mit über 58000 Einträgen
- ein Lexikon mit Wörtern, die in der Definition unbekannter Worte benutzt werden (near-synonyms)
- ein Lexikon mit Vor- und Nachnamen

Eine Anfrage im FERRET-System wird dadurch beantwortet, daß sie in eine Frame-Darstellung transformiert und mit den Frames der Texte verglichen wird.

Vorteile des FERRET-Systems sind seine Robustheit und seine Schnelligkeit im Vergleich zu Systemen mit konventionellen Parsern und das bessere Textverstehen[6] im Vergleich zu konventionellen IR-Systemen. Überdies bietet die Frame-Darstellung die Möglichkeit der Übersetzung der Texte in andere Sprachen.

Nachteile sind der Zeitaufwand für die Textbearbeitung und die manuelle Erstellung der Schemata für die Scripts.

4.3 SCISOR

SCISOR *(System for Conceptual Information Summarization, Organization and Retrieval)* heißt ein System [Rau 88, Rau 90], das auch einen Parser zum Textverstehen benutzt, aber Informationen in Konzepten abspeichert. SCISOR verarbeitet Meldungen über geplante, erfolgreiche und gescheiterte Firmenübernahmen. Auf natürlichsprachliche Anfragen kann das System einfache Antworten wie ja/nein liefern, eine zeitliche Ablaufreihenfolge über mehrere Dokumente ausgeben und neue Informationen in eine Ablaufreihenfolge integrieren. Außerdem ist es in der Lage, unvollständige oder fehlerhafte Eingaben teilweise zu tolerieren.

SCISOR ist — ähnlich wie FRUMP — an ein Informationsnetz angeschlossen und integriert ankommende Texte in seine Wissensbasis, wenn sie zu seinem Bereich gehören. Die Text-Verarbeitung in SCISOR kombiniert zwei Analysestrategien: eine *bottom-up* und sprachgesteuerte Interpretation und eine *top-down*, erwartungsgesteuerte Verarbeitung.

- Die bottom-up Analyse, die durch den Parser *TRUMP* ausgeführt wird, beginnt damit, ankommende Texte satzweise zu parsen. Dabei werden linguistische Strukturen entdeckt und in konzeptuelle Strukturen überführt.
- Die top-down Analyse beginnt mit konzeptuellen Erwartungen, z. B. mit dem Wissen, daß bei einer Firmenübernahme zwei Firmen involviert sind, und versucht diese Erwartungen mit partiellen Informationen aus dem Text (die der Parser liefert) anzufüllen.

[6] Verarbeitung von Synonymie und Polysemie

Bottom-up Interpretation liefert sehr genaue Parsing-Ergebnisse und entdeckt fehlerhafte Eingaben. Top-down Methoden sind toleranter gegenüber unbekannten Wörtern und lexikalischen Fehlern, liefern aber eher falsche Interpretationen. Durch die Kombination beider Methoden in einer natürlichsprachlichen Komponente kann die Tiefe und Genauigkeit des Analyseprozesses verbessert werden. Die Mängel der einzelnen Methoden fallen weniger stark ins Gewicht.

SCISOR verwendet vier verschiedene Wissensquellen, um die Bedeutung aus Texten zu extrahieren:

Role-Filler Erwartungen: Sie sind die erste Informationsquelle für die erwartungsgesteuerte Analyse und bestehen aus Einträgen der Form (Attribut, Bedingungen für den Wert, Beispiel)[7]. Im Laufe der Textverarbeitung werden Instanzen dieser Erwartungen gefüllt.

Ereigniserwartungen: Erwartungen über Ereignisse, die in der Zukunft geschehen können, werden mit Hilfe bekannter Nachrichten erzeugt und benutzt, um Werte vorherzusagen, wenn die Ereignisse tatsächlich eintreten.

Linguistisches Wissen: Grammatisches, lexikalisches und bereichsabhängiges Wissen werden vom Parser und dem Sprachgenerator für die natürlichsprachliche Ausgabe (*King-Generator*) benutzt.

Weltwissen-Erwartungen: Das Weltwissen enthält bereichsabhängige Verallgemeinerungen, die als Relationen zwischen Konzepten implementiert sind.

Ein Eingabetext durchläuft mehrere Analyseschritte: Der *Topic Analyzer*, ein spezieller Filter, entscheidet, ob der Text zum Bereich der Firmenübernahmen gehört und weiterverarbeitet wird. Bestehend aus angereicherten, wortbasierten Verfahren übernimmt er eine duale Klassifikationsaufgabe und Teile der lexikalischen Analyse. Die natürlichsprachliche Komponente verarbeitet die Texte und erstellt Konzepte, die dann in die Wissensbasis eingetragen werden.

Die Antwort auf eine Suchanfrage gliedert sich in zwei Phasen. Zuerst wird mittels einer einfachen Suche der Kontext aufgespannt, in dem nach einer Antwort zu suchen ist. Anschließend wird in einer teuren Suche durch Graphen-Matching die korrekte Antwort festgelegt. In Tests erreicht SCISOR Genauigkeitswerte von 80-90% (Kombination von Recall und Precision) bei der Extraktion wichtiger Fakten aus den Texten. Das System ist portabel und durch die Kombination von top-down und bottom-up Verfahren robuster als einfachere, Parserbasierte Systeme.

Nachteilig ist der hohe Zeitaufwand für die manuelle Erstellung der vier Wissensquellen.

4.4 CIRCUS

Das CIRCUS-System benutzt ein Verfahren der *Informationsextraktion* als Basis für eine Text-Klassifikation mit hoher Precision. Es wurde auf der MUC-3 Konferenz getestet [Lehnert 91] und extrahiert Fakten aus Nachrichten über terroristische Anschläge.

[7] Ein Beispiel aus dem Bereich der Firmenübernahme ist etwa: (Preis pro Aktie, kleine Zahl, 45$).

Der CIRCUS-Parser, der die Information extrahiert, liefert als Ausgabe instantiierte Konzeptknoten. Ein Konzeptknoten wird erzeugt, wenn ein bestimmtes Wort aus einer vordefinierten Menge von Wörtern (sog. *Triggerwörter*) in einem speziellen linguistischen Kontext im Dokument auftritt. Er besteht aus einem Slot für die Triggerwörter und mehreren Slots für Informationen aus dem Text, wobei in diesen Slots syntaktische Hilfen stehen können, die angeben, an welcher Stelle im Text die spezielle Information gefunden werden kann. Außerdem können die Slots mit weichen oder harten Constraints verbunden sein. Weiche Constraints geben Präferenzen für bestimmte Informationen an, harte Constraints definieren Bedingungen, die zum Füllen des Slots eingehalten werden müssen.

Diese Art der Informationsextraktion nennt man auch *selektive Konzeptextraktion.* Die Konzeptknoten können zur Textklassifikation auf verschiedene Arten weiterverarbeitet werden:

- Ein einfacher Algorithmus, der linguistische Zusammenhänge berücksichtigt, ist der *Algorithmus für relevante Signaturen.* Eine Signatur besteht aus einem Paar (Wort, Konzeptknoten), wobei das Wort das Triggerwort für die Erzeugung des Knotens war. Eine Signatur wird als relevant bezeichnet, wenn Texte, in denen sie auftritt, stark mit einer bestimmten Klasse korrelieren.

  Der Algorithmus gliedert sich in eine Trainings- und eine Testphase. In der Trainingsphase erzeugt er eine Statistik über die Verteilung der Signaturen in relevanten und irrelevanten Texten. Übersteigen die Werte des Auftretens einer Signatur in relevanten Texten bestimmte Schwellwerte, so wird die Signatur als relevant für die Klasse angesehen. Mit diesen Schwellwerten kann man den Tradeoff zwischen Recall und Precision steuern. Hohe Schwellwerte liefern wenige relevante Signaturen, haben daher eine hohe Precision und einen niedrigen Recall und umgekehrt.

  In der Testphase werden Texte mit mindestens einer relevanten Signatur als relevant eingestuft.

  Dieser Algorithmus liefert falsche Ergebnisse, wenn der Kontext in der Umgebung eines Triggerwortes zusätzliche Information liefert, durch die der Text irrelevant wird.

- Um dieses Problem zu lösen, verwendet man den *Algorithmus für angereicherte relevante Signaturen.* Dazu wird die im Konzeptknoten enthaltene Information über den Kontext der Triggerworte in einem Slotfüller der Form (Konzeptknotentyp, Slotname, semantische Eigenschaft) abgebildet.

  Eine angereicherte relevante Signatur besteht aus einer Signatur und einem Slotfüller, die beide stark korrelieren mit der Zuweisung einer Klasse zu Texten. Wenn ein Text eine solche angereicherte relevante Signatur enthält, ist sichergestellt, daß er sowohl ein relevantes Schlüsselwort als auch relevante Information im Kontext des Schlüsselwortes enthält. Der Algorithmus bleibt derselbe wie vorher, die Ergebnisse haben eine höhere Precision, aber einen niedrigeren Recall.

- Ein dritter, *fallbasierter Textklassifikationsalgorithmus* versucht auch solche Texte richtig einzuordnen, in denen keine aussagekräftigen Schlüsselwörter existieren und die Bedeutung erst aus dem Kontext klar wird. Dazu benutzt man Fälle, die den

Kontext eines einzelnen Satzes durch fünf Slots beschreiben: Signaturen, Täter, Opfer, Ziele und Waffen. In der Trainingsphase werden alle Fälle mit der Klassifikation des zugehörigen Textes in eine Fallbasis eingetragen. Da nicht festgelegt werden kann, welche Fälle relevant sind, dienen Relevanz-Indizes der Verbindung zwischen Fällen. Wenn viele der Fälle mit gemeinsamen Indizes in relevanten Dokumenten vorhanden sind, werden die Indizes in der Testphase benutzt, um zwischen relevanten und irrelevanten Fällen zu unterscheiden.

Eine genaue Beschreibung der Algorithmen und viele Testergebnisse finden sich in [Riloff 94]. Der fallbasierte Algorithmus liefert die beste Kombination von Recall und Precision (z. B. 100% Precision und 61% Recall), er ist aber auch der umfangreichste Algorithmus.

Das CIRCUS-System liefert sehr gute Werte und basiert auf einfachen Algorithmen. Die Algorithmen sind bereichsunabhängig und leicht zu portieren. Allerdings wird zur Erzeugung der Konzeptknoten ein bereichsspezifisches Wörterbuch benötigt, dessen Erzeugung recht arbeitsintensiv ist. Da der Text zur Erzeugung der Konzeptknoten nur geskimmt wird, umgeht man die Probleme des tiefen Textparsens.

5 Probabilistische konzeptbasierte Verfahren

5.1 Die PCIR-Architektur

In der PCIR-Architektur *(Probabilistic Concept-based Information Retrieval)* [Fung 90a, Fung 90b] benutzt man zur Beantwortung der Anfrage ein probabilistisches Netzwerk, wendet darauf Inferenzregeln an und propagiert Wahrscheinlichkeiten durch das Netz. Sie integriert Techniken und Konzepte aus dem Bereich der probabilistischen Netzwerke (Repräsentation der Wahrscheinlichkeitsverteilung und probabilistisches Schließen), der Künstlichen Intelligenz (Techniken zur heuristischen Suche) und der Statistik (χ^2-Test zur Feststellung der Unabhängigkeit).

Anfragen sind eingeschränkt auf Konzepte, d. h. die Formulierung der Anfrage geschieht mittels vorher festgelegter Begriffe. So modelliert das PCIR-System beispielsweise das Konzept 'Terrorismus', was man sich als Wurzel eines Baumes mit anderen Konzepten, etwa 'Mord', 'Kidnapping' oder 'Opfer' vorstellen kann. Alle in diesem Baum enthaltenen Konzepte sind festgelegt und können abgefragt werden.

Das probabilistische Netzwerk, auch Wissensbasis genannt, enthält diese Konzepte sowie gewichtete Verbindungen zwischen einzelnen Konzepten, die Wahrscheinlichkeiten darstellen. Diese Wahrscheinlichkeiten geben an, wie weit das Vorhandensein eines Konzeptes durch das Vorhandensein von Unterkonzepten determiniert wird. Zunächst muß ein solches Netzwerk erstellt werden, wofür im PCIR-System die Wissensakquisitionskomponente CONSTRUCTOR verantwortlich ist. Benötigt wird eine Anzahl von repräsentativen Dokumenten, d. h. in diesem speziellen Fall enthalten Teile von ihnen das Konzept Terrorismus. Für jedes dieser Dokumente muß man manuell bestimmen, welche Konzepte, Unterkonzepte und Eigenschaften es enthält. Gegenwärtig bestehen die Eigenschaften im Auftreten oder in der Abwesenheit von Schlüsselwörtern, daher sind ihre Werte binär.

So sind z. B. die Wörter "explosiv" oder "explodieren" Eigenschaften für das Konzept Explosion.

Eingabe für CONSTRUCTOR sind folglich Dokumente mit Angaben zu Konzepten und Eigenschaften. Daraus wird ein Netzwerk mit gewichteten Verbindungen zwischen Konzepten und zwischen Konzepten und Eigenschaften erstellt. Eigenschaften werden dann mit Konzepten verbunden, wenn sie Evidenzen für das Konzept sind. Wie sehr ein Konzept von einer Eigenschaft abhängt, drückt das Gewicht der Verbindung aus. Die gleiche Bedeutung haben Verbindungen zwischen Konzepten.

Ist also die Wissensbasis erstellt, können entsprechend formulierte Anfragen beantwortet werden. Dazu muß jedes Dokument[8] auf Vorhandensein des Konzepts untersucht werden. Zurückgeliefert wird die Wahrscheinlichkeit für das Vorhandensein des Konzeptes. Durch einen Vergleich mit einem vorher vom Benutzer definierten Schwellwert wird entschieden, ob das Dokument ausgegeben wird oder nicht.

Die Berechnung der Wahrscheinlichkeit geschieht folgendermaßen: Aus dem Dokument extrahiert man erst die Eigenschaften. Da diese ebenfalls in der Wissensbasis als Knoten des Netzwerks enthalten sind, kann durch Benutzung der Werte der Eigenschaften ein Inferenzprozeß im Netzwerk gestartet werden. Dieser Prozeß ist beendet, wenn die Wahrscheinlichkeit für das gesuchte Konzept berechnet wurde. Wie der Inferenzprozeß genau abläuft, ist abhängig vom zugrundeliegenden Unsicherheitskalkül. Im PCIR-System wird der verteilte Algorithmus aus [Chang 89] benutzt.

Vorteile des Systems sind:

- Die Informationen der Dokumente können durch ein Netzwerk von Konzepten und Relationen zwischen ihnen sinnvoll repräsentiert werden, was die Wissensakquisition erleichtert.
- Konzepte reduzieren die Berechnungskomplexitäten beim Inferenzvorgang, da das Netzwerk i. a. nicht sehr dicht ist (wenige Kanten und viele Knoten). Im Vergleich zu einer vollständigen semantischen Analyse ist der Suchraum wesentlich verkleinert.
- Verarbeitung unsicherer Daten
- Testergebnisse liefern gute Recall- und Precision-Werte (70-80%) im Verhältnis zur Einfachheit des Systems.

Ein Nachteil des Systems ist die Beschränkung der Anfragen auf Konzepte. Weitaus problematischer jedoch ist die Tatsache, daß Konzepte, Eigenschaften und das Vorhandensein der Konzepte in den Dokumenten, die man zum Aufbau des Netzwerkes benutzt, manuell angegeben werden müssen.

[8] Diese Dokumente müssen nicht identisch mit denen sein, die zur Eingabe für CONSTRUCTOR benutzt wurden

6 Konnektionistische Verfahren

6.1 Das Match-Plus-System

Mit Hilfe des Kontext-Vektoren-Ansatzes [Gallant 92] soll versucht werden, zu einer Anfrage auch die Dokumente zu finden, die Synonyme von Termen der Anfrage enthalten. Ziel ist, die Vorteile eines Thesaurus ohne seine Nachteile (Bereichsabhängigkeit und manueller Aufbau) nutzen zu können.

Terme und darauf aufbauend auch Dokumente und Anfragen bildet man bei diesem Ansatz als Punkte in einen hochdimensionalen Vektorraum ($\approx$ 300) ab. Zu einer Anfrage werden dann die Dokumente ausgegeben, die im Vektorraum die nächsten Nachbarn der Anfrage, d. h. die Nachbarn mit dem kleinsten euklidischen Abstand, sind. Nach einer Normalisierung der Kontextvektoren entsprechen die nächsten Nachbarn den Vektoren, die bei Vektormultiplikation mit der Anfrage das größte Vektorprodukt liefern.

Um Kontextvektoren aufzubauen, wird zunächst eine Eigenschaftsmenge von n Termen spezifiziert, die geeignet ist, um Terme voneinander unterscheiden zu können. Die Eigenschaftsmenge kann etwa aus hochfrequenten Termen nach Entfernung von Stoppwörtern bestehen.

Für jeden Wortstamm k, der in mindestens einem Dokument enthalten ist, wird nun der Kontextvektor V_k aufgebaut.

- Die j-te Komponente ist stark positiv ($V_k^j \gg 0$), wenn Wort k eine ähnliche Bedeutung hat wie Term j aus der Eigenschaftsmenge.
- Sie ist 0 ($V_k^j = 0$), wenn Wort k keine Verbindung zu Term j hat.
- Sie ist stark negativ ($V_k^j \ll 0$) , wenn die Bedeutung des Wortes k konträr ist zu Term j.

Diese Interpretation entspricht der Interpretation der Gewichte von Verbindungen bei neuronalen Netzen.

Gewichte werden nach der *bootstrap*-Methode zugewiesen, wobei man für eine kleine Menge von Wortstämmen die Verbindungen von Hand einträgt. Dann werden den Kontextvektoren neuer Stämme Werte in Abhängigkeit zur Wortposition, d. h. Werte ähnlich denen ihrer Nachbarstämme im Text, zugewiesen. Sind die Kontextvektoren für Wortstämme ermittelt, berechnet man Kontextvektoren für Dokumente. Für alle Terme werden Gewichte unter Verwendung der inversen Dokumenthäufigkeit ermittelt. Anschließend bildet man die gewichtete Summe der Kontextvektoren für alle Stämme des Dokuments und normalisiert diese. Die Kontextvektoren für die Anfrage werden ähnlich berechnet mit dem Unterschied, daß die Gewichte anders definiert sind. Entweder wird eine Defaulteinstellung von 1.0 benutzt oder der Benutzer gibt selbst Werte an.

Auch dieses Verfahren kann man durch Relevanz-Feedback verbessern, indem die Dokument-Kontextvektoren der vom Benutzer als relevant gekennzeichneten Dokumente mit Gewichten zur Anfrage hinzugefügt werden. Vorteile dieses Verfahrens sind:

- Über die Anzahl der ausgegebenen Dokumente entscheidet der Benutzer und die Entfernung vom Kontextvektor der Anfrage liefert ein Maß für die Qualität der Dokumente.
- Die Ähnlichkeit von Vektoren ist einfach zu bestimmen. Je höher das Vektorprodukt, desto ähnlicher sind die Vektoren.
- Schlüsselwortverfahren können leicht mit Kontextvektoren verbunden werden. Dabei werden die Schlüsselworte als Filter für die Dokumente benutzt und die Reihenfolge der Ausgabe entscheidet sich durch den Kontext-Vektoren Ansatz.
- Die Idee ist einfach zu implementieren.

In ersten Tests mit dem Match-Plus System, das diesen einfachen Ansatz realisiert, wurden Werte vergleichbar mit dem SMART-System erreicht. Fraglich ist, wie weit die bootstrap-Methode brauchbare Gewichte liefert, da der Zeitaufwand für eine manuelle Einstellung aller Gewichte sehr hoch ist. Treten als Terme Wörter mit Mehrdeutigkeiten auf, können überhaupt keine sinnvollen Gewichte zugewiesen werden, deshalb müssen mehrdeutige Wörter aus der Eigenschaftsmenge ausgeschlossen werden.

7 XPS

Bei der Benutzung von Expertensystemen in IR-Systemen stellt sich die Frage, an welcher Stelle das XPS eingesetzt wird und welches der beiden Systeme die Kontrolle hat. Soll das XPS in das IR-System integriert werden, müssen Expertenwissen, Heuristiken und Inferenzen in eine prozedurale Form umgewandelt werden, was natürlich einen Verlust an Flexibilität bedeutet.

7.1 IOTA

Im IOTA-System [Chiaramella 87] steuert das XPS das IR-System. Dazu müssen Prozeduraufrufe (z. B. 'Anfrage parsen') in Regeln verpackt werden, damit das XPS sie verarbeiten kann. Das IR-System ist nur noch dafür zuständig, die Antworten zu den Fragen zu finden. Die Anfragen werden dazu in eine Boolesche Verknüpfung von Schlüsselwörtern umgewandelt. An Wissen stehen eine Wissensbasis mit allgemeinem Weltwissen in Form von Produktionsregeln, zwei Wörterbücher mit Flexionen von Wörtern und ein Thesaurus bereit. Das IOTA-System bietet grundsätzlich keine neuen Ideen, nur Verbesserungen und Komfort. Es erkennt einen Text vollständig, braucht aber dafür auch viel Zeit.

7.2 I^3R

Das I^3R System [Chiaramella 87] besteht aus einem Interface-Manager und mehreren Expertensystemen, die zentral gesteuert werden. Jedes Expertensystem ist für einen speziellen Teil der Anfragenverarbeitung zuständig und teilt mit den anderen eine gemeinsame Datenstruktur, eine Blackboard. Das System benutzt eine Wissensbasis mit Dokumenten

und ihrer semantischen Repräsentation, benutzerspezifischem Wissen und Bereichswissen, das in einer frameähnlichen Struktur abgespeichert ist.

Das System hat Experten für:

- die Umwandlung der Anfrage in eine interne Repräsentation;
- das Indexieren der Dokumente und Anfragen;
- Suchstrategien;
- das Auffinden von Synonymen (Thesaurus);
- das Blättern in der Wissensbasis.

7.3 CODER

Noch mehr Experten hat das CODER-System (Composite Document Expert - Extended, Effective Retrieval) [Fox 87], das zum Testen von KI-Methoden im Information Retrieval entwickelt wurde. Auch hier greifen die Experten auf eine gemeinsame Blackboard-Struktur zu. Es existieren Experten für mehrere Wissensrepräsentationsschemata, für die Verwendung einer Zeitlogik, für Benutzermodellierung, einfaches Parsing und Suchheuristiken.

Beim Einsatz von Expertensystemen liegt die Hauptaufgabe dieser Systeme in der Sprachverarbeitung, gewissermaßen kann man sie also als Erweiterung zu Systemen mit NLP betrachten. Allerdings sind die Methoden zur Anfragenbeantwortung (z. B. Indexieren) teilweise klassisch, so daß diese eigentlich nur in ein modernes Gerüst verpackt sind.

8 Bewertung und Vergleich

Die beschriebenen Systeme unterscheiden sich hinsichtlich vieler Kriterien, einige sollen beispielhaft erläutert werden:

1. Genauigkeit: Die Genauigkeit der Systeme wird in Recall und Precision gemessen. Grundsätzlich liefern klassifizierende Systeme bessere Werte, da die Klassifikationsaufgabe i. a. weniger komplex als eine natürlichsprachliche Informationsextraktion aus Texten ist. Ansonsten schneiden die konventionellen Systeme am schlechtesten ab, da sie wortbasiert arbeiten. Beinhalten die Systeme einen Parser, wird die semantische Bedeutung sehr gut erfaßt, entsprechend hoch ist auch die Genauigkeit. Die regelbasierten Verfahren liefern mittlere Werte, da sie Teile der Kontexte (z. B. Redewendungen) in Dokumenten durch Regeln verarbeiten.

2. Robustheit: Hierin liegt ein Vorteil der statistischen Verfahren, denen fehlerhafte Eingaben wenig ausmachen. Je mehr NLP in das System eingebracht wird, desto anfälliger wird es. Daher verwenden einige der vorgestellten Systeme einen Textskimmer, der schneller und robuster ist als ein Parser.

3. Systemkomplexität: Die Systeme wurden in diesem Überblick in weiten Teilen nach ihrer Komplexität geordnet. Eine Ausnahme bildet das Match-Plus-System, das einen einfachen Ansatz realisiert. Die Expertensysteme sind die mächtigsten Systeme, sie enthalten oft umfangreiche Sprachverarbeitung, viele Wörterbücher und eine gute Benutzeroberfläche.

4. Portabilität: Auch hier bieten die statistischen Verfahren einen Vorteil, da sie aufgrund ihrer rein mathematischen Fundierung bereichsunabhängig sind. TCS wurde als Shell entwickelt und ist daher portabel, allerdings muß die Regelbasis für jeden Bereich gefüllt werden. Die konzeptbasierten Verfahren enthalten vordefinierte Konzepte, die auf den speziellen Bereich zugeschnitten sind und daher auch ausgetauscht werden müssen.

5. Laufzeit: FRUMP und SCISOR wurden an Netzwerke angeschlossen, so daß sie unter Echtzeitbedingungen arbeiten. Über die anderen komplexeren Systeme liegen in der Literatur keine Daten vor. Es kann aber davon ausgegangen werden, daß sich die Laufzeit durch Einsatz eines Parsers beträchtlich erhöht.

Alle obigen Systeme haben das gleiche Problem. Je genauer das System arbeiten soll, desto besser muß es den Kontext der Dokumente erfassen. Eine höhere Genauigkeit geht meist zu Lasten der Geschwindigkeit, der Bereichsunabhängigkeit, der Komplexität und der Robustheit. Insofern unterscheiden sich die Systeme hinsichtlich der Mächtigkeit eingebrachter NLP-Techniken. Es hängt von der jeweiligen Anwendung und den Bedürfnissen des Benutzers ab, wie genau und dadurch auch wie langsam und komplex ein IR-System sein soll.

9 Übertragbarkeit der Verfahren auf die Dokumentanalyse

Nach der Beschreibung und Bewertung von Methoden des Information Retrieval gilt es nun, diese Kenntnisse für ein Textanalysesystem auszunutzen. Daher wird sequentiell auf die vorgestellten Verfahren im Hinblick auf den möglichen Einsatz in der Dokumentanalyse eingegangen.

- Der klassische IR-Ansatz ist sehr gut erforscht und besitzt eine sichere, mathemathische Fundierung. Daher sind auf dieser Idee basierende Systeme i. a. sehr robust und erreichen auch gute Werte. Jedoch bleibt kontextuelles Wissen unberücksichtigt.

- Das LSI-Verfahren berücksichtigt nur die Wörter des Textes, die möglicherweise Synonymbeziehungen zwischen Dokumenten widerspiegeln und vernachlässigt daher Funktionswörter. Auch hier kann nur von einer bedingten Verwertung kontextueller Information gesprochen werden, denn für Geschäftsbriefe[9] typische Redewendungen werden nicht erfaßt. Hinzu kommt die Frage, ob die ausgebildeten Ähnlichkeiten

[9] Die Domäne der Geschäftsbriefe wird im ALV-Projekt exemplarisch zur Dokumentanalyse herangezogen.

aufgrund der im ALV-System vorliegenden kleinen Testmenge wirklich groß genug für eine gute Klassifikation sind. Aus den vorliegenden Artikeln zu diesem Thema war überdies nicht ersichtlich, wie komplex die Eigenvektorzerlegung ist, d. h. ob die mathematische Basis einfach realisiert werden kann.

- Der erweiterte klassische Ansatz ist ideal, um Kontexte beliebiger Größe zu betrachten. Leider ist nicht ersichtlich, ob die Größe der Kontexte manuell angegeben werden muß oder ob sie berechnet werden kann. Eine genauere Untersuchung dieses Ansatzes wäre sicherlich interessant.

- Die regelbasierten Verfahren bieten die Möglichkeit, eigene Pattern zu definieren, wodurch Kontexte ausgedrückt werden. Durch die Einführung von Konzepten kann man lokale Zusammenhänge erfassen. Übertragen auf die Klassifikation von Geschäftsbriefen bedeutet dies, daß Klassen oder Teilklassen Konzepte gemein haben können und dadurch trotzdem von den übrigen Klassen abgegrenzt werden.

- Die konzeptbasierten Verfahren sind im Entwurf und in der Implementation durch den Einsatz eines Skimmers oder Parsers sehr komplex. Sie liefern die genauesten Informationen über den Text, aber für eine Klassifikationsaufgabe ist es nicht notwendig, den Text allzu tief zu analysieren.

- Die Idee des Match-Plus-Verfahrens bezieht ebenso wie LSI Kontexte nur bedingt ein. Außerdem ist es fraglich, ob die *bootstrap*-Methode wirklich praktikabel ist. Wahrscheinlich müßte daher die Gewichtung doch manuell erfolgen, was für jeden Brief erneut einen hohen Aufwand bedeuten würde.

Das erste im ALV-Projekt realisierte Textklassifikationssystem INFOCLAS [Dittrich 92] setzt die klassischen, statistischen Methoden aus 2.1.2 ein, um Deskriptoren (wichtige Terme) eines Textes zu erhalten, die dann die Basis für eine Klassifikation bilden. Die Erfahrungen, die beim Einsatz dieses Systems gewonnen werden konnten, dienten gemeinsam mit den Ergebnissen dieser Arbeit als Entscheidungsgrundlage für die Konzeption eines weiteren inhaltsbasierten Textklassifikationssystems.

Nach eingehender Betrachtung verbleiben zum Einsatz in der Textanalyse lediglich zwei der obigen Methoden. Der Ansatz mit globalen und lokalen Ähnlichkeiten und der regelbasierte Ansatz scheinen realisierbar. Die endgültige Entscheidung für einen regelbasierten Ansatz kann so motiviert werden:

- Die Wahrscheinlichkeiten der Texterkenner können als Glaubwürdigkeiten für Regeln interpretiert und durch ein Regelnetzwerk propagiert werden.

- In Regeln können verschiedenartige Bedingungen verpackt werden, beispielsweise kann für die Klassifikation auch die Brieflänge und die Anzahl der Textblöcke herangezogen werden.

- Das System kann als Shell konzipiert werden, so daß es leicht auf andere Domänen übertragbar ist.

Das daraus resultierende RULECLAS-System [Wenzel 94] hat sich insbesondere durch die Verknüpfung von Regeln und Konzepten bewährt und gezeigt, wie flexibel regelbasierte Systeme durch eine leicht zu modifizierende Regelbasis sein können.

Literatur

[Chang 89] K. C. Chang and R. M. Fung. *Node Aggregation for Distributed Inference in Bayesian Networks.* Proceedings of the Eleventh IJCAI, Detroit, Michigan, August 1989, pp. 265–270

[Chiaramella 87] Y. Chiaramella and B. Defude. *A Prototype of an Intelligent System for Information Retrieval: IOTA.* Information Processing and Management, vol. 23, no. 4, 1987, pp. 285–303

[DeJong 82] G. DeJong. *An Overview of the FRUMP Sytem.* In W. G. Lehnert, M. H. Ringle. Strategies for Natural Language Processing. Lawrence Erlbaum Associates, Hillsdale, 1982, pp. 149–175

[Dittrich 92] S. Dittrich und R. Hoch. *Automatische, Deskriptor-basierte Unterstützung der Dokumentanalyse zur Fokussierung und Klassifizierung von Geschäftsbriefen.* DFKI Document, D-92-19, Juli 1992

[Fox 87] Edward A. Fox. *Development of the CODER System: A Testbed for AI Methods in Information Retrieval.* Information Processing and Management, vol. 23, no. 4, 1987, pp. 341–366

[Fung 90a] R. M. Fung, S. L. Crawford, L. A. Appelbaum and R. M. Tong. *An Architecture for Probabilistic Concept-Based Information Retrieval.* ACM SIGIR 1990, Proceedings of the 13th International Conference on Research and Development in Information Retrieval, Brussels, Belgium, September 1990, pp. 455–467

[Fung 90b] R. M. Fung and S. L. Crawford. *Constructor: A System for the Induction of Probabilistic Models.* AAAI-90, Proceedings of the Eight International Conference on Artificial Intelligence, Boston, Massachusetts, June 1990, pp. 762–769

[Gallant 92] Stephen I. Gallant. *HNCs MatchPlus System.* SIGIR-Forum 1992, pp. 34–38

[Hayes 88] P. J. Hayes, L. E. Knecht and M. J. Cellio. *A News Story Categorization System.* Proceedings of the second Conference of Applied Natural Language Processing, Association for Computational Linguistics, Austin, Texas, February 1988, pp. 9–17

[Hayes 90] P. J. Hayes, P. M. Andersen, I. B. Nirenburg and L. M. Schmandt. *TCS: A Shell for Content-Based Text Categorization.* Proceedings of the Sixth Conference on AI Applications, Santa Barbara, California, March 1990, pp. 320–326

[Hayes 92] P. J. Hayes. *Intelligent High-Volume Text Processing Using Shallow, Domain-Specific Techniques.* In Paul S. Jacobs, Text-Based Intelligent Systems, Lawrence Erlbaum Associates, Inc., Hillsdale, 1992, pp. 227–243

[Lehnert 91] W. Lehnert, C. Cardie, D. Fisher, E. Riloff and R. Williams. *University of Massachussets: Description of the CIRCUS System as Used for MUC-3.* Proceedings of the Third Message Understanding Conference, MUC-3, San Diego, California, May 1991, pp. 223–234

[Lochbaum 87] K. E. Lochbaum and L. A. Streeter. *Comparing and Combining the Effectiveness of Latent Semantic Indexing and the Ordinary Vector Space Model for Information Retrieval.* Information Processing and Management, vol. 25, no. 6, 1987, pp. 665–676

[Mauldin 87] M. L. Mauldin, J. Carbonell and R. Thomason. *Beyond the Keyword Barrier: Knowledge-Based Information Retrieval.* Information Services and Use, vol. 7, 1987, pp. 103–117

[Mauldin 91] Michael L. Mauldin. *Retrieval Performance in FERRET.* ACM SIGIR 1991, 14th International Conference on Research and Development in Information Retrieval, pp. 347–355

[Rau 88] L. F. Rau and P. S. Jacobs. *Integrating Top-Down and Bottom-Up Strategies in a Text Processing System.* Second Conference on Applied NLP, Austin, Texas, February 1988, pp. 129–135

[Rau 90] L. F. Rau and P. S. Jacobs. *SCISOR: Extracting Information from On-Line News.* Communications of the ACM, vol. 33, no. 11, November 1990, pp. 88–97

[Riloff 94] E. Riloff and W. Lehnert. *Information Extraction as a Basis for High-Precision Text Classification.* Transactions on Office Information Systems (TOIS), 1994, to appear

[Salton 71] Gerard Salton. *The SMART Retrieval System.* Prentice-Hall, 1971

[Salton 83] Gerard Salton. *Introduction to Modern Information Retrieval.* McGraw-Hill Computer Science Series, McGraw-Hill, Inc., 1983

[Salton 89] Gerard Salton. *Automated Text Processing.* Addison-Wesley, 1989

[Salton 91] G. Salton and C. Buckley. *Global Text Matching for Information Retrieval.* Science, vol. 253, August 1991, pp. 1012–1015

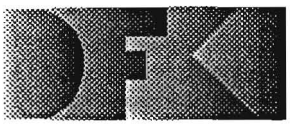
[Salton 92] G. Salton and C. Buckley. *Automatic Text Structuring Experiments.* In Paul S. Jacobs, Text-Based Intelligent Systems, Lawrence Erlbaum Associates, Inc., Hillsdale, 1992, pp. 199–211

[Tong 85] R. M. Tong and D. G. Shapiro. *Experimental Investigations of Uncertainty in a Rule-Based System for Information Retrieval.* International J. Man-Machine Studies, vol. 22, 1985, pp. 265–282

[Tong 89] R. M. Tong, L. A. Appelbaum and V. N. Askman. *A Knowledge Representation for Conceptual Information Retrieval.* International Journal of Intelligent Systems, vol. 4, 1989, pp. 259–283

[Tong 91] R. M. Tong and L. Blumer Balcom. *Advanced Decision Systems: Description of the CODEX Sytem as Used for MUC-3.* Proceedings of the Third Message Understanding Conference, MUC-3, San Diego, California, May 1991, pp. 129–137

[Wenzel 94] Claudia Wenzel. *RULECLAS - Regelbasierte Textklassifikation in der Dokumentanalyse unter Einbeziehung kontextueller Information.* Diplomarbeit DFKI Kaiserslautern, 1994



Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

DFKI
-Bibliothek-
PF 2080
67608 Kaiserslautern
FRG

DFKI Publikationen

Die folgenden DFKI Veröffentlichungen sowie die aktuelle Liste von allen bisher erschienenen Publikationen können von der oben angegebenen Adresse oder per anonymem ftp von ftp.dfki.uni-kl.de (131.246.241.100) unter pub/Publications bezogen werden.

Die Berichte werden, wenn nicht anders gekennzeichnet, kostenlos abgegeben.

DFKI Publications

The following DFKI publications or the list of all published papers so far are obtainable from the above address or via anonymous ftp from ftp.dfki.uni-kl.de (131.246.241.100) under pub/Publications.

The reports are distributed free of charge except if otherwise indicated.

DFKI Research Reports

RR-93-10
Martin Buchheit, Francesco M. Donini, Andrea Schaerf: Decidable Reasoning in Terminological Knowledge Representation Systems
35 pages

RR-93-11
Bernhard Nebel, Hans-Jürgen Bürckert:
Reasoning about Temporal Relations:
A Maximal Tractable Subclass of Allen's Interval Algebra
28 pages

RR-93-12
Pierre Sablayrolles: A Two-Level Semantics for French Expressions of Motion
51 pages

RR-93-13
Franz Baader, Karl Schlechta:
A Semantics for Open Normal Defaults via a Modified Preferential Approach
25 pages

RR-93-14
Joachim Niehren, Andreas Podelski, Ralf Treinen:
Equational and Membership Constraints for Infinite Trees
33 pages

RR-93-15
Frank Berger, Thomas Fehrle, Kristof Klöckner, Volker Schölles, Markus A. Thies, Wolfgang Wahlster: PLUS - Plan-based User Support
Final Project Report
33 pages

RR-93-16
Gert Smolka, Martin Henz, Jörg Würtz: Object-Oriented Concurrent Constraint Programming in Oz
17 pages

RR-93-17
Rolf Backofen:
Regular Path Expressions in Feature Logic
37 pages

RR-93-18
Klaus Schild: Terminological Cycles and the Propositional μ-Calculus
32 pages

RR-93-20
Franz Baader, Bernhard Hollunder:
Embedding Defaults into Terminological Knowledge Representation Formalisms
34 pages

RR-93-22
Manfred Meyer, Jörg Müller:
Weak Looking-Ahead and its Application in Computer-Aided Process Planning
17 pages

RR-93-23
Andreas Dengel, Ottmar Lutzy:
Comparative Study of Connectionist Simulators
20 pages

RR-93-24
Rainer Hoch, Andreas Dengel:
Document Highlighting —
Message Classification in Printed Business Letters
17 pages

RR-93-25
Klaus Fischer, Norbert Kuhn: A DAI Approach to Modeling the Transportation Domain
93 pages

RR-93-26
Jörg P. Müller, Markus Pischel: The Agent Architecture InteRRaP: Concept and Application
99 pages

RR-93-27
Hans-Ulrich Krieger:
Derivation Without Lexical Rules
33 pages

RR-93-28
Hans-Ulrich Krieger, John Nerbonne, Hannes Pirker: Feature-Based Allomorphy
8 pages

RR-93-29
Armin Laux: Representing Belief in Multi-Agent Worlds viaTerminological Logics
35 pages

RR-93-30
Stephen P. Spackman, Elizabeth A. Hinkelman:
Corporate Agents
14 pages

RR-93-31
Elizabeth A. Hinkelman, Stephen P. Spackman:
Abductive Speech Act Recognition, Corporate Agents and the COSMA System
34 pages

RR-93-32
David R. Traum, Elizabeth A. Hinkelman:
Conversation Acts in Task-Oriented Spoken Dialogue
28 pages

RR-93-33
Bernhard Nebel, Jana Koehler:
Plan Reuse versus Plan Generation: A Theoretical and Empirical Analysis
33 pages

RR-93-34
Wolfgang Wahlster:
Verbmobil Translation of Face-To-Face Dialogs
10 pages

RR-93-35
Harold Boley, François Bry, Ulrich Geske (Eds.):
Neuere Entwicklungen der deklarativen KI-Programmierung — *Proceedings*
150 Seiten
Note: This document is available only for a nominal charge of 25 DM (or 15 US-$).

RR-93-36
Michael M. Richter, Bernd Bachmann, Ansgar Bernardi, Christoph Klauck, Ralf Legleitner, Gabriele Schmidt: Von IDA bis IMCOD: Expertensysteme im CIM-Umfeld
13 Seiten

RR-93-38
Stephan Baumann: Document Recognition of Printed Scores and Transformation into MIDI
24 pages

RR-93-40
Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, Werner Nutt, Andrea Schaerf:
Queries, Rules and Definitions as Epistemic Statements in Concept Languages
23 pages

RR-93-41
Winfried H. Graf: LAYLAB: A Constraint-Based Layout Manager for Multimedia Presentations
9 pages

RR-93-42
Hubert Comon, Ralf Treinen:
The First-Order Theory of Lexicographic Path Orderings is Undecidable
9 pages

RR-93-43
M. Bauer, G. Paul: Logic-based Plan Recognition for Intelligent Help Systems
15 pages

RR-93-44
Martin Buchheit, Manfred A. Jeusfeld, Werner Nutt, Martin Staudt: Subsumption between Queries to Object-Oriented Databases
36 pages

RR-93-45
Rainer Hoch: On Virtual Partitioning of Large Dictionaries for Contextual Post-Processing to Improve Character Recognition
21 pages

RR-93-46
Philipp Hanschke: A Declarative Integration of Terminological, Constraint-based, Data-driven, and Goal-directed Reasoning
81 pages

RR-93-48
Franz Baader, Martin Buchheit, Bernhard Hollunder:
Cardinality Restrictions on Concepts
20 pages

RR-94-01
Elisabeth André, Thomas Rist:
Multimedia Presentations:
The Support of Passive and Active Viewing
15 pages

RR-94-02
Elisabeth André, Thomas Rist:
Von Textgeneratoren zu Intellimedia-Präsentationssystemen
22 Seiten

RR-94-03
Gert Smolka:
A Calculus for Higher-Order Concurrent Constraint Programming with Deep Guards
34 pages

RR-94-05
Franz Schmalhofer, J.Stuart Aitken, Lyle E. Bourne jr.:
Beyond the Knowledge Level: Descriptions of Rational Behavior for Sharing and Reuse
81 pages

RR-94-06
Dietmar Dengler:
An Adaptive Deductive Planning System
17 pages

RR-94-07
Harold Boley: Finite Domains and Exclusions as First-Class Citizens
25 pages

RR-94-08
Otto Kühn, Björn Höfling: Conserving Corporate Knowledge for Crankshaft Design
17 pages

RR-94-10
Knut Hinkelmann, Helge Hintze:
Computing Cost Estimates for Proof Strategies
22 pages

RR-94-11
Knut Hinkelmann: A Consequence Finding Approach for Feature Recognition in CAPP
18 pages

RR-94-12
Hubert Comon, Ralf Treinen:
Ordering Constraints on Trees
34 pages

RR-94-13
Jana Koehler: Planning from Second Principles —A Logic-based Approach
49 pages

RR-94-14
Harold Boley, Ulrich Buhrmann, Christof Kremer:
Towards a Sharable Knowledge Base on Recyclable Plastics
14 pages

RR-94-15
Winfried H. Graf, Stefan Neurohr: Using Graphical Style and Visibility Constraints for a Meaningful Layout in Visual Programming Interfaces
20 pages

RR-94-16
Gert Smolka: A Foundation for Higher-order Concurrent Constraint Programming
26 pages

RR-94-17
Georg Struth:
Philosophical Logics—A Survey and a Bibliography
56 pages

DFKI Technical Memos

TM-92-04
Jürgen Müller, Jörg Müller, Markus Pischel, Ralf Scheidhauer:
On the Representation of Temporal Knowledge
61 pages

TM-92-05
Franz Schmalhofer, Christoph Globig, Jörg Thoben:
The refitting of plans by a human expert
10 pages

TM-92-06
Otto Kühn, Franz Schmalhofer: Hierarchical skeletal plan refinement: Task- and inference structures
14 pages

TM-92-08
Anne Kilger: Realization of Tree Adjoining Grammars with Unification
27 pages

TM-93-01
Otto Kühn, Andreas Birk: Reconstructive Integrated Explanation of Lathe Production Plans
20 pages

TM-93-02
Pierre Sablayrolles, Achim Schupeta:
Conlfict Resolving Negotiation for COoperative Schedule Management
21 pages

TM-93-03
Harold Boley, Ulrich Buhrmann, Christof Kremer:
Konzeption einer deklarativen Wissensbasis über recyclingrelevante Materialien
11 pages

TM-93-04
Hans-Günther Hein:
Propagation Techniques in WAM-based Architectures — The FIDO-III Approach
105 pages

TM-93-05
Michael Sintek: Indexing PROLOG Procedures into DAGs by Heuristic Classification
64 pages

TM-94-01
Rainer Bleisinger, Klaus-Peter Gores:
Text Skimming as a Part in Paper Document Understanding
14 pages

TM-94-02
Rainer Bleisinger, Berthold Kröll:
Representation of Non-Convex Time Intervals and Propagation of Non-Convex Relations
11 pages

DFKI Documents

D-93-09
Hans-Ulrich Krieger, Ulrich Schäfer:
TDL ExtraLight User's Guide
35 pages

D-93-10
Elizabeth Hinkelman, Markus Vonerden, Christoph Jung: Natural Language Software Registry (Second Edition)
174 pages

D-93-11
Knut Hinkelmann, Armin Laux (Eds.):
DFKI Workshop on Knowledge Representation Techniques — Proceedings
88 pages

D-93-12
Harold Boley, Klaus Elsbernd, Michael Herfert, Michael Sintek, Werner Stein:
RELFUN Guide: Programming with Relations and Functions Made Easy
86 pages

D-93-14
Manfred Meyer (Ed.): Constraint Processing – Proceedings of the International Workshop at CSAM'93, July 20-21, 1993
264 pages
Note: This document is available only for a nominal charge of 25 DM (or 15 US-$).

D-93-15
Robert Laux:
Untersuchung maschineller Lernverfahren und heuristischer Methoden im Hinblick auf deren Kombination zur Unterstützung eines Chart-Parsers
86 Seiten

D-93-16
Bernd Bachmann, Ansgar Bernardi, Christoph Klauck, Gabriele Schmidt: Design & KI
74 Seiten

D-93-20
Bernhard Herbig:
Eine homogene Implementierungsebene für einen hybriden Wissensrepräsentationsformalismus
97 Seiten

D-93-21
Dennis Drollinger:
Intelligentes Backtracking in Inferenzsystemen am Beispiel Terminologischer Logiken
53 Seiten

D-93-22
Andreas Abecker:
Implementierung graphischer Benutzungsoberflächen mit Tcl/Tk und Common Lisp
44 Seiten

D-93-24
Brigitte Krenn, Martin Volk:
DiTo-Datenbank: Datendokumentation zu Funktionsverbgefügen und Relativsätzen
66 Seiten

D-93-25
Hans-Jürgen Bürckert, Werner Nutt (Eds.):
Modeling Epistemic Propositions
118 pages
Note: This document is available only for a nominal charge of 25 DM (or 15 US-$).

D-93-26
Frank Peters: Unterstützung des Experten bei der Formalisierung von Textwissen
INFOCOM:
Eine interaktive Formalisierungskomponente
58 Seiten

D-93-27
Rolf Backofen, Hans-Ulrich Krieger, Stephen P. Spackman, Hans Uszkoreit (Eds.):
Report of theEAGLES Workshop on Implemented Formalisms at DFKI, Saarbrücken
110 pages

D-94-01
Josua Boon (Ed.):
DFKI-Publications: The First Four Years
1990 - 1993
75 pages

D-94-02
Markus Steffens: Wissenserhebung und Analyse zum Entwicklungsprozeß eines Druckbehälters aus Faserverbundstoff
90 pages

D-94-03
Franz Schmalhofer: Maschinelles Lernen: Eine kognitionswissenschaftliche Betrachtung
54 pages

D-94-04
Franz Schmalhofer, Ludger van Elst:
Entwicklung von Expertensystemen: Prototypen, Tiefenmodellierung und kooperative Wissensevolution
22 pages

D-94-06
Ulrich Buhrmann:
Erstellung einer deklarativen Wissensbasis über recyclingrelevante Materialien
117 pages

D-94-08
Harald Feibel: IGLOO 1.0 - Eine grafikunterstützte Beweisentwicklungsumgebung
58 Seiten

D-94-07
Claudia Wenzel, Rainer Hoch:
Eine Übersicht über Information Retrieval (IR) und NLP-Verfahren zur Klassifikation von Texten
25 Seiten

Eine Übersicht über Information Retrieval (IR) und NLP-Verfahren zur Klassifikation von Texten

Claudia Wenzel, Rainer Hoch

D-94-07
Document