Band 20 – Heft 1 – Jahrgang 2005 – ISSN 0175-1336
Zeitschrift für Computerlinguistik und Sprachtechnologie
GLDV-Journal for Computational Linguistics and Language Technology

LDV Forum

Themenschwerpunkt

Text Mining

Herausgegeben von
Alexander Mehler und Christian Wolff

Gesellschaft für linguistische Datenverarbeitung www.gldv.org

Text Mining

LDV/ Impressum

LDV-Forum Zeitschrift für Computerlinguistik und Sprachtechnologie
ISSN 0175-1336 GLDV-Journal for Computational Linguistics and Language
Band 20 - 2005 - Heft 1 Technology – Offizielles Organ der GLDV

Herausgeber Gesellschaft für Linguistische Datenverarbeitung e. V. (GLDV)

Juniorprofessor Dr. Alexander Mehler, Universität Bielefeld,
alexander.mehler@uni-bielefeld.de
Prof. Dr. Christian Wolff, Universität Regensburg
christian.wolff@sprachlit.uni-regensburg.de

Anschrift der Redaktion Prof. Dr. Christian Wolff,
Universität Regensburg
Institut für Medien-, Informations- und Kulturwissenschaft
D-93040 Regensburg

Wissenschaftlicher Beirat Vorstand, Beirat und Arbeitskreisleiter der GLDV
http://www.gldv.org/cms/vorstand.php,
http://www.gldv.org/cms/topics.php

Erscheinungsweise 2 Hefte im Jahr, halbjährlich zum 31. Mai und 31. Oktober. Preprints und redaktionelle Planungen sind über die Website der GLDV einsehbar (*http://www.gldv.org*).

Einreichung von Beiträgen Unaufgefordert eingesandte Fachbeiträge werden vor Veröffentlichung von mindestens zwei ReferentInnen begutachtet. Manuskripte sollten deshalb möglichst frühzeitig eingereicht werden und bei Annahme zur Veröffentlichung in jedem Fall elektronisch und zusätzlich auf Papier übermittelt werden. Die namentlich gezeichneten Beiträge geben ausschließlich die Meinung der AutorInnen wieder. Einreichungen sind an die Herausgeber zu übermitteln.

Bezugsbedingungen Für Mitglieder der GLDV ist der Bezugspreis des LDV-Forums im Jahresbeitrag mit eingeschlossen. Jahresabonnements können zum Preis von 25,- € (inkl. Versand), Einzelexemplare zum Preis von 15,- € (zzgl. Versandkosten) bei der Redaktion bestellt werden.

Satz und Druck Christoph Pfeiffer, Regensburg, mit *LaTeX* (*pdfeTeX* / *MiKTeX*) und *Adobe InDesign CS 3.0.1* , Druck: Druck TEAM KG, Regensburg

Christian Wolff und Alexander Mehler

Editorial

Liebe GLDV-Mitglieder, liebe Leserinnen und Leser des LDV-Forum,

nach langer, bis vor das Jahr 2003 zurückreichender Vorbereitungszeit können wir Ihnen nun Heft 1 des 20. Jahrgangs des LDV-Forums vorlegen. In bewährter Tradition handelt es sich dabei um ein Themenheft, dessen Beiträge dem Thema *Text Mining* gewidmet sind.

Die inhaltliche Abrundung und der Abschluß dieses Heftes ist nicht zuletzt der Tatsache geschuldet, dass mit Alexander Mehler ein weiterer Herausgeber für das LDV-Forum gefunden werden konnte. Die Herausgeber hoffen, dass mit der nun erreichten Zusammenstellung von Beiträgen eine fruchtbare und interessante Darstellung eines noch jungen Arbeitsgebiets erreicht werden konnte. Den Autorinnen und Autoren – und selbstverständlich in gleicher Weise auch den Leserinnen und Lesern – sei jedenfalls für die lange Geduld bis zum Erscheinen des Hefts sehr herzlich gedankt.

Der im vergangenen Jahr angekündigte Ausbau der Website der GLDV (*http://www.gldv.org*) zu einem computerlinguistischen und sprachtechnologischen Informationsportal ist mittlerweile vollzogen. Unter der Regie von Bernhard Schröder (Bonn) konnte ein Content Management-System erfolgreich für die Belange der GLDV adaptiert werden. Der Inhalt des LDV-Forum kann sich daher wie geplant auf Fachbeiträge konzentrieren.

Da – dem Gegenstand angemessen – die Beiträge dieses Hefts über erhebliche formale Anteile verfügen, war es nicht möglich, die für das Vorgängerheft entwickelte gestalterische und publikationstechnische Lösung vollständig beizubehalten. Wir haben uns aber bemüht, beim Satz der Beiträge in TEX eine behutsame Annäherung an die bisherige Gestaltung vorzunehmen. Dabei ist erneut Herrm cand. phil. Christoph Pfeiffer großer Dank geschuldet, der wieder den Satz übernommen hat und dabei seine umfangreichen TEX-Kentnisse erfolgreich einbringen konnte.

Mit Erscheinen dieses Heftes zum 31. Mai 2005 wird der reguläre Publikationstakt des LDV-Forum endlich wieder erreicht. Ihm wird im Herbst diesen Jahres als zweites Heft des 20. Bandes ein Themenheft mit dem Schwerpunkt Corpuslinguistik folgen.

Regensburg und Bielefeld, im Mai 2005

Christian Wolff und Alexander Mehler

LDV FORUM - Band 20(1) - 2005
Themenheft Text Mining

Alexander Mehler und Christian Wolff

Einleitung: Perspektiven und Positionen des Text Mining

1 Einleitung

Beiträge zum Thema *Text Mining* beginnen vielfach mit dem Hinweis auf die enorme Zunahme *online* verfügbarer Dokumente, ob nun im Internet oder in Intranets (Losiewicz et al. 2000; Merkl 2000; Feldman 2001; Mehler 2001; Joachims & Leopold 2002). Der hiermit einhergehenden „Informationsflut" wird das Ungenügen des *Information Retrieval* (IR) bzw. seiner gängigen Verfahren der Informationsaufbereitung und Informationserschließung gegenübergestellt. Es wird bemängelt, dass sich das IR weitgehend darin erschöpft, Teilmengen von Textkollektionen auf Suchanfragen hin aufzufinden und in der Regel bloß listenförmig anzuordnen.

Das auf diese Weise dargestellte Spannungsverhältnis von Informationsexplosion und Defiziten bestehender IR-Verfahren bildet den Hintergrund für die Entwicklung von Verfahren zur automatischen Verarbeitung textueller Einheiten, die sich stärker an den Anforderungen von Informationssuchenden orientieren. Anders ausgedrückt: Mit der Einführung der *Neuen Medien* wächst die Bedeutung digitalisierter Dokumente als Primärmedium für die Verarbeitung, Verbreitung und Verwaltung von Information in öffentlichen und betrieblichen Organisationen. Dabei steht wegen der Menge zu verarbeitender Einheiten die Alternative einer intellektuellen Dokumenterschließung nicht zur Verfügung. Andererseits wachsen die Anforderung an eine automatische Textanalyse, der das klassische IR nicht gerecht wird.

Der Mehrzahl der hiervon betroffenen textuellen Einheiten fehlt die explizite Strukturiertheit formaler Datenstrukturen. Vielmehr weisen sie je nach Text- bzw. Dokumenttyp ganz unterschiedliche Strukturierungsgrade auf. Dabei korreliert die Flexibilität der Organisationsziele negativ mit dem *Grad an explizierter Strukturiertheit* und positiv mit der *Anzahl* jener Texte und Texttypen (E-Mails, Memos, Expertisen, technische Dokumentationen etc.), die im Zuge ihrer Realisierung produziert bzw. rezipiert werden. Vor diesem Hintergrund entsteht ein Bedarf an *Texttechnologien*, die ihren Benutzern nicht nur „intelligente" Schnittstellen zur Textrezeption anbieten, sondern zugleich auf inhaltsorientierte Text-

analysen zielen, um auf diese Weise aufgabenrelevante Daten explorieren und kontextsensitiv aufbereiten zu helfen.

> *Das Text Mining ist mit dem Versprechen verbunden, eine solche Technologie darzustellen bzw. sich als solche zu entwickeln.*

Dieser einheitlichen Problembeschreibung stehen konkurrierende Textmining-Spezifikationen gegenüber, was bereits die Vielfalt der Namensgebungen verdeutlicht. So finden sich neben der Bezeichnung *Text Mining* (Joachims & Leopold 2002; Tan 1999) die Alternativen

- *Text Data Mining* (Hearst 1999b; Merkl 2000),
- *Textual Data Mining* (Losiewicz et al. 2000),
- *Text Knowledge Engineering* (Hahn & Schnattinger 1998),
- *Knowledge Discovery in Texts* (Kodratoff 1999) oder
- *Knowledge Discovery in Textual Databases* (Feldman & Dagan 1995).

Dabei lässt bereits die Namensgebung erkennen, dass es sich um Analogiebildungen zu dem (nur unwesentlich älteren) Forschungsgebiet des *Data Mining* (DM; als Bestandteil des *Knowledge Discovery in Databases* – KDD) handelt. Diese Namensvielfalt findet ihre Entsprechung in widerstreitenden Aufgabenzuweisungen. So setzt beispielsweise Sebastiani (2002) Informationsextraktion und Text Mining weitgehend gleich, wobei er eine Schnittmenge zwischen Text Mining und Textkategorisierung ausmacht (siehe auch Dörre et al. 1999). Demgegenüber betrachten Kosala & Blockeel (2000) Informationsextraktion und Textkategorisierung lediglich als Teilbereiche des ihrer Ansicht nach umfassenderen Text Mining, während Hearst (1999a) im Gegensatz hierzu Informationsextraktion und Textkategorisierung explizit aus dem Bereich des *explorativen* Text Mining ausschließt.

2 Sichten auf das Text Mining

Trotz der zuletzt erläuterten Begriffsvielfalt sind mehrere Hauptströmungen erkennbar, die teils aufgabenorientierte, teils methodische Kriterien in den Vordergrund ihres Text Mining-Begriffs rücken. Dabei handelt es sich IR-, DM-, methoden- und wissensorientierte Ansätze.

2.1 Die Information Retrieval-Perspektive

Bereits Jacobs (1992) konzipiert ein *textbasiertes intelligentes System*, das auf eine Verbesserung von Retrieval-Ergebnissen durch automatische Zusammenfassung von Texten, ihre Kategorisierung und hypertextuelle Vernetzung zielt und greift damit den in späteren Jahren im Bereich von Suchmaschinen erfolgreichen Ansätzen zur Analyse von Hypertextstrukturen vor (vgl. Salton et al. 1994; Allan 1997).

Mit dem Ansatz von Jacobs vergleichbar thematisiert Göser (1997) – in dieser Zeitschrift und als einer der Ersten im deutschsprachigen Bereich – das Text Mining aus der Perspektive des inhaltsbasierten, benutzerorientierten Information Retrieval.

Ansätzen dieser Art ist die Auffassung gemeinsam, dass das Text Mining der Verbesserung des Information Retrieval mittels Textzusammenfassungen und Informationsextraktion diene. Obgleich mehrere Ansätze das IR als Konstituente des Text Mining-Prozesses identifizieren, besteht weitgehend Einigkeit darüber, dass IR und Text Mining verschiedene Bereiche darstellen. Diese kritische Abkehr bringt unter anderem folgende Perspektive zum Ausdruck:

2.2 Die Data-Mining-Perspektive

Fayyad et al. (1996a, b) beschreiben *Knowledge Discovery in Databases* (KDD) als einen Ansatz zur Identifikation von *„valid, novel, potentially useful, and ultimately understandable patterns“*, der neben Datenaufbereitungs-, Evaluations- und Interpretationsschritten explorative Datenanalysen in Form des *data mining* umfasst.

Eine wiederkehrende Interpretation des Text Mining besteht nun darin, dieses als *Data Mining auf textuellen Daten* zu definieren (Rajman & Besançon 1998). Text Mining bedeutet demgemäß kein verbessertes Information Retrieval, sondern die Exploration von (interpretationsbedürftigen) Daten aus Texten. In Analogie hierzu beschreibt Kodratoff (1999) *Knowledge Discovery in Texts* (KDT) als Exploration von „nützlichem“ Wissen aus Texten. Ein vergleichbarer Ansatz stammt von Losiewicz et al. (2000), die in ihrem Modell IR-, IE-, KDD- und Visualisierungskomponenten vereinigen. All diesen Ansätzen ist gemeinsam, dass sie trotz der Analogie zum KDD die Unterscheidung von KDT (Gesamtprozess) und Text Mining (Teilprozess) ebenso vermissen lassen, wie eine Definition der für das KDD zentralen Begriffe des *Wissens*, der *Nützlichkeit* und der *Verständlichkeit*.

2.3 Die methodische Perspektive

In ihrem Leitartikel zum Themenheft *Text Mining* der Zeitschrift *KI* bezeichnen Joachims & Leopold (2002) das Text Mining als *„eine Menge von Methoden zur (halb-)automatischen Auswertung großer Mengen natürlichsprachlicher Texte“*, womit sie als Folge der reklamierten Multidisziplinarität seine Methodenpluralität betonen. Das Einsatzgebiet dieser Methoden sehen sie in der partiellen, fehlertoleranten und in der Regel statistischen Textanalyse, ob zu dem Zweck der Textkategorisierung, der Informationsextraktion und Textzusammenfassung oder der Visualisierung von Textrelationen. Im Zentrum dieser Konzeption steht die Feststellung der methodischen Unselbstständigkeit des Text Mining: Als ein Sammelbegriff subsumiert es vielfältige Textanalysemethoden, auf deren Weiterentwicklung und Integration fokussiert wird.

2.4 Die wissensorientierte Perspektive

Im Gegensatz hierzu zielt Hearst (1999a) auf die wissensorientierte Eingrenzung des Text Mining, und zwar unter expliziter Abgrenzung von Ansätzen der korpusanalytischen Computerlinguistik und des inhaltsbasierten Information Retrieval. Hearst betont die vielfach kritisierte (Wiegand 1999) Metapher des „Goldschürfens“. Sie definiert Text Mining als textbasierte Datenanalyse zur Exploration von *„heretofore unknown“*, *„never–before encountered information“* in Bezug auf jene „realweltlichen“ (nicht aber sprachlichen) Zusammenhänge, welche die Texte annahmegemäß thematisieren. Unter Absehung von ihrem Vorverarbeitungsstatus bilden Information Retrieval (IR), Informationsextraktion (IE), und Textkategorisierung (TK) folglich keine Kernbestandteile des Text Mining, da sie keine Information *explorieren*, sondern bloß Textmengen mittels Indexmengen erschließen (IR), vorgegebene Schemata mit ihren textuellen Instanzen abgleichen (IE) bzw. Texte auf vordefinierte Kategorien abbilden (TK).

Dabei ist allerdings zu verdeutlichen, dass IR, IE und TK jeweils im Kern funktional definiert sind und mit diesen Konzepten kein Hinweis auf eine konkrete Umsetzungsmethode gegeben ist: Ein Text Mining-Verfahren kann in diesem Sinn durchaus geeignet sein, für ein IR-System geeignete Beschreibungsterme zu ermitteln oder inhaltliche relevante Querbezüge zwischen verschiedenen Termen zu beschreiben.

Anstatt das Text Mining begrifflich weiter einzugrenzen, nennt Hearst Musterbeispiele, die als Prüfsteine für die „Mining“-Tauglichkeit von Textanalyse-Systemen dienen sollen. So verweist sie auf Zitationsanalysen, die zeigen, dass Patente weitgehend auf öffentlich finanzierter Forschung beruhen. Ein weiteres

Beispiel bildet die Analyse von Patientenakten, die kausale Zusammenhänge zwischen der Nichteinnahme von Spurenelementen und Syndromen belegen. Im Zentrum dieser Fallbeispiele steht die Überlegung, dass die jeweils explorierte Information in keinem der analysierten Texte isoliert thematisiert wird, sondern erst durch die Analyse mehrerer Texte zu gewinnen ist.

3 Zwei Grundpositionen

Die Verschiedenheit dieser vier Konzeptionen lässt erahnen, dass sich das Text Mining erst zu formieren beginnt, ohne auf einen bereits anerkannten Text Mining-Begriff zurückgreifen zu können. Dies betrifft in gleicher Weise das zugehörige Methoden- und Aufgabenspektrum. Dennoch lassen sich zwei Grundpositionen ausmachen, welche das Spektrum bestehender Text Mining-Ansätze aufspannen:

3.1 Methodenorientierte Ansätze

Das untere Ende des Spektrums bestehender Mining-Begriffe bilden *methodenorientierte Ansätze*. Sie untersuchen, welche Methoden welche Textanalyse-Aufgaben mit welchem Erfolg zu lösen erlauben, und zwar in Ergänzung, Erweiterung oder Ersetzung von herkömmlichen Methoden des Information Retrieval, der Informationsextraktion oder der Textzusammenfassung.

Im Zentrum steht die Konzeption von Methoden entlang der Prämisse, dass wegen des Fehlens bzw. der unzureichenden Skalierbarkeit von Verfahren zur automatischen Generierung propositionaler Textrepräsentationsmodelle statistische, textoberflächenstrukturelle Analysen unumgänglich sind. Dies betrifft insbesondere Situationen, in denen textuelle Massendaten zu analysieren sind, wie sie im Rahmen der Presse-, Wissenschafts- und betrieblichen Kommunikation anfallen.

Diese Massendaten sind mittlerweile vielfach webbasiert zugänglich und liegen in einer überschaubaren Zahl gängiger, mehr oder weniger strukturierter Formate vor (Office-Formate, das Portable Document Format (PDF), die HyperTextMarkup Language (HTML), zunehmend auch als XML-Dateien (extensible Markup Language)). Vor diesem Hintergrund erweist sich das Web Mining als eine Weiterentwicklung des Text Mining, was weiter unten erläutert wird.

Pragmatisch gesprochen werden massendatentaugliche Ansätze bevorzugt, die (zwar nur) partielle Analysen (dafür aber) zuverlässig und fehlertolerant produzieren, und zwar gegenüber solchen Ansätzen, die zwar (tiefen–)semantische

Analysen erlauben, aufgrund ihrer Arbeitsweise aber weder massendatentauglich noch ausnahmetolerant sind. Folgerichtig werden für die konzeptionierten Methoden nur im statistischen Sinne, nicht aber im diskurssemantischen Sinne explorative Qualitäten gefordert. Anstatt also zu beanspruchen, „verborgene realweltliche Zusammenhänge" anhand von automatischen Textanalysen zu rekonstruieren, werden Texte in einer Weise analysiert, die es Rezipienten der Analyseergebnisse ermöglichen soll, relevante Zusammenhänge effizienter zu *entdecken* oder auch nur zu *identifizieren*.

Diese Perspektive macht deutlich, dass Text Mining-Verfahren in vielen Fällen keine eigenständige Anwendung konstituieren bzw. eine vorgegebene Aufgabenstellung vollständig zu lösen in der Lage sind, sondern dass vielmehr erst die Kopplung z. B. mit intellektuellen Überarbeitungsverfahren ein wunschgemäßes Ergebnis der Textexploration liefert. Dies wird am Beispiel des *ontology engineering* deutlich, das auf die Exploration von (normativen) Wissensstrukturen aus großen Textmengen zielt. Obwohl derzeit kein Text Mining-Verfahren in der Lage sein dürfte, sozusagen „auf Knopfdruck" eine Ontologie zu generieren, können Ergebnisse des Text Mining intellektuell weiterverarbeitet und z. B. mit Hilfe geeigneter Ontologie-Editoren optimiert werden (vgl. dazu Böhm et al. 2002).

Die Last der Exploration nützlicher, unerwarteter Information liegt unter dieser Perspektive auf Seiten der Rezipienten, und für diese Sichtweise erscheint die Metapher des Schürfens durchaus angemessen, da ein gefundener Rohdiamant ohne Weiterverarbeitung (mit anderen Methoden) nur wenig Nutzen aufweist.

3.2 Wissensorientierte Ansätze

Hearsts Vision eines realweltliche Zusammenhänge anhand von Textanalysen selbstständig explorierenden Systems bildet das obere Ende des Text Mining-Spektrums. Die Explorationslast liegt nun umgekehrt auf Seiten des „künstlichen" Text Mining-Systems.

Es ist evident, dass dieser Ansatz an ein propositionales Textrepräsentationsmodell gebunden ist, das Explorationsresultate über Ähnlichkeitsvergleiche textueller Einheiten auf der Basis des strukturindifferenten *Bag-of-words*-Modells des IR hinaus erwartbar macht. Ein Paradebeispiel bilden Anstrengungen zum automatischen Aufbau von so genannten Ontologien und ihre Nutzbarmachung im Zusammenhang des *Semantic Web* (Fensel et al. 2003; Handschuh & Staab 2003). Dem hiermit einhergehenden höheren Automatisierungsanspruch steht allerdings der Mangel an bereits etablierten Systemen und Verfahren gegenüber.

Abgesehen von der Problematik des Begriffs der automatischen Informations- bzw. Wissensexploration (Wiegand 1999) stellt sich jedoch die Frage, ob hier nicht auch dann ein uneinlösbarer Anspruch vorliegt, wenn nicht von Text Mining, sondern korrekter von *explorativer Textdatenanalyse* – von einer Anwendung von Verfahren der explorativen Datenanalyse auf textuelle Daten also – gesprochen wird (Mehler 2004b, a).

Dem Verzicht auf explorative Textanalysen à la Hearst steht eine Vielzahl erprobter und etablierter Methoden gegenüber – vgl. hierzu Hotho et al. (2005) (in diesem Band). Umgekehrt existieren kaum massendatentaugliche Anwendungen, die den Hearstschen Ansprüchen genügen. Offenbar besteht also ein – schon aus der KI-Forschung her bekannter – *trade-off* zwischen Massendatentauglichkeit, Fehlertoleranz und Robustheit auf der einen und analytischem, semantischem Auflösungsvermögen auf der anderen Seite. Der Aspekt der Massendatenanalyse verweist dabei ebenso wie das Schlagwort des *Semantic Web* auf einen Anwendungsbereich des Text Mining, der unter der eigenständigen Bezeichnung *Web Mining* firmiert.

4 Web Mining

Vor dem Hintergrund der unzähligen Menge verfügbarer Webseiten, ihrer Strukturen und Änderungsraten sowie der zahllosen Nutzer und ihrer heterogenen Informationsbedürfnisse problematisieren Kobayashi & Takeda (2000) die beschränkten Möglichkeiten des klassischen Information Retrieval im Web. Hiermit ist ein Aufgabendruck angesprochen, der oben für das Text Mining als richtungsweisend ausgemacht wurde. Dies erlaubt es, mit dem *Web Mining* einen Ausblick auf einen wichtigen Anwendungsbereich des Text Mining zu geben, wobei mit Kosala & Blockeel (2000) drei Teilbereiche zu unterscheiden sind:

4.1 Web Content Mining

Das *Web Content Mining* zielt auf ein verbessertes Browsing mit Hilfe von Verfahren des inhaltsorientierten Information Retrieval (Landauer & Dumais 1997), der Textkategorisierung und -klassifikation wie auch mit Hilfe von annotationsbasierten Abfragesprachen im Rahmen strukturierter Retrieval-Modelle. Ein Paradebeispiel bildet die Suchmaschine *Vivísimo* (Stein & zu Eissen 2004), die Clustering-Verfahren zur Strukturierung von Retrieval-Ergebnissen einsetzt. Anders als die Textkategorisierung und -klassifikation rekurrieren ihre hyper-

textuellen Entsprechungen jedoch auf eine erweiterte Merkmalsselektion, indem sie HTML-Tags (und insbesondere Metatags), DOM-Strukturen[1] und benachbarte Webpages inkorporieren.

4.2 Web Structure Mining

Das *Web Structure Mining* zielt auf die Typisierung von Webdokumenten unter anderem auf der Basis ihrer Linkstrukturen. Ein Paradebeispiel bildet die Ermittlung von Webpages als Kandidaten für *hubs* und *authorities* (Kleinberg (1999), vgl. auch Brin & Page (1998); Page et al. (1998); Lifantsev (1999)). In diesem Zusammenhang ist die Kategorisierung von *web hierarchies*, *directories*, *corporate sites* und *web sites* (Amitay et al. 2003) von Ansätzen zu unterscheiden, die auf die Segmentierung *einzelner* Webpages zielen (Mizuuchi & Tajima 1999). Diesen mikrostrukturellen Analysen stehen makrostrukturelle Betrachtungen der Topologie des Webs gegenüber. So untersucht beispielsweise Adamic (1999) Kürzeste-Wege- und Clusterungseigenschaften von Webpages unter dem Begriff des *Small Worlds*-Phänomens wie es für soziale Netzwerke kennzeichnend ist (Milgram 1967).

4.3 Web Usage Mining

Das *Web Usage Mining* bezieht sich schließlich auf die Analyse des Rezeptionsverhaltens von Web-Nutzern. Hierzu werden unter anderem Zipfsche Modelle herangezogen (Zipf 1949; Cooley et al. 1999). Im Kern sagen diese Modelle aus, dass quantitative Indikatoren der Rezeption webbasierter Dokumente dem semiotischen Präferenzgesetz der Ordnung nach der Wichtigkeit (Tuldava 1998) folgen. In diesem Sinne existiert beispielsweise eine sehr geringe Zahl von Webpages, die häufig angesteuert und lange rezipiert werden. Ihr steht eine große Zahl von Seiten gegenüber, die selten angesteuert und in der Regel nur sehr kurz rezipiert werden, wobei zwischen beiden Bereichen ein fließender Übergang beobachtbar ist, der insgesamt eine extrem schiefe Verteilung erkennen lässt.

Soweit das Web Usage Mining lediglich auf Nutzungsinformation bezüglich besuchter Webseiten (Zuordnungen von Nutzern und Adressen) zurückgreift, überschreitet es die Schwelle zur Textexploration i. e. S. allerdings noch nicht.

1 *Document Object Model*.

4.4 Fazit

Mit dem Web Mining steht dem Text Mining ein breites Bewährungsfeld offen, wobei Menge und Struktur der verfügbaren Webdokumente die Entwicklung stärker strukturorientierter Ansätze erwarten lässt. Dabei dürfte der Konflikt zwischen Massendatentauglichkeit auf der einen und semantischem Auflösungsvermögen auf der anderen Seite, der oben an der Unterscheidung von methoden- und wissensorientierten Verfahren festgemacht wurde, nur durch eine stärkere *computerlinguistische* und zugleich *textlinguistische* Fundierung zu lösen sein.

Der Grund für diese Einschätzung ist darin zu sehen, dass die Ablösung oder doch wenigstens Ergänzung des strukturindifferenten *Bag-of-words*-Modells sich an *Textstruktur*-Modellen orientieren sollte, deren Instanzen nachgewiesenermaßen effizient explorierbar sind. Das Resultat einer solchen Fundierung könnte ferner zeigen, welche äußerst engen Grenzen wissensorientierten Mining-Ansätzen gesetzt sind. Die Kritik der Metapher des *Goldschürfens* bzw. der textbasierten Wissensexploration nimmt diese Grenzziehung im Grunde genommen bereits vorweg (Wiegand 1999; Weber 1999).

Massendatengetriebene Ansätze (im Sinne eines *Text Data Mining*) und wissensorientierte Verfahren schließen keineswegs einander aus: Zum einen zeigen Entwicklungen innerhalb der Computerlinguistik der vergangenen Jahre, dass datenorientierte Verfahren ein unverzichtbares Werkzeug zur Rekonstruktion linguistischen Wissens bilden. Als Beispiele hierfür sind unter anderem das *data oriented parsing* (vgl. Bod et al. 2003), das POS-Tagging (vgl. Brants 2000) oder die latente semantische Analyse (Landauer & Dumais 1997; Schütze 1997) und semantische Räume (Rieger 1989) zu nennen. Auf der anderen Seite erlaubt die Rückkoppelung datenanalytischer Verfahren an explizite (linguistische) Wissensstrukturen die Verbesserung von Text Mining-Resultaten (vgl. Heyer et al. 2001). Hier liegt möglicherweise ein erhebliches Potential für die Optimierung der meist auf rein statistischen Methoden beruhenden Text Mining-Verfahren. Zu überlegen ist insbesondere, wie die Felder *Text Mining* und *Corpuslinguistik* angesichts ihrer sich überlappenden Gegenstandsbereiche noch fruchtbarer interagieren können (Heyer et al. 2005). Letztere befaßt sich bereits sehr viel stärker (und länger) mit Fragen der expliziten Strukturierung großer Textmengen, ihrer (linguistischen) Annotation und ihrer repräsentativen und standardisierten Zusammensetzung, Aspekte, die auch für Optimierung und Bewertung des Text Mining relevant sind. Im Licht des voranstehend zum Web Mining Gesagten ist dieses Potenzial dort besonders offensichtlich, wo das Web als Datengrundlage für die Corpuserstellung herangezogen wird (Kilgarriff & Grefenstette 2003).

Im Zusammenhang dieser Kombinationsmöglichkeiten wird sich das Text Mining auch dahingehend zu bewähren haben, inwieweit es über das „intelligente“ Information-Retrieval (Baeza-Yates & Ribeiro-Neto 1999) bzw. Formen der adaptiven Informationsextraktion (Wilks & Catizone 1999) hinausgeht, um mehr als ein Sammelbegriff für Methoden der explorativen Datenanalyse (Joachims & Leopold 2002) zu gelten, die auf textuelle Daten angewandt werden.

5 Überblick über das Themenheft

Das vorliegende Themenheft deckt das Spektrum methoden- und wissensorientierter Mining-Ansätze ab.

Andreas Hotho, Andreas Nürnberger und Gerhard Paass geben in ihrem Beitrag einen umfassenden Überblick über das Text Mining aus *methodischer Sicht*. Ausgehend von einer disziplinären Einordnung des Text Mining im Kontext verwandter Ansätze (wie Data Mining oder maschinelles Lernen) und Anwendungsbereiche (wie Information Retrieval, Informationsextraktion und Natural Language Processing) erläutern sie grundlegende Methoden der Vorverarbeitung und Repräsentation textueller Einheiten sowie ihrer automatischen Kategorisierung, Klassifikation und Informationsextraktion. Ein besonderes Augenmerk gilt dabei Methoden der Visualisierung von Analyseresultaten, womit der für das Mining kennzeichnende Aspekt der verständlichen Ergebnisaufbereitung angesprochen wird. Schließlich erläutern die Autoren die derzeit wichtigsten Anwendungsbereiche des Text Minings.

Ausgehend von dem Modell des *semantischen Raums* von Burghard Rieger (Rieger 1989) beschreibt Edda Leopold in ihrem Beitrag Verfahren zur Exploration von Ähnlichkeitsrelationen sprachlicher Einheiten. Dies betrifft die latente semantische Analyse ebenso wie ihre probabilistischen Erweiterungen. Als besonders vielversprechend erweisen sich dabei Versuche einer Verbindung von Kategorisierungs- und Klassifikationsverfahren mit Hilfe von *Support Vector Machines*, die Leopold zur Lösung des Dimensionenreduktionsproblems im Rahmen von semantischen Räumen einsetzt, ohne auf die Auswertung hochdimensionaler Merkmalsvektoren verzichten zu müssen.

Eine Synthese der methoden- bzw. wissensorientierten Perspektive schlagen Bloehdorn et al. mit dem Entwurf eines *Ontology-based Framework for Text Mining* vor. Sie gehen davon aus, dass sich Vor- und Nachteile der verschiedenen Perspektiven (massendatentauglich, ressourcensparsam, fehlerträchtig *versus* teuer, qualitativ und infolgedessen im Skopus beschränkt) nicht nur in Einklang bringen lassen, sondern sich sogar wechselseitig befruchten können. Ausgehend

von einer formalen Definition grundlegender ontologischer Konzepte stellen sie eine Systemarchitektur vor, in der vorhandenes ontologisches Wissen für ontologiebasierte Text Mining-Komponenten (Modul *TextToOnto*) fruchtbar gemacht werden können. Die Ontologie ist dabei selbst Erkenntnisziel (Anreicherung der Wissensstruktur, Lernen von Relationen) und Erkenntniswerkzeug, als die ontologischen Strukturen für Anwendungen wie Clusterung und Klassifikation zum Einsatz gebracht werden.

Matthias Dehmer schließlich thematisiert den Aufgabenbereich des *Web Structure Mining*. Ausgehend von einer kritischen Erörterung der Aussagekraft von Indizes von Hypertextgraphen leitet Dehmer zur Klassifikation solcher Graphen über. Die Grundlage hierfür bildet die Einsicht, dass Strukturvergleiche von Webdokumenten nicht länger an den summarischen Indizes ansetzen können, wie sie in der Frühphase der Hypertextmodellierung entwickelt wurden (Botafogo et al. 1992). Demgegenüber zielt Dehmer auf die Entwicklung von Maßen, welche die Ähnlichkeit von Hypertextgraphen automatisch bewerten können sollen.

6 Weiterführende Informationen

Text Mining ist eine noch junge wissenschaftliche, anwendungsorientierte Disziplin. Tabelle (1) gibt ein quantitatives Indiz und mag bei der Einordnung behilflich sein. Die Trefferhäufigkeiten für *Data Mining*, *Text Mining* und *Web Mining* in Google, Google Scholar und Inspec sprechen ein deutliches Bild.

	Google	Google Scholar	INSPEC
Data Mining	6.850.000	122.000	13.784
Text Mining	**301.000**	**4.180**	**409**
Web Mining	136.000	2.790	557

Tabelle 1: Trefferhäufigkeiten für Data Mining, Text Mining und Web Mining (Stand: Mai 2005).

6.1 Literatur zum Text Mining

Es kann aufgrund des voranstehend Gesagten kaum verwundern, dass bisher nur wenige Lehrbücher zum Text Mining vorliegen. Die nachfolgende Liste soll einen knappen Überblick zu den derzeit verfügbaren Werken geben:

- Als ein erstes Beispiel kann das weit verbreitete Data Mining-Lehrbuch von Witten & Frank (2000) gelten, das Text Mining zwar nur am Rande behandelt (Witten & Frank 2000, 331ff.), dafür aber eine Vielzahl analytischer Verfahren vorstellt, die auch für das Text Mining relevant sind.

- Aus computerlinguistischer Sicht empfehlenswert ist Manning & Schütze (2003). Die Autoren vermeiden zwar, das Konzept des *Mining* explizit einzuführen, aber ihr Anspruch „Statistical NLP as we define it comprises all quantitative approaches to automated language processing [...]" (Manning & Schütze 2003, xxxi) und die damit verbundene ausführliche Behandlung auch der automatischen Verarbeitung von textuellen Massendaten macht dieses Lehrbuch zu einer nützlichen Einführung in Mining-relevante Verfahren. Aus der Sicht quantitativer Methoden innerhalb der Textlinguistik ist die Einführung von Altmann (1988) empfehlenswert, welche grundlegende Verteilungsmodelle zur Beschreibung quantitativer Merkmale textueller Einheiten erläutert, auch wenn dieses Buch sonst in keinem direkten Verhältnis zum Text Mining steht.

- Intensiv mit der systemischen Einordnung des Text Mining im Spannungsfeld von numerischer Datenanalyse, Information Retrieval und generischen Verfahren der Strukturidentifikation setzen sich Weiss et al. (2004) auseinander, wobei die Autoren zunächst von der grundsätzlichen Analogie des Text Mining zum Data Mining ausgehen („Text and documents can be transformed into measured values, such as the presence or absence of words, and the same methods that have proven successful for predictive data mining can be applied to text.", (Weiss et al. 2004, v). Diese Einführung zeichnet sich weiterhin durch eine Sammlung praktischer Anwendungsstudien aus.

- Die Charakteristika des Web Mining als wichtigstem Anwendungsgebiet des Text Mining thematisiert (Chakrabarti 2002). Vertieft behandelt werden dort neben Fragen der Akquisition von Web-Dokumenten insbesondere Verfahren des maschinellen Lernens basierend auf hypertextuellen Datenbeständen. Die Darstellung der Verfahren wird durch die Beschreibung ausgewählter Anwendungen (*social network analysis, resource discovery*) ergänzt.

- Eine erste deutschsporachige Monographie zum Text Mining legen Heyer et al. (2005) vor, die vor dem Hintergrund zahlreicher anwendungsnaher Studien ein Gesamtbild des Text Mining-Prozesses skizzieren, das

neben statistischen Analyseverfahren für große Textcorpora auch linguistische Aspekte und traditionelle sprachliche Kategorien wie voranstehend angemahnt ins Blickfeld rücken.

Einige aus Workshops und Konferenzen hervorgegangene Sammelbände der letzten Jahre bieten eine gute Übersicht über aktive Forschungsfelder mit Bezug zum Text Mining; zu nennen sind hier Berry (2003), Franke et al. (2003) und Sirmakessis (2004). In ihnen steht weniger die systematische Erschließung des Gegenstandsbereichs Text Mining, sondern die Darstellung typischer Verfahren und Anwendungen im Mittelpunkt, von denen nachfolgend Beispiele genannt seien:

- Trenderkennung und Themenidentifikation durch Text Mining,
- Auffinden von Synonymen in Textcorpora,
- adaptives und kollaboratives Information Retrieval sowie
- Clustering und Merkmalsextraktion aus Texten.

6.2 Tagungen

So vielfältig wie die Anwendungsmöglichkeiten des Text Mining sind auch die Tagungen und Workshops, in denen sich einschlägige Beiträge finden:

- Konferenzen mit primär *computerlinguistischem* oder *sprachtechnologischem* Bezug – *International Conference on Computational Linguistics* (COLING), *Meeting of the (EuroAssociation for Computational Linguistics* (ACL, EACL), *International Conference on Linguistic Resources and Evaluation* (LREC), in Deutschland die *GLDV-Frühjahrstagung* (GLDV).
- Text Mining-Ansätze im Umfeld des *Data Mining* und des *maschinellen Lernens* – *International Conference on Machine Learning* (ICML), *European Conference on Machine Learning* (ECML), *International Conference on Knowledge Discovery and Data Mining* (KDD), *Principles and Practice of Knowledge Discovery in Databases* (PKDD), *International Conference on Data Mining, Text Mining and their Business Applications*.
- Da Text Mining-Verfahren mittlerweile auch in der *KI-Forschung* als wichtige Methode akzeptiert werden, finden sich in einschlägigen KI-Tagungen vermehrt Beiträge mit Text Mining-Bezug – *International Joint Conference on Artificial Intelligence* (IJCAI), *National Conference on Artificial Intelligence* (AAAI).

- Weitere relevante Konferenzen finden sich in den Bereichen *Information Retrieval* (*Conference on Research and Development in Information Retrieval* (SIGIR)), *Wissensmanagement* (*International Conference on Information and Knowledge Management* (CIKM), *International Conference on Knowledge Management* (I-Know)) sowie auf dem Gebiet *webbasierter Informationssysteme* (*International World Wide Web Conference* (WWW)) und der *automatischen Klassifikation* (*Annual Conference of the German Classification Society*).

Diese Breite an Konferenzen mit Text Mining-relevanten Inhalten zeigt, dass sich das Text Mining transdisziplinär etabliert hat, wobei Forscher aus den Bereichen Computerlinguistik, Informatik und verwandten Disziplinen zunehmend interdisziplinär kooperieren. Sie findet sich denn auch in dem vorliegenden Themenheft wieder, dessen Autoren aus den Bereichen Computerlinguistik und quantitative Linguistik sowie Informatik und Mathematik stammen.

Literatur

Adamic, L. A. (1999). The small world web. In S. Abiteboul & A.-M. Vercoustre (Eds.), *Proc. 3rd European Conf. Research and Advanced Technology for Digital Libraries, ECDL*, number 1696 in Lecture Notes in Computer Science (pp. 443–452). Berlin/Heidelberg/New York: Springer.

Allan, J. (1997). Building hypertext using information retrieval. *Information Processing and Management*, *33*(2), 145–159.

Altmann, G. (1988). *Wiederholungen in Texten*. Bochum: Brockmeyer.

Amitay, E., Carmel, D., Darlow, A., Lempel, R., & Soffer, A. (2003). The connectivity sonar: detecting site functionality by structural patterns. In *Proc. of the 14th ACM conference on Hypertext and Hypermedia*, (pp. 38–47).

Baeza-Yates, R. & Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Reading, Massachusetts: Addison-Wesley.

Berry, M. W. (2003). *Survey of text mining*. New York: Springer.

Böhm, K., Heyer, G., Quasthoff, U., & Wolff, C. (2002). Topic map generation using text mining. *J.UCS - Journal of Universal Computer Science*, *8*(6), 623–633.

Bod, R., Scha, R., & Sima'an, K. (2003). *Data-Oriented Parsing*. Stanford: CSLI Publications.

Botafogo, R. A., Rivlin, E., & Shneiderman, B. (1992). Structural analysis of hypertexts: Identifying hierarchies and useful metrics. *ACM Transactions on Information Systems*, *10*(2), 142–180.

Brants, T. (2000). TnT – a statistical part-of-speech tagger. In *Proceedings of the Sixth Conference on Applied Natural Language Processing (ANLP-2000)*, Seattle, WA.

Brin, S. & Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, *30*, 107–117.

Chakrabarti, S. (2002). *Mining the Web: Discovering Knowledge from Hypertext Data*. San Francisco: Morgan Kaufmann.

Cooley, R., Mobasher, B., & Srivastava, J. (1999). Data preparation for mining world wide web browsing patterns. *Journal of Knowledge and Information Systems*, *1*(1), 5–32.

Dörre, J., Gerstl, P., & Seiffert, R. (1999). Text mining: Finding nuggets in mountains of textual data. In Chaudhuri, S. & Madigan, D. (Eds.), *The Fifth ACM SIKKDD International Conference on Knowledge Discovery and Data Mining*, (pp. 398–401)., New York. ACM.

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996a). The kdd process for extracting useful knowledge from volumes of data. *Communications of the ACM*, *39*(11), 27–34.

Fayyad, U. M., Piatetsky-Shapiro, G., & Smyth, P. (1996b). From data mining to knowledge discovery: An overview. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, & R. Uthurusamy (Eds.), *Advances in Knowledge Discovery and Data Mining* (pp. 1–34). Menlo Park, California: AIII Press/MIT Press.

Feldman, R. (2001). Mining unstructured data. In *Tutorial Notes for ACM SIGKDD 1999 International Conference on Knowledge Discovery and Data Mining*, (pp. 182–236). ACM.

Feldman, R. & Dagan, I. (1995). Knowledge discovery in textual databases (kdt). In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD'95)*, (pp. 112–117).

Fensel, D., Hendler, J., Lieberman, H., & Wahlster, W. (2003). *Spinning the Semantic Web. Bringing the World Wide Web to Its Full Potential*. Cambridge, Massachusetts: MIT Press.

Franke, J., Nakhaeizadeh, G., & Renz, I. (2003). *Text Mining, Theoretical Aspects and Applications*. Physica-Verlag.

Göser, S. (1997). Inhaltsbasiertes Information Retrieval: Die TextMining-Technologie. *LDV Forum*, *14*(1), 48–52.

Hahn, U. & Schnattinger, K. (1998). Towards text knowledge engineering. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98) and of the 10th Conference on Innovative Applications of Artificial Intelligence (IAAI-98)*, (pp. 524–531)., Menlo Park. AAAI Press.

Handschuh, S. & Staab, S. (2003). *Annotation for the Semantic Web*, volume 96 of *Frontiers in Artificial Intelligence and Applications*. IOS.

Hearst, M. A. (1999a). Untangling text data mining. In *Proceedings of ACL'99: the 37th Annual Meeting of the Association for Computational Linguistics, University of Maryland, 1999*.

Hearst, M. A. (1999b). User interfaces and visualization. In R. A. Baeza-Yates & B. Ribeiro-Neto (Eds.), *Modern Information Retrieval* chapter 10, (pp. 257–323). Addison Wesley.

Heyer, G., Läuter, M., Quasthoff, U., & Wolff, C. (2001). Wissensextraktion durch linguistisches Postprocessing bei der Corpusanalyse. In Lobin, H. (Ed.), *Sprach- und Texttechnologie in digitalen Medien. Proc. GLDV-Jahrestagung 2001*, (pp. 71–83).

Heyer, G., Quasthoff, U., & Wittig, T. (2005). *Wissensrohstoff Text. Text Mining: Konzepte, Algorithmen, Ergebnisse*. Bochum: W3L.

Hotho, A., Nürnberger, A., & Paaß, G. (2005). A brief survey of text mining. *LDV-Forum*, *20*(1), 19–63.

Jacobs, P. S. (1992). Introduction: Text power and intelligent systems. In P. S. Jacobs (Ed.), *Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval* (pp. 1–8). Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Joachims, T. & Leopold, E. (2002). Themenheft: Text-Mining. Vorwort der Herausgeber. *Künstliche Intelligenz*, *2*, 4.

Kilgarriff, A. & Grefenstette, G. (2003). Introduction to the special issue on the web as corpus. *Computational Linguistics*, *29*(3), 333–347.

Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM*, *46*(5), 604–632.

Kobayashi, M. & Takeda, K. (2000). Information retrieval on the web. *ACM Computing Surveys*, *32*(2), 144–173.

Kodratoff, Y. (1999). Knowledge discovery in texts: A definition and applications. In Rás, Z. W. & Skowron, A. (Eds.), *Proceedings of the 11th International Symposium on Foundations of Intelligent Systems (ISMIS '99)*, (pp. 16–29)., Berlin/Heidelberg/New York. Springer.

Kosala, R. & Blockeel, H. (2000). Web mining research: A survey. *SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery & Data Mining*, *2*(1), 1–15.

Landauer, T. K. & Dumais, S. T. (1997). A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, *104*(2), 211–240.

Lifantsev, M. (1999). Rank computation methods for web documents. Technical Report TR-76, ECSL, Department of Computer Science, SUNY at Stony Brook, Stony Brook/NY.

Losiewicz, P., Oard, D. W., & Kosthoff, R. N. (2000). Textual data mining to support science and technology management. *Journal of Intelligent Information Systems*, *15*, 99–119.

Manning, C. D. & Schütze, H. (2003). *Foundations of Statistical Natural Language Processing* (6. Aufl. ed.). Cambridge, Massachusetts: MIT Press.

Mehler, A. (2001). Aspects of text mining. From computational semiotics to systemic functional hypertexts. *Australian Journal of Information Systems*, *8*(2), 129–141.

Mehler, A. (2004a). Automatische Synthese Internet-basierter Links für digitale Bibliotheken. *Osnabrücker Beiträge zur Sprachtheorie*, *68*, 31–53.

Mehler, A. (2004b). Textmining. In H. Lobin & L. Lemnitzer (Eds.), *Texttechnologie. Perspektiven und Anwendungen* (pp. 329–352). Tübingen: Stauffenburg.

Merkl, D. (2000). Text data mining. In R. Dale, H. Moisl, & H. Somers (Eds.), *Handbook of Natural Language Processing* (pp. 889–903). New York: Dekker.

Milgram, S. (1967). The small world problem. *Psychology Today*, *61*, 60 – 67.

Mizuuchi, Y. & Tajima, K. (1999). Finding context paths for web pages. In *Proceedings of the 10th ACM Conference on Hypertext and Hypermedia*, (pp. 13–22).

Page, L., Brin, S., Motwani, R., & Winograd, T. (1998). The pagerank citation ranking: Bringing order to the web. Technical report, Stanford University, Stanford Digital Library Technologies Project, Stanford/CA.

Rajman, M. & Besançon, R. (1998). Text mining — knowledge extraction from unstructured textual data. In Rizzi, A., Vichi, M., & Bock, H.-H. (Eds.), *Advances in Data Science and Classification: Proc. of 6th Conference of International Federation of Classification Societies (IFCS-98)*, (pp. 473–480)., Berlin/Heidelberg/New York. Springer.

Rieger, B. (1989). *Unscharfe Semantik: die empirische Analyse, quantitative Beschreibung, formale Repräsentation und prozedurale Modellierung vager Wortbedeutungen in Texten*. Frankfurt a.M.: Peter Lang.

Salton, G., Allan, J., & Buckley, C. (1994). Automatic structuring and retrieval of large text files. *Communications of the ACM*, *37*(2), 97–108.

Schütze, H. (1997). *Ambiguity Resolution in Language Learning: Computational and Cognitive Models*, volume 71 of *CSLI Lecture Notes*. Stanford: CSLI Publications.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, *34*(1), 1–47.

Sirmakessis, S. (2004). *Text Mining and its Applications*. Number 138 in Studies in Fuzziness and Soft Computing. Berlin, DE: Springer-Verlag.

Stein, B. & zu Eissen, S. M. (2004). Automatische Kategorisierung für Web-basierte Suche - Einführung, Techniken und Projekte. *KI - Künstliche Intelligenz*, *18*(4), 11–17.

Tan, A.-H. (1999). Text mining: The state of the art and the challenges. In *Proc. of the Pacific Asia Conference on Knowledge Discovery and Data Mining PAKDD'99*, (pp. 65–70).

Tuldava, J. (1998). *Probleme und Methoden der quantitativ-systemischen Lexikologie*. Trier: WVT.

Weber, N. (1999). *Die Semantik von Bedeutungsexplikationen*, volume 3 of *Sprache, Sprechen und Computer/Computer Studies in Language and Speech*. Frankfurt am Main: Lang.

Weiss, S. M., Indurkhya, N., Zhang, T., & Damerau, F. J. (2004). *Text Mining. Predictive Methods for Analyzing Unstructured Information*. New York: Springer.

Wiegand, H. E. (1999). Wissen, Wissensrepräsentation und Printwörterbücher. In Heid, U., Evert, Lehmann, E., & Rohrer, C. (Eds.), *Proceedings of the 9th Euralex International Congress, August 8.-12. 2000, Stuttgart*, (pp. 15–38)., Stuttgart. Institut für maschinelle Sprachverarbeitung.

Wilks, Y. & Catizone, R. (1999). Can we make information extraction more adaptive. In Pazienza, M. T. (Ed.), *Information Extraction. Towards Scalable, Adaptable Systems*, (pp. 1–16)., Berlin/Heidelberg/New York. Springer.

Witten, I. H. & Frank, E. (2000). *Data Mining. Practical Machine Learning Tools and Techniques with Java Implementations*. San Francisco: Morgan Kaufmann.

Zipf, G. K. (1949). *Human Behavior and the Principle of Least Effort. An Introduction to Human Ecology*. Cambridge/MA: Addison-Wesley.

Andreas Hotho, Andreas Nürnberger, and Gerhard Paaß

A Brief Survey of Text Mining

The enormous amount of information stored in unstructured texts cannot simply be used for further processing by computers, which typically handle text as simple sequences of character strings. Therefore, specific (pre-)processing methods and algorithms are required in order to extract useful patterns. Text mining refers generally to the process of extracting interesting information and knowledge from unstructured text. In this article, we discuss text mining as a young and interdisciplinary field in the intersection of the related areas information retrieval, machine learning, statistics, computational linguistics and especially data mining. We describe the main analysis tasks preprocessing, classification, clustering, information extraction and visualization. In addition, we briefly discuss a number of successful applications of text mining.

1 Introduction

As computer networks become the backbones of science and economy enormous quantities of machine readable documents become available. There are estimates that 85% of business information lives in the form of text (TMS05 2005). Unfortunately, the usual logic-based programming paradigm has great difficulties in capturing the fuzzy and often ambiguous relations in text documents. Text mining aims at disclosing the concealed information by means of methods which on the one hand are able to cope with the large number of words and structures in natural language and on the other hand allow to handle vagueness, uncertainty and fuzziness.

In this paper we describe text mining as a truly interdisciplinary method drawing on information retrieval, machine learning, statistics, computational linguistics and especially data mining. We first give a short sketch of these methods and then define text mining in relation to them. Later sections survey state of the art approaches for the main analysis tasks preprocessing, classification, clustering, information extraction and visualization. The last section exemplifies text mining in the context of a number of successful applications.

1.1 Knowledge Discovery

In literature we can find different definitions of the terms knowledge discovery or knowledge discovery in databases (KDD) and data mining. In order to distinguish data mining from KDD we define KDD according to Fayyad as follows (Fayyad et al. 1996):

> Knowledge Discovery in Databases (KDD) is the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data.

The analysis of data in KDD aims at finding hidden patterns and connections in these data. By data we understand a quantity of facts, which can be, for instance, data in a database, but also data in a simple text file. Characteristics that can be used to measure the quality of the patterns found in the data are the comprehensibility for humans, validity in the context of given statistic measures, novelty and usefulness. Furthermore, different methods are able to discover not only new patterns but to produce at the same time generalized models which represent the found connections. In this context, the expression "potentially useful" means that the samples to be found for an application generate a benefit for the user. Thus the definition couples knowledge discovery with a specific application.

Knowledge discovery in databases is a process that is defined by several processing steps that have to be applied to a data set of interest in order to extract useful patterns. These steps have to be performed iteratively and several steps usually require interactive feedback from a user. As defined by the CRoss Industry Standard Process for Data Mining (Crisp DM[1]) model (crispdm and CRISP99 1999) the main steps are: (1) business understanding[2], (2) data understanding, (3) data preparation, (4) modelling, (5) evaluation, (6) deployment (cf. fig. 1[3]). Besides the initial problem of analyzing and understanding the overall task (first two steps) one of the most time consuming steps is data preparation. This is especially of interest for text mining which needs special preprocessing methods to convert textual data into a format

1 CRoss Industry Standard Process for Data Mining Homepage, `http://www.crisp-dm.org/` [accessed May 2005].

2 Business understanding could be defined as understanding the problem we need to solve. In the context of text mining, for example, that we are looking for groups of similar documents in a given document collection.

3 The figure is taken from the Crisp-DM homepage, `http://www.crisp-dm.org/Process/index.htm` [accessed May 2005].

which is suitable for data mining algorithms. The application of data mining algorithms in the modelling step, the evaluation of the obtained model and the deployment of the application (if necessary) are closing the process cycle. Here the modelling step is of main interest as text mining frequently requires the development of new or the adaptation of existing algorithms.

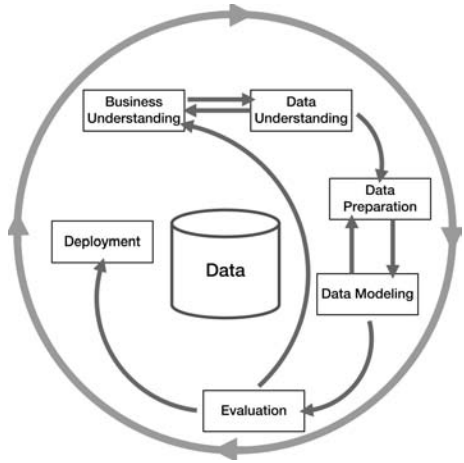

Figure 1: Phases of Crisp DM

1.2 Data Mining, Machine Learning and Statistical Learning

Research in the area of data mining and knowledge discovery is still in a state of great flux. One indicator for this is the sometimes confusing use of terms. On the one side there is *data mining as synonym for KDD*, meaning that data mining contains all aspects of the knowledge discovery process. This definition is in particular common in practice and frequently leads to problems to distinguish the terms clearly. The second way of looking at it considers *data mining as part of the KDD-Processes* (see Fayyad et al. (1996)) and describes the modelling phase, i.e. the application of algorithms and methods for the calculation of the searched patterns or models. Other authors like for instance Kumar & Joshi (2003) consider data mining in addition as the search for valuable information in *large quantities of data*. In this article, we equate data mining with the modelling phase of the KDD process.

The roots of data mining lie in most diverse areas of research, which underlines the interdisciplinary character of this field. In the following we briefly discuss the relations to three of the addressed research areas: Databases, machine learning and statistics.

Databases are necessary in order to analyze large quantities of data efficiently. In this connection, a database represents not only the medium for consistent storing and accessing, but moves in the closer interest of research, since the analysis of the data with data mining algorithms can be supported by databases and thus the use of database technology in the data mining process might be useful. An overview of data mining from the database perspective can be found in Chen et al. (1996).

Machine Learning (ML) is an area of artificial intelligence concerned with the development of techniques which allow computers to "learn" by the analysis of data sets. The focus of most machine learning methods is on symbolic data. ML is also concerned with the algorithmic complexity of computational implementations. Mitchell presents many of the commonly used ML methods in Mitchell (1997).

Statistics has its grounds in mathematics and deals with the science and practice for the analysis of empirical data. It is based on statistical theory which is a branch of applied mathematics. Within statistical theory, randomness and uncertainty are modelled by probability theory. Today many methods of statistics are used in the field of KDD. Good overviews are given in Hastie et al. (2001); Berthold & Hand (1999); Maitra (2002).

1.3 Definition of Text Mining

Text mining or knowledge discovery from text (KDT) — for the first time mentioned in Feldman & Dagan (1995) — deals with the machine supported analysis of text. It uses techniques from information retrieval, information extraction as well as natural language processing (NLP) and connects them with the algorithms and methods of KDD, data mining, machine learning and statistics. Thus, one selects a similar procedure as with the KDD process, whereby not data in general, but text documents are in focus of the analysis. From this, new questions for the used data mining methods arise. One problem is that we now have to deal with problems of — from the data modelling perspective — unstructured data sets.

If we try to define text mining, we can refer to related research areas. For each of them, we can give a different definition of text mining, which is motivated by the specific perspective of the area:

Text Mining = Information Extraction. The first approach assumes that text mining essentially corresponds to information extraction (cf. section 3.3) — the extraction of facts from texts.

Text Mining = Text Data Mining. Text mining can be also defined — similar to data mining — as the application of algorithms and methods from the fields machine learning and statistics to texts with the goal of finding useful patterns. For this purpose it is necessary to pre-process the texts accordingly. Many authors use information extraction methods, natural language processing or some simple preprocessing steps in order to extract data from texts. To the extracted data then data mining algorithms can be applied (see Nahm & Mooney (2002); Gaizauskas (2003)).

Text Mining = KDD Process. Following the knowledge discovery process model (crispdm and CRISP99 1999), we frequently find in literature text mining as a process with a series of partial steps, among other things also information extraction as well as the use of data mining or statistical procedures. Hearst summarizes this in Hearst (1999) in a general manner as the extraction of not yet discovered information in large collections of texts. Also Kodratoff (1999) and Gomez in Hidalgo (2002) consider text mining as process orientated approach on texts.

In this article, we consider text mining mainly as text data mining. Thus, our focus is on methods that extract useful patterns from texts in order to, e.g., categorize or structure text collections or to extract useful information.

1.4 Related Research Areas

Current research in the area of text mining tackles problems of text representation, classification, clustering, information extraction or the search for and modelling of hidden patterns. In this context the selection of characteristics and also the influence of domain knowledge and domain-specific procedures plays an important role. Therefore, an adaptation of the known data mining algorithms to text data is usually necessary. In order to achieve this, one frequently relies on the experience and results of research in information retrieval, natural language processing and information extraction. In all of these areas we also apply data mining methods and statistics to handle their specific tasks:

Information Retrieval (IR). Information retrieval is the finding of documents which contain answers to questions and not the finding of answers itself (Hearst

1999). In order to achieve this goal statistical measures and methods are used for the automatic processing of text data and comparison to the given question. Information retrieval in the broader sense deals with the entire range of information processing, from data retrieval to knowledge retrieval (see Sparck-Jones & Willett (1997) for an overview). Although, information retrieval is a relatively old research area where first attempts for automatic indexing where made in 1975 (Salton et al. 1975), it gained increased attention with the rise of the World Wide Web and the need for sophisticated search engines.

Even though, the definition of information retrieval is based on the idea of questions and answers, systems that retrieve documents based on keywords, i.e. systems that perform *document retrieval* like most search engines, are frequently also called information retrieval systems.

Natural Language Processing (NLP). The general goal of NLP is to achieve a better understanding of natural language by use of computers (Kodratoff 1999). Others include also the employment of simple and durable techniques for the fast processing of text, as they are presented e.g. in Abney (1991). The range of the assigned techniques reaches from the simple manipulation of strings to the automatic processing of natural language inquiries. In addition, linguistic analysis techniques are used among other things for the processing of text.

Information Extraction (IE). The goal of information extraction methods is the extraction of specific information from text documents. These are stored in data base-like patterns (see Wilks (1997)) and are then available for further use. For further details see section 3.3.

In the following, we will frequently refer to the above mentioned related areas of research. We will especially provide examples for the use of machine learning methods in information extraction and information retrieval.

2 Text Encoding

For mining large document collections it is necessary to pre-process the text documents and store the information in a data structure, which is more appropriate for further processing than a plain text file. Even though, meanwhile several methods exist that try to exploit also the syntactic structure and semantics of text, most text mining approaches are based on the idea that a text document can be represented by a set of words, i.e. a text document is described based on the set of words contained in it (*bag-of-words* representation). However, in

order to be able to define at least the importance of a word within a given document, usually a vector representation is used, where for each word a numerical "importance" value is stored. The currently predominant approaches based on this idea are the vector space model (Salton et al. 1975), the probabilistic model (Robertson 1977) and the logical model (van Rijsbergen 1986).

In the following we briefly describe, how a bag-of-words representation can be obtained. Furthermore, we describe the vector space model and corresponding similarity measures in more detail, since this model will be used by several text mining approaches discussed in this article.

2.1 Text Preprocessing

In order to obtain all words that are used in a given text, a *tokenization* process is required, i.e. a text document is split into a stream of words by removing all punctuation marks and by replacing tabs and other non-text characters by single white spaces. This tokenized representation is then used for further processing. The set of different words obtained by merging all text documents of a collection is called the *dictionary* of a document collection.

In order to allow a more formal description of the algorithms, we define first some terms and variables that will be frequently used in the following: Let D be the set of documents and $T = \{t_1, \ldots, t_m\}$ be the dictionary, i.e. the set of all different terms occurring in D, then the absolute frequency of term $t \in T$ in document $d \in D$ is given by $\mathrm{tf}(d, t)$. We denote the term vectors $\vec{t_d} = (\mathrm{tf}(d, t_1), \ldots, \mathrm{tf}(d, t_m))$. Later on, we will also need the notion of the centroid of a set X of term vectors. It is defined as the mean value $\vec{t_X} := \frac{1}{|X|} \sum_{\vec{t_d} \in X} \vec{t_d}$ of its term vectors. In the sequel, we will apply tf also on subsets of terms: For $T' \subseteq T$, we let $\mathrm{tf}(d, T') := \sum_{t \in T'} \mathrm{tf}(d, t)$.

2.1.1 Filtering, Lemmatization and Stemming

In order to reduce the size of the dictionary and thus the dimensionality of the description of documents within the collection, the set of words describing the documents can be reduced by filtering and lemmatization or stemming methods.

Filtering methods remove words from the dictionary and thus from the documents. A standard filtering method is stop word filtering. The idea of stop word filtering is to remove words that bear little or no content information, like articles, conjunctions, prepositions, etc. Furthermore, words that occur extremely often can be said to be of little information content to distinguish

between documents, and also words that occur very seldom are likely to be of no particular statistical relevance and can be removed from the dictionary (Frakes & Baeza-Yates 1992). In order to further reduce the number of words in the dictionary, also (index) term selection methods can be used (see Sect. 2.1.2).

Lemmatization methods try to map verb forms to the infinite tense and nouns to the singular form. However, in order to achieve this, the word form has to be known, i.e. the part of speech of every word in the text document has to be assigned. Since this tagging process is usually quite time consuming and still error-prone, in practice frequently stemming methods are applied.

Stemming methods try to build the basic forms of words, i.e. strip the plural 's' from nouns, the 'ing' from verbs, or other affixes. A stem is a natural group of words with equal (or very similar) meaning. After the stemming process, every word is represented by its stem. A well-known rule based stemming algorithm has been originally proposed by Porter (Porter 1980). He defined a set of production rules to iteratively transform (English) words into their stems.

2.1.2 Index Term Selection

To further decrease the number of words that should be used also indexing or keyword selection algorithms can be used (see, e.g. Deerwester et al. (1990); Witten et al. (1999)). In this case, only the selected keywords are used to describe the documents. A simple method for keyword selection is to extract keywords based on their entropy. E.g. for each word t in the vocabulary the entropy as defined by Lochbaum & Streeter (1989) can be computed:

$$W(t) = 1 + \frac{1}{\log_2 |D|} \sum_{d \in D} P(d,t) \log_2 P(d,t) \quad \text{with} \quad P(d,t) = \frac{\mathrm{tf}(d,t)}{\sum_{l=1}^{n} \mathrm{tf}(d_l,t)} \quad (1)$$

Here the entropy gives a measure how well a word is suited to separate documents by keyword search. For instance, words that occur in many documents will have low entropy. The entropy can be seen as a measure of the importance of a word in the given domain context. As index words a number of words that have a high entropy relative to their overall frequency can be chosen, i.e. of words occurring equally often those with the higher entropy can be preferred.

In order to obtain a fixed number of index terms that appropriately cover the documents, a simple greedy strategy can be applied: From the first document in the collection select the term with the highest relative entropy (or information gain as described in Sect. 3.1.1) as an index term. Then mark this document and all other documents containing this term. From the first of the remaining

unmarked documents select again the term with the highest relative entropy as an index term. Then mark again this document and all other documents containing this term. Repeat this process until all documents are marked, then unmark them all and start again. The process can be terminated when the desired number of index terms have been selected. A more detailed discussion of the benefits of this approach for clustering – with respect to reduction of words required in order to obtain a good clustering performance – can be found in Borgelt & Nürnberger (2004).

An index term selection methods that is more appropriate if we have to learn a classifier for documents is discussed in Sect. 3.1.1. This approach also considers the word distributions within the classes.

2.2 The Vector Space Model

Despite of its simple data structure without using any explicit semantic information, the vector space model enables very efficient analysis of huge document collections. It was originally introduced for indexing and information retrieval (Salton et al. 1975) but is now used also in several text mining approaches as well as in most of the currently available document retrieval systems.

The vector space model represents documents as vectors in m-dimensional space, i.e. each document d is described by a numerical feature vector $w(d) = (x(d,t_1),\ldots,x(d,t_m))$. Thus, documents can be compared by use of simple vector operations and even queries can be performed by encoding the query terms similar to the documents in a query vector. The query vector can then be compared to each document and a result list can be obtained by ordering the documents according to the computed similarity (Salton et al. 1994). The main task of the vector space representation of documents is to find an appropriate encoding of the feature vector.

Each element of the vector usually represents a word (or a group of words) of the document collection, i.e. the size of the vector is defined by the number of words (or groups of words) of the complete document collection. The simplest way of document encoding is to use binary term vectors, i.e. a vector element is set to one if the corresponding word is used in the document and to zero if the word is not. This encoding will result in a simple Boolean comparison or search if a query is encoded in a vector. Using Boolean encoding the importance of all terms for a specific query or comparison is considered as similar. To improve the performance usually term weighting schemes are used, where the weights reflect the importance of a word in a specific document of the considered collection. Large weights are assigned to terms that are used frequently in

relevant documents but rarely in the whole document collection (Salton & Buckley 1988). Thus a weight $w(d,t)$ for a term t in document d is computed by term frequency $\mathrm{tf}(d,t)$ times inverse document frequency $\mathrm{idf}(t)$, which describes the term specificity within the document collection. In Salton et al. (1994) a weighting scheme was proposed that has meanwhile proven its usability in practice. Besides term frequency and inverse document frequency — defined as $idf(t) := log(N/n_t)$ —, a length normalization factor is used to ensure that all documents have equal chances of being retrieved independent of their lengths:

$$w(d,t) = \frac{\mathrm{tf}(d,t)\log(N/n_t)}{\sqrt{\sum_{j=1}^{m} tf(d,t_j)^2(\log(N/n_{t_j}))^2}}, \tag{2}$$

where N is the size of the document collection D and n_t is the number of documents in D that contain term t.

Based on a weighting scheme a document d is defined by a vector of term weights $w(d) = (w(d,t_1),\ldots,w(d,t_m))$ and the similarity S of two documents d_1 and d_2 (or the similarity of a document and a query vector) can be computed based on the inner product of the vectors (by which – if we assume normalized vectors – the cosine between the two document vectors is computed), i.e.

$$S(d_1,d_2) = \sum_{k=1}^{m} w(d_1,t_k)\cdot w(d_2,t_k). \tag{3}$$

A frequently used distance measure is the Euclidian distance. We calculate the distance between two text documents $d_1, d_2 \in D$ as follows:

$$dist(d_1,d_2) = \sqrt[2]{\sum_{k=1}^{m} |w(d_1,t_k) - w(d_2,t_k)|^2}\ . \tag{4}$$

However, the Euclidean distance should only be used for normalized vectors, since otherwise the different lengths of documents can result in a smaller distance between documents that share less words than between documents that have more words in common and should be considered therefore as more similar.

Note that for normalized vectors the scalar product is not much different in behavior from the Euclidean distance, since for two vectors $\vec{x}$ and $\vec{y}$ it is

$$\cos\varphi = \frac{\vec{x}\vec{y}}{|\vec{x}|\cdot|\vec{y}|} = 1 - \frac{1}{2}d^2\left(\frac{\vec{x}}{|\vec{x}|},\frac{\vec{y}}{|\vec{y}|}\right).$$

For a more detailed discussion of the vector space model and weighting schemes

see, e.g. Baeza-Yates & Ribeiro-Neto (1999); Greiff (1998); Salton & Buckley (1988); Salton et al. (1975).

2.3 Linguistic Preprocessing

Often text mining methods may be applied without further preprocessing. Sometimes, however, additional linguistic preprocessing (c.f. Manning & Schütze (2001)) may be used to enhance the available information about terms. For this, the following approaches are frequently applied:

Part-of-speech tagging (POS) determines the part of speech tag, e.g. noun, verb, adjective, etc. for each term.

Text chunking aims at grouping adjacent words in a sentence. An example of a chunk is the noun phrase "the current account deficit".

Word Sense Disambiguation (WSD) tries to resolve the ambiguity in the meaning of single words or phrases. An example is 'bank' which may have – among others – the senses 'financial institution' or the 'border of a river or lake'. Thus, instead of terms the specific meanings could be stored in the vector space representation. This leads to a bigger dictionary but considers the semantic of a term in the representation.

Parsing produces a full parse tree of a sentence. From the parse, we can find the relation of each word in the sentence to all the others, and typically also its function in the sentence (e.g. subject, object, etc.).

Linguistic processing either uses lexica and other resources as well as handcrafted rules. If a set of examples is available machine learning methods as described in section 3, especially in section 3.3, may be employed to learn the desired tags.

It turned out, however, that for many text mining tasks linguistic preprocessing is of limited value compared to the simple bag-of-words approach with basic preprocessing. The reason is that the co-occurrence of terms in the vector representation serves as an automatic disambiguation, e.g. for classification (Leopold & Kindermann 2002). Recently some progress was made by enhancing bag of words with linguistic feature for text clustering and classification (Hotho et al. 2003; Bloehdorn & Hotho 2004).

3 Data Mining Methods for Text

One main reason for applying data mining methods to text document collections is to structure them. A structure can significantly simplify the access to a document collection for a user. Well known access structures are library catalogues or book indexes. However, the problem of manual designed indexes is the time required to maintain them. Therefore, they are very often not up-to-date and thus not usable for recent publications or frequently changing information sources like the World Wide Web. The existing methods for structuring collections either try to assign keywords to documents based on a given keyword set (classification or categorization methods) or automatically structure document collections to find groups of similar documents (clustering methods). In the following we first describe both of these approaches. Furthermore, we discuss in Sect. 3.3 methods to automatically extract useful information patterns from text document collections. In Sect. 3.4 we review methods for visual text mining. These methods allow in combination with structuring methods the development of powerful tools for the interactive exploration of document collections. We conclude this section with a brief discussion of further application areas for text mining.

3.1 Classification

Text classification aims at assigning pre-defined classes to text documents (Mitchell 1997). An example would be to automatically label each incoming news story with a topic like "sports", "politics", or "art". Whatever the specific method employed, a data mining classification task starts with a *training set* $D = (d_1, \ldots, d_n)$ of documents that are already labelled with a class $L \in \mathbb{L}$ (e.g. sport, politics). The task is then to determine a *classification model*

$$f : D \rightarrow \mathbb{L} \qquad f(d) = L \tag{5}$$

which is able to assign the correct class to a new document d of the domain.

To measure the performance of a classification model a random fraction of the labelled documents is set aside and not used for training. We may classify the documents of this *test set* with the classification model and compare the estimated labels with the true labels. The fraction of correctly classified documents in relation to the total number of documents is called *accuracy* and is a first performance measure.

Often, however, the target class covers only a small percentage of the documents. Then we get a high accuracy if we assign each document to the alternative class. To avoid this effect different measures of classification success are often used. *Precision* quantifies the fraction of retrieved documents that are in fact relevant, i.e. belong to the target class. *Recall* indicates which fraction of the relevant documents is retrieved.

$$\text{precision} = \frac{\#\{\text{relevant} \cap \text{retrieved}\}}{\#\text{retrieved}} \qquad \text{recall} = \frac{\#\{\text{relevant} \cap \text{retrieved}\}}{\#\text{relevant}} \tag{6}$$

Obviously there is a trade off between precision and recall. Most classifiers internally determine some "degree of membership" in the target class. If only documents of high degree are assigned to the target class, the precision is high. However, many relevant documents might have been overlooked, which corresponds to a low recall. When on the other hand the search is more exhaustive, recall increases and precision goes down. The *F-score* is a compromise of both for measuring the overall performance of classifiers.

$$F = \frac{2}{1/\text{recall} + 1/\text{precision}} \tag{7}$$

3.1.1 Index Term Selection

As document collections often contain more than 100,000 different words we may select the most informative ones for a specific classification task to reduce the number of words and thus the complexity of the classification problem at hand. One commonly used ranking score is the *information gain* which for a term t_j is defined as

$$IG(t_j) = \sum_{c=1}^{2} p(L_c) \log_2 \frac{1}{p(L_c)} - \sum_{m=0}^{1} p(t_j{=}m) \sum_{c=1}^{2} p(L_c|t_j{=}m) \log_2 \frac{1}{p(L_c|t_j{=}m)} \tag{8}$$

Here $p(L_c)$ is the fraction of training documents with classes L_1 and L_2, $p(t_j{=}1)$ and $p(t_j{=}0)$ is the number of documents with / without term t_j and $p(L_c|t_j{=}m)$ is the conditional probability of classes L_1 and L_2 if term t_j is contained in the document or is missing. It measures how useful t_j is for predicting L_1 from an information-theoretic point of view. We may determine $IG(t_j)$ for all terms and remove those with very low information gain from the dictionary.

In the following sections we describe the most frequently used data mining methods for text categorization.

3.1.2 Naïve Bayes Classifier

Probabilistic classifiers start with the assumption that the words of a document d_i have been generated by a probabilistic mechanism. It is supposed that the class $L(d_i)$ of document d_i has some relation to the words which appear in the document. This may be described by the conditional distribution $p(t_1, \ldots, t_{n_i}|L(d_i))$ of the n_i words given the class. Then the *Bayesian formula* yields the probability of a class given the words of a document (Mitchell 1997)

$$p(L_c|t_1, \ldots, t_{n_i}) = \frac{p(t_1, \ldots, t_{n_i}|L_c)p(L_c)}{\sum_{L \in \mathbb{L}} p(t_1, \ldots, t_{n_i}|L)p(L)}$$

Note that each document is assumed to belong to exactly one of the k classes in $\mathbb{L}$. The prior probability $p(L)$ denotes the probability that an arbitrary document belongs to class L before its words are known. Often the prior probabilities of all classes may be taken to be equal. The conditional probability on the left is the desired *posterior probability* that the document with words $t_1, \ldots, t_{n_i}$ belongs to class L_c. We may assign the class with highest posterior probability to our document.

For document classification it turned out that the specific order of the words in a document is not very important. Even more we may assume that for documents of a given class a word appears in the document irrespective of the presence of other words. This leads to a simple formula for the conditional probability of words given a class L_c

$$p(t_1, \ldots, t_{n_i}|L_c) = \prod_{j=1}^{n_i} p(t_j|L_c)$$

Combining this "naïve" independence assumption with the Bayes formula defines the *Naïve Bayes classifier* (Good 1965). Simplifications of this sort are required as many thousand different words occur in a corpus.

The naïve Bayes classifier involves a learning step which simply requires the estimation of the probabilities of words $p(t_j|L_c)$ in each class by its relative frequencies in the documents of a training set which are labelled with L_c. In the classification step the estimated probabilities are used to classify a new instance according to the Bayes rule. In order to reduce the number of probabilities $p(t_j|L_m)$ to be estimated, we can use index term selection methods as discussed above in Sect. 3.1.1.

Although this model is unrealistic due to its restrictive independence assumption it yields surprisingly good classifications (Dumais et al. 1998; Joachims 1998). It may be extended into several directions (Sebastiani 2002).

As the effort for manually labeling the documents of the training set is high, some authors use unlabeled documents for training. Assume that from a small training set it has been established that word t_i is highly correlated with class L_c. If from unlabeled documents it may be determined that word t_j is highly correlated with t_i, then also t_j is a good predictor for class L_c. In this way unlabeled documents may improve classification performance. In Nigam et al. (2000) the authors used a combination of Expectation-Maximization (EM, Dempster et al. (1977)) and a naïve Bayes classifier and were able to reduce the classification error by up to 30%.

3.1.3 Nearest Neighbor Classifier

Instead of building explicit models for the different classes we may select documents from the training set which are "similar" to the target document. The class of the target document subsequently may be inferred from the class labels of these similar documents. If k similar documents are considered, the approach is also known as *k-nearest neighbor classification*.

There is a large number of similarity measures used in text mining. One possibility is simply to count the number of common words in two documents. Obviously this has to be normalized to account for documents of different lengths. On the other hand words have greatly varying information content. A standard way to measure the latter is the cosine similarity as defined in (3). Note that only a small fraction of all possible terms appear in this sums as $w(d,t) = 0$ if the term t is not present in the document d. Other similarity measures are discussed in Baeza-Yates & Ribeiro-Neto (1999).

For deciding whether document d_i belongs to class L_m, the similarity $S(d_i, d_j)$ to all documents d_j in the training set is determined. The k most similar training documents (neighbors) are selected. The proportion of neighbors having the same class may be taken as an estimator for the probability of that class, and the class with the largest proportion is assigned to document d_i. The optimal number k of neighbors may be estimated from additional training data by cross-validation.

Nearest neighbor classification is a nonparametric method and it can be shown that for large data sets the error rate of the 1-nearest neighbor classifier is never larger than twice the optimal error rate (Hastie et al. 2001). Several studies have shown that k-nearest neighbor methods have very good performance in

practice (Joachims 1998). Their drawback is the computational effort during classification, where basically the similarity of a document with respect to all other documents of a training set has to be determined. Some extensions are discussed in Sebastiani (2002).

3.1.4 Decision Trees

Decision trees are classifiers which consist of a set of rules which are applied in a sequential way and finally yield a decision. They can be best explained by observing the training process, which starts with a comprehensive training set. It uses a divide and conquer strategy: For a training set M with labelled documents the word t_i is selected, which can predict the class of the documents in the best way, e.g. by the information gain (8). Then M is partitioned into two subsets, the subset M_i^+ with the documents containing t_i, and the subset M_i^- with the documents without t_i. This procedure is recursively applied to M_i^+ and M_i^-. It stops if all documents in a subset belong to the same class L_c. It generates a tree of rules with an assignment to actual classes in the leaves.

Decision trees are a standard tool in data mining (Quinlan 1986; Mitchell 1997). They are fast and scalable both in the number of variables and the size of the training set. For text mining, however, they have the drawback that the final decision depends only on relatively few terms. A decisive improvement may be achieved by *boosting decision trees* (Schapire & Singer 1999), i.e. determining a set of complementary decision trees constructed in such a way that the overall error is reduced. Schapire & Singer (2000) use even simpler one step decision trees containing only one rule and get impressive results for text classification.

3.1.5 Support Vector Machines and Kernel Methods

A Support Vector Machine (SVM) is a supervised classification algorithm that recently has been applied successfully to text classification tasks (Joachims 1998; Dumais et al. 1998; Leopold & Kindermann 2002). As usual a document d is represented by a – possibly weighted – vector $(t_{d1}, \ldots, t_{dN})$ of the counts of its words. A single SVM can only separate two classes — a positive class L_1 (indicated by $y = +1$) and a negative class L_2 (indicated by $y = -1$). In the space of input vectors a hyperplane may be defined by setting $y = 0$ in the following linear equation.

$$y = f(\vec{t_d}) = b_0 + \sum_{j=1}^{N} b_j t_{dj}$$

The SVM algorithm determines a hyperplane which is located between the positive and negative examples of the training set. The parameters b_j are adapted in such a way that the distance ξ – called *margin* – between the hyperplane and the closest positive and negative example documents is maximized, as shown in Fig. 3.1.5. This amounts to a constrained quadratic optimization problem which can be solved efficiently for a large number of input vectors.

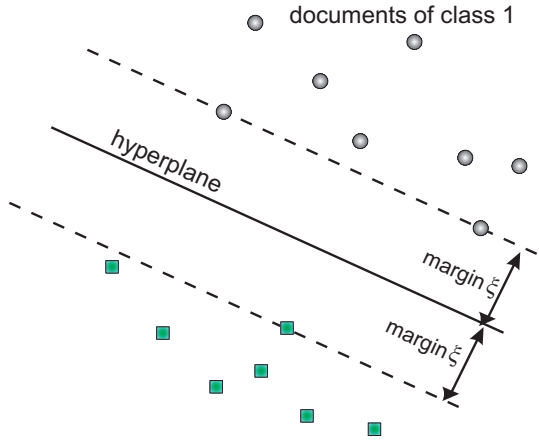

Figure 2: Hyperplane with maximal distance (margin) to examples of positive and negative classes constructed by the support vector machine.

The documents having distance ξ from the hyperplane are called *support vectors* and determine the actual location of the hyperplane. Usually only a small fraction of documents are support vectors. A new document with term vector $\vec{t_d}$ is classified in L_1 if the value $f(\vec{t_d}) > 0$ and into L_2 otherwise. In case that the document vectors of the two classes are not linearly separable a hyperplane is selected such that as few as possible document vectors are located on the "wrong" side.

SVMs can be used with non-linear predictors by transforming the usual input features in a non-linear way, e.g. by defining a *feature map*

$$\phi(t_1, \ldots, t_N) = \left(t_1, \ldots, t_N, t_1^2, t_1 t_2, \ldots, t_N t_{N-1}, t_N^2\right)$$

Subsequently a hyperplane may be defined in the expanded input space. Obviously such non-linear transformations may be defined in a large number of ways.

The most important property of SVMs is that learning is nearly independent of the dimensionality of the feature space. It rarely requires feature selection as it inherently selects data points (the support vectors) required for a good classification. This allows good generalization even in the presence of a large number of features and makes SVM especially suitable for the classification of texts (Joachims 1998). In the case of textual data the choice of the kernel function has a minimal effect on the accuracy of classification: Kernels that imply a high dimensional feature space show slightly better results in terms of precision and recall, but they are subject to overfitting (Leopold & Kindermann 2002).

3.1.6 Classifier Evaluations

During the last years text classifiers have been evaluated on a number of benchmark document collections. It turns out that the level of performance of course depends on the document collection. Table 1 gives some representative results achieved for the Reuters 20 newsgroups collection (Sebastiani 2002, p.38). Concerning the relative quality of classifiers boosted trees, SVMs, and k-nearest neighbors usually deliver top-notch performance, while naïve Bayes and decision trees are less reliable.

Method	F_1-value
naïve Bayes	0.795
decision tree C4.5	0.794
k-nearest neighbor	0.856
SVM	0.870
boosted tree	0.878

Table 1: Performance of Different Classifiers for the Reuters collection

3.2 Clustering

Clustering method can be used in order to find groups of documents with similar content. The result of clustering is typically a partition (also called) clustering $\mathbb{P}$, a set of clusters P. Each cluster consists of a number of documents d. Objects — in our case documents — of a cluster should be similar and dissimilar to documents of other clusters. Usually the quality of clusterings is considered better if the contents of the documents within one cluster are more similar and between the clusters more dissimilar. Clustering methods

group the documents only by considering their distribution in document space (for example, a n-dimensional space if we use the vector space model for text documents).

Clustering algorithms compute the clusters based on the attributes of the data and measures of (dis)similarity. However, the idea of what an ideal clustering result should look like varies between applications and might be even different between users. One can exert influence on the results of a clustering algorithm by using only subsets of attributes or by adapting the used similarity measures and thus control the clustering process. To which extent the result of the cluster algorithm coincides with the ideas of the user can be assessed by evaluation measures. A survey of different kinds of clustering algorithms and the resulting cluster types can be found in Steinbach et al. (2003).

In the following, we first introduce standard evaluation methods and present then details for hierarchical clustering approaches, k-means, bi-section-k-means, self-organizing maps and the EM-algorithm. We will finish the clustering section with a short overview of other clustering approaches used for text clustering.

3.2.1 Evaluation of Clustering Results

In general, there are two ways to evaluate clustering results. One the one hand statistical measures can be used to describe the properties of a clustering result. On the other hand some given classification can be seen as a kind of gold standard which is then typically used to compare the clustering results with the given classification. We discuss both aspects in the following.

Statistical Measures In the following, we first discuss measures which cannot make use of a given classification $\mathbb{L}$ of the documents. They are called indices in statistical literature and evaluate the quality of a clustering on the basis of statistic connections. One finds a large number of indices in literature (see Fickel (1997); Duda & Hart (1973)). One of the most well-known measures is the mean square error. It permits to make statements on quality of the found clusters dependent on the number of clusters. Unfortunately, the computed quality is always better if the number of cluster is higher. In Kaufman & Rousseeuw (1990) an alternative measure, the silhouette coefficient, is presented which is independent of the number of clusters. We introduce both measures in the following.

Mean square error If one keeps the number of dimensions and the number of clusters constant the mean square error (Mean Square error, MSE) can be used

likewise for the evaluation of the quality of clustering. The mean square error is a measure for the compactness of the clustering and is defined as follows:

Definition 1 (MSE) *The means square error (MSE) for a given clustering* $\mathbb{P}$ *is defined as*

$$MSE(\mathbb{P}) = \sum_{P \in \mathbb{P}} MSE(P), \tag{9}$$

whereas the means square error for a cluster P is given by:

$$MSE(P) = \sum_{d \in P} dist(d, \mu_P)^2, \tag{10}$$

and $\mu_P = \frac{1}{|P|} \sum_{d \in P} \vec{t_d}$ *is the centroid of the clusters P and dist is a distance measure.*

Silhouette Coefficient One clustering measure that is independent from the number of clusters is the silhouette coefficient SC(P) (cf. Kaufman & Rousseeuw (1990)). The main idea of the coefficient is to find out the location of a document in the space with respect to the cluster of the document and the next similar cluster. For a good clustering the considered document is nearby the own cluster whereas for a bad clustering the document is closer to the next cluster. With the help of the silhouette coefficient one is able to judge the quality of a cluster or the entire clustering (details can be found in Kaufman & Rousseeuw (1990)). Kaufman & Rousseeuw (1990) gives characteristic values of the silhouette coefficient for the evaluation of the cluster quality. A value for SC(P) between 0.7 and 1.0 signals excellent separation between the found clusters, i.e. the objects within a cluster are very close to each other and are far away from other clusters. The structure was very well identified by the cluster algorithm. For the range from 0.5 to 0.7 the objects are clearly assigned to the appropriate clusters. A larger level of noise exists in the data set if the silhouette coefficient is within the range of 0.25 to 0.5 whereby also here still clusters are identifiable. Many objects could not be assigned clearly to one cluster in this case due to the cluster algorithm. At values under 0.25 it is practically impossible to identify a cluster structure and to calculate meaningful (from the view of application) cluster centers. The cluster algorithm more or less "guessed" the clustering.

Comparative Measures The purity measure is based on the well-known precision measure for information retrieval (cf. Pantel & Lin (2002)). Each resulting cluster P from a partitioning $\mathbb{P}$ of the overall document set D is treated as if it were the result of a query. Each set L of documents of a partitioning $\mathbb{L}$, which is

obtained by manual labelling, is treated as if it is the desired set of documents for a query which leads to the same definitions for precision, recall and f-score as defined in Equations 6 and 7. The two partitions $\mathbb{P}$ and $\mathbb{L}$ are then compared as follows.

The precision of a cluster $P \in \mathbb{P}$ for a given category $L \in \mathbb{L}$ is given by

$$\text{Precision}(P, L) := \frac{|P \cap L|}{|P|}. \tag{11}$$

The overall value for purity is computed by taking the weighted average of maximal precision values:

$$\text{Purity}(\mathbb{P}, \mathbb{L}) := \sum_{P \in \mathbb{P}} \frac{|P|}{|D|} \max_{L \in \mathbb{L}} \text{Precision}(P, L). \tag{12}$$

The counterpart of purity is:

$$\text{InversePurity}(\mathbb{P}, \mathbb{L}) := \sum_{L \in \mathbb{L}} \frac{|L|}{|D|} \max_{P \in \mathbb{P}} \text{Recall}(P, L), \tag{13}$$

where $\text{Recall}(P, L) := \text{Precision}(L, P)$ and the well known

$$\text{F-Measure}(\mathbb{P}, \mathbb{L}) := \sum_{L \in \mathbb{L}} \frac{|L|}{|D|} \max_{P \in \mathbb{P}} \frac{2 \cdot \text{Recall}(P, L) \cdot \text{Precision}(P, L)}{\text{Recall}(P, L) + \text{Precision}(P, L)}, \tag{14}$$

which is based on the F-score as defined in Eq. 7.

The three measures return values in the interval $[0, 1]$, with 1 indicating optimal agreement. Purity measures the homogeneity of the resulting clusters when evaluated against a pre-categorization, while inverse purity measures how stable the pre-defined categories are when split up into clusters. Thus, purity achieves an "optimal" value of 1 when the number of clusters k equals $|D|$, whereas inverse purity achieves an "optimal" value of 1 when k equals 1. Another name in the literature for inverse purity is microaveraged precision. The reader may note that, in the evaluation of clustering results, microaveraged precision is identical to microaveraged recall (cf. e.g. Sebastiani (2002)). The F-measure works similar as inverse purity, but it depreciates overly large clusters, as it includes the individual precision of these clusters into the evaluation.

While (inverse) purity and F-measure only consider 'best' matches between 'queries' and manually defined categories, the *entropy* indicates how large the

information content uncertainty of a clustering result with respect to the given classification is

$$E(\mathbb{P}, L) = \sum_{P \in \mathbb{P}} \text{prob}(P) \cdot E(P), \text{where} \tag{15}$$

$$E(P) = -\sum_{L \in \mathbb{L}} \text{prob}(L|P) \log(\text{prob}(L|P)) \tag{16}$$

where $\text{prob}(L|P) = \text{Precision}(P, L)$ and $\text{prob}(P) = \frac{|P|}{|D|}$. The entropy has the range $[0, log(|\mathbb{L}|)]$, with 0 indicating optimality.

3.2.2 Partitional Clustering

Hierarchical Clustering Algorithms Manning & Schütze (2001); Steinbach et al. (2000) got their name since they form a sequence of groupings or clusters that can be represented in a hierarchy of clusters. This hierarchy can be obtained either in a top-down or bottom-up fashion. Top-down means that we start with one cluster that contains all documents. This cluster is stepwise refined by splitting it iteratively into sub-clusters. One speaks in this case also of the so called "divisive" algorithm. The bottom-up or "agglomerative" procedures start by considering every document as individual cluster. Then the most similar clusters are iteratively merged, until all documents are contained in one single cluster. In practice the divisive procedure is almost of no importance due to its generally bad results. Therefore, only the agglomerative algorithm is outlined in the following.

The agglomerative procedure considers initially each document d of the the whole document set D as an individual cluster. It is the first cluster solution. It is assumed that each document is member of exactly one cluster. One determines the similarity between the clusters on the basis of this first clustering and selects the two clusters p, q of the clustering $\mathbb{P}$ with the minimum distance $dist(p, q)$. Both cluster are merged and one receives a new clustering. One continues this procedure and re-calculates the distances between the new clusters in order to join again the two clusters with the minimum distance $dist(p, q)$. The algorithm stops if only one cluster is remaining.

The distance can be computed according to Eq. 4. It is also possible to derive the clusters directly on the basis of the similarity relationship given by a matrix. For the computation of the similarity between clusters that contain more than one element different distance measures for clusters can be used, e.g. based

on the outer cluster shape or the cluster center. Common linkage procedures that make use of different cluster distance measures are single linkage, average linkage or Ward's procedure. The obtained clustering depends on the used measure. Details can be found, for example, in Duda & Hart (1973).

By means of so-called dendrograms one can represent the hierarchy of the clusters obtained as a result of the repeated merging of clusters as described above. The dendrograms allows to estimate the number of clusters based on the distances of the merged clusters. Unfortunately, the selection of the appropriate linkage method depends on the desired cluster structure, which is usually unknown in advance. For example, single linkage tends to follow chain-like clusters in the data, while complete linkage tends to create ellipsoid clusters. Thus prior knowledge about the expected distribution and cluster form is usually necessary for the selection of the appropriate method (see also Duda & Hart (1973)). However, substantially more problematic for the use of the algorithm for large data sets is the memory required to store the similarity matrix, which consists of $n(n-1)/2$ elements where n is the number of documents. Also the runtime behavior with $O(n^2)$ is worse compared to the linear behavior of KMeans as discussed in the following.

***k*-means** is one of the most frequently used clustering algorithms in practice in the field of data mining and statistics (see Duda & Hart (1973); Hartigan (1975)). The procedure which originally comes from statistics is simple to implement and can also be applied to large data sets. It turned out that especially in the field of text clustering *k*-means obtains good results. Proceeding from a starting solution in which all documents are distributed on a given number of clusters one tries to improve the solution by a specific change of the allocation of documents to the clusters. Meanwhile, a set of variants exists whereas the basic principle goes back to Forgy (1965) or MacQueen (1967). In literature for vector quantization KMeans is also known under the name LloydMaxAlgorithm (Gersho & Gray 1992). The basic principle is shown in the following algorithm:

k-means essentially consists of the steps three and four in the algorithm, whereby the number of clusters k must be given. In step three the documents are assigned to the nearest of the k centroids (also called cluster *prototype*). Step four calculates a new centroids on the basis of the new allocations. We repeat the two steps in a loop (step five) until the cluster centroids do not change any more. The algorithm 5.1 corresponds to a simple hill climbing procedure which typically gets stuck in a local optimum (the finding of the global optimum is a NP complete problem). Apart from a suitable method to determine the starting solution (step one), we require a measure for calculating the distance or

Algorithm 1 The KMeans algorithm

Input: set D, distance measure *dist*, number k of cluster
Output: A partitioning $\mathbb{P}$ of the set D of documents (i. e., a set $\mathbb{P}$ of k disjoint subsets of D with $\bigcup_{P\in\mathbb{P}} P = D$).

1: Choose randomly k data points from D as starting centroids $\vec{t_{P_1}} \ldots \vec{t_{P_k}}$.
2: **repeat**
3: Assign each point of P to the closest centroid with respect to *dist*.
4: (Re-)calculate the cluster centroids $\vec{t_{P_1}} \ldots \vec{t_{P_k}}$ of clusters $P_1 \ldots P_k$.
5: **until** cluster centroids $\vec{t_{P_1}} \ldots \vec{t_{P_k}}$ are stable
6: **return** set $\mathbb{P} := \{P_1, \ldots, P_k\}$, of clusters.

similarity in step three (cf. section 2.1). Furthermore the abort criterion of the loop in step five can be chosen differently e.g. by stopping after a fix number of iterations.

Bi-Section-k-means One fast text clustering algorithm, which is also able to deal with the large size of the textual data is the Bi-Section-KMeans algorithm. In Steinbach et al. (2000) it was shown that Bi-Section-KMeans is a fast and high-quality clustering algorithm for text documents which is frequently outperforming standard KMeans as well as agglomerative clustering techniques.

Bi-Section-KMeans is based on the KMeans algorithm. It repeatedly splits the largest cluster (using KMeans) until the desired number of clusters is obtained. Another way of choosing the next cluster to be split is picking the one with the largest variance. Steinbach et al. (2000) showed neither of these two has a significant advantage.

Self Organizing Map (SOM, cf. Kohonen (1982)) are a special architecture of neural networks that cluster high-dimensional data vectors according to a similarity measure. The clusters are arranged in a low-dimensional topology that preserves the neighborhood relations in the high dimensional data. Thus, not only objects that are assigned to one cluster are similar to each other (as in every cluster analysis), but also objects of nearby clusters are expected to be more similar than objects in more distant clusters. Usually, two-dimensional grids of squares or hexagons are used (cf. Fig. 3).

The network structure of a self-organizing map has two layers (see Fig. 3). The neurons in the input layer correspond to the input dimensions, here the words

of the document vector. The output layer (map) contains as many neurons as clusters needed. All neurons in the input layer are connected with all neurons in the output layer. The weights of the connection between input and output layer of the neural network encode positions in the high-dimensional data space (similar to the cluster prototypes in k-means). Thus, every unit in the output layer represents a cluster center. Before the learning phase of the network, the two-dimensional structure of the output units is fixed and the weights are initialized randomly. During learning, the sample vectors (defining the documents) are repeatedly propagated through the network. The weights of the most similar prototype $\vec{w}_s$ (*winner neuron*) are modified such that the prototype moves toward the input vector $\vec{w}_i$, which is defined by the currently considered document d, i.e. $\vec{w}_i := \vec{t}_d$ (*competitive learning*). As similarity measure usually the Euclidean distance is used. However, for text documents the scalar product (see Eq. 3) can be applied. The weights $\vec{w}_s$ of the winner neuron are modified according to the following equation:

$$\vec{w}_s{}' = \vec{w}_s + \sigma \cdot (\vec{w}_s - \vec{w}_i),$$

where σ is a learning rate.

To preserve the neighborhood relations, prototypes that are close to the winner neuron in the two-dimensional structure are also moved in the same direction. The weight change decreases with the distance from the winner neuron. Therefore, the adaption method is extended by a neighborhood function v (see also Fig. 3):

$$\vec{w}_s{}' = \vec{w}_s + v(i,s) \cdot \sigma \cdot (\vec{w}_s - \vec{w}_i),$$

where σ is a learning rate. By this learning procedure, the structure in the high-dimensional sample data is non-linearly projected to the lower-dimensional topology. After learning, arbitrary vectors (i.e. vectors from the sample set or prior 'unknown' vectors) can be propagated through the network and are mapped to the output units. For further details on self-organizing maps see Kohonen (1984). Examples for the application of SOMs for text mining can be found in Lin et al. (1991); Honkela et al. (1996); Kohonen et al. (2000); Nürnberger (2001); Roussinov & Chen (2001) and in Sect. 3.4.2.

Model-based Clustering Using the EM-Algorithm Clustering can also be viewed from a statistical point of view. If we have k different clusters we may either assign a document d_i with certainty to a cluster (hard clustering) or assign

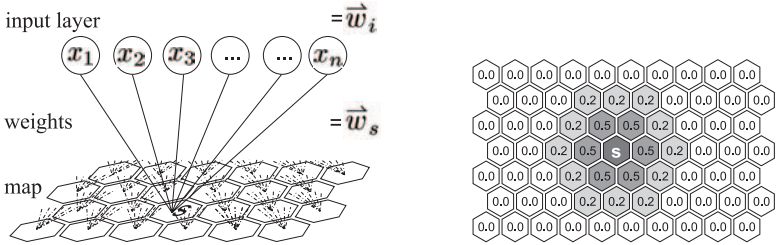

Figure 3: Network architecture of self-organizing maps (*left*) and possible neighborhood function v for increasing distances from s (*right*)

d_i with probability q_{ic} to P_c (soft clustering), where $q_i = (q_{i1}, \ldots, q_{ik})$ is a probability vector $\sum_{c=1}^{k} q_{ic} = 1$.

The underlying statistical assumption is that a document was created in two stages: First we pick a cluster P_c from $\{1, \ldots, k\}$ with fixed probability q_c; then we generate the words t of the document according to a cluster-specific probability distribution $p(t|P_c)$. This corresponds to a mixture model where the probability of an observed document $(t_1, \ldots, t_{n_i})$ is

$$p(t_1, \ldots, t_{n_i}) = \sum_{c=1}^{k} q_c p(t_1, \ldots, t_{n_i}|P_c) \tag{17}$$

Each cluster P_c is a mixture component. The mixture probabilities q_c describe an unobservable "cluster variable" z which may take the values from $\{1, \ldots, k\}$. A well established method for estimating models involving unobserved variables is the EM-algorithm (Hastie et al. 2001), which basically replaces the unknown value with its current probability estimate and then proceeds as if it has been observed. Clustering methods for documents based on mixture models have been proposed by Cheeseman & Stutz (1996) and yield excellent results. Hofmann (2001) formulates a variant that is able to cluster terms occurring together instead of documents.

3.2.3 Alternative Clustering Approaches

Co-clustering algorithm designate the simultaneous clustering of documents and terms (Dhillon et al. 2003). They follow thereby another paradigm than the "classical" cluster algorithm as KMeans which only clusters elements of the

one dimension on the basis of their similarity to the second one, e.g. documents based on terms.

Fuzzy Clustering While most classical clustering algorithms assign each datum to exactly one cluster, thus forming a crisp partition of the given data, fuzzy clustering allows for *degrees of membership*, to which a datum belongs to different clusters (Bezdek 1981). These approaches are frequently more stable. Applications to text are described in, e.g., Mendes & Sacks (2001); Borgelt & Nürnberger (2004).

The Utility of Clustering We have described the most important types of clustering approaches, but we had to leave out many other. Obviously there are many ways to define clusters and because of this we cannot expect to obtain something like the 'true' clustering. Still clustering can be insightful. In contrast to classification, which relies on a prespecified grouping, cluster procedures label documents in a new way. By studying the words and phrases that characterize a cluster, for example, a company could learn new insights about its customers and their typical properties. A comparison of some clustering methods is given in Steinbach et al. (2000).

3.3 Information Extraction

Natural language text contains much information that is not directly suitable for automatic analysis by a computer. However, computers can be used to sift through large amounts of text and extract useful information from single words, phrases or passages. Therefore *information extraction* can be regarded as a restricted form of full natural language understanding, where we know in advance what kind of semantic information we are looking for. The main task is to extract parts of text and assign specific attributes to it.

As an example consider the task to extract executive position changes from news stories: "Robert L. James, chairman and chief executive officer of McCann-Erickson, is going to retire on July 1st. He will be replaced by John J. Donner, Jr., the agencies chief operating officer." In this case we have to identify the following information: Organization (McCann-Erickson), position (chief executive officer), date (July 1), outgoing person name (Robert L. James), and incoming person name (John J. Donner, Jr.).

The task of information extraction naturally decomposes into a series of processing steps, typically including tokenization, sentence segmentation, part-

of-speech assignment, and the identification of named entities, i.e. person names, location names and names of organizations. At a higher level phrases and sentences have to be parsed, semantically interpreted and integrated. Finally the required pieces of information like "position" and "incoming person name" are entered into the database. Although the most accurate information extraction systems often involve handcrafted language-processing modules, substantial progress has been made in applying data mining techniques to a number of these steps.

3.3.1 Classification for Information Extraction

Entity extraction was originally formulated in the Message Understanding Conference (Chinchor 1997). One can regard it as a word-based tagging problem: The word, where the entity starts, get tag "B", continuation words get tag "I" and words outside the entity get tag "O". This is done for each type of entity of interest. For the example above we have for instance the person-words "by (O) John (B) J. (I) Donner (I) Jr. (I) the (O)".

Hence we have a sequential classification problem for the labels of each word, with the surrounding words as input feature vector. A frequent way of forming the feature vector is a binary encoding scheme. Each feature component can be considered as a test that asserts whether a certain pattern occurs at a specific position or not. For example, a feature component takes the value 1 if the previous word is the word "John" and 0 otherwise. Of course we may not only test the presence of specific words but also whether the words starts with a capital letter, has a specific suffix or is a specific part-of-speech. In this way results of previous analysis may be used.

Now we may employ any efficient classification method to classify the word labels using the input feature vector. A good candidate is the Support Vector Machine because of its ability to handle large sparse feature vectors efficiently. Takeuchi & Collier (2002) used it to extract entities in the molecular biology domain.

3.3.2 Hidden Markov Models

One problem of standard classification approaches is that they do not take into account the predicted labels of the surrounding words. This can be done using probabilistic models of sequences of labels and features. Frequently used is the hidden Markov model (HMM), which is based on the conditional distributions of current labels $L^{(j)}$ given the previous label $L^{(j-1)}$ and the distribution of the

current word $t^{(j)}$ given the current and the previous labels $L^{(j)}, L^{(j-1)}$.

$$L^{(j)} \sim p(L^{(j)}|L^{(j-1)}) \qquad t^{(j)} \sim p(t^{(j)}|L^{(j)}, L^{(j-1)}) \tag{18}$$

A training set of words and their correct labels is required. For the observed words the algorithm takes into account all possible sequences of labels and computes their probabilities. An efficient learning method that exploits the sequential structure is the Viterbi algorithm (Rabiner 1989). Hidden Markov models were successfully used for named entity extraction, e.g. in the Identifinder system (Bikel et al. 1999).

3.3.3 Conditional Random Fields

Hidden Markov models require the conditional independence of features of different words given the labels. This is quite restrictive as we would like to include features which correspond to several words simultaneously. A recent approach for modelling this type of data is called *conditional random field* (CRF, cf. Lafferty et al. (2001)). Again we consider the observed vector of words $\mathbf{t}$ and the corresponding vector of labels $\mathbf{L}$. The labels have a graph structure. For a label L_c let $N(c)$ be the indices of neighboring labels. Then $(\mathbf{t}, \mathbf{L})$ is a conditional random field when conditioned on the vector $\mathbf{t}$ of all terms the random variables obey the Markov property

$$p(L_c|\mathbf{t}, L_d; d \neq c) = p(L_c|\mathbf{t}, L_d; d \in N(c)) \tag{19}$$

i.e. the whole vector $\mathbf{t}$ of observed terms and the labels of neighbors may influence the distribution of the label L_c. Note that we do not model the distribution $p(\mathbf{t})$ of the observed words, which may exhibit arbitrary dependencies.

We consider the simple case that the words $\mathbf{t} = (t_1, t_2, \ldots, t_n)$ and the corresponding labels $L_1, L_2, \ldots, L_n$ have a chain structure and that L_c depends only on the preceding and succeeding labels L_{c-1} and L_{c+1}. Then the conditional distribution $p(\mathbf{L}|\mathbf{t})$ has the form

$$p(\mathbf{L}|\mathbf{t}) = \frac{1}{\text{const}} \exp\left(\sum_{j=1}^{n}\sum_{r=1}^{k_j} \lambda_{jr} f_{jr}(L_j, \mathbf{t}) + \sum_{j=1}^{n-1}\sum_{r=1}^{m_j} \mu_{jr} g_{jr}(L_j, L_{j-1}, \mathbf{t})\right) \tag{20}$$

where $f_{jr}(L_j, \mathbf{t})$ and $g_{jr}(L_j, L_{j-1}, \mathbf{t})$ are different features functions related to L_j and the pair L_j, L_{j-1} respectively. CRF models encompass hidden Markov

models, but they are much more expressive because they allow arbitrary dependencies in the observation sequence and more complex neighborhood structures of labels. As for most machine learning algorithms a training sample of words and the correct labels is required. In addition to the identity of words arbitrary properties of the words, like part-of-speech tags, capitalization, prefixes and suffixes, etc. may be used leading to sometimes more than a million features. The unknown parameter values λ_{jr} and μ_{jr} are usually estimated using conjugate gradient optimization routines (McCallum 2003).

McCallum (2003) applies CRFs with feature selection to named entity recognition and reports the following F1-measures for the CoNLL corpus: person names 93%, location names 92%, organization names 84%, miscellaneous names 80%. CRFs also have been successfully applied to noun phrase identification (McCallum 2003), part-of-speech tagging (Lafferty et al. 2001), shallow parsing (Sha & Pereira 2003), and biological entity recognition (Kim et al. 2004).

3.4 Explorative Text Mining: Visualization Methods

Graphical visualization of information frequently provides more comprehensive and better and faster understandable information than it is possible by pure text based descriptions and thus helps to mine large document collections. Many of the approaches developed for text mining purposes are motivated by methods that had been proposed in the areas of explorative data analysis, information visualization and visual data mining. For an overview of these areas of research see, e.g., U. Fayyad (2001); Keim (2002). In the following we will focus on methods that have been specifically designed for text mining or — as a subgroup of text mining methods and a typical application of visualization methods — information retrieval.

In text mining or information retrieval systems visualization methods can improve and simplify the discovery or extraction of relevant patterns or information. Information that allow a visual representation comprises aspects of the document collection or result sets, keyword relations, ontologies or — if retrieval systems are considered — aspects of the search process itself, e.g. the search or navigation path in hyperlinked collections.

However, especially for text collections we have the problem of finding an appropriate visualization for abstract textual information. Furthermore, an *interactive* visual data exploration interface is usually desirable, e.g. to zoom in local areas or to select or mark parts for further processing. This results in great demands on the user interface and the hardware. In the following we give a

brief overview of visualization methods that have been realized for text mining and information retrieval systems.

3.4.1 Visualizing Relations and Result Sets

Interesting approaches to visualize keyword-document relations are, e.g., the Cat-a-Cone model (Hearst & Karadi 1997), which visualizes in a three dimensional representation hierarchies of categories that can be interactively used to refine a search. The InfoCrystal (Spoerri 1995) visualizes a (weighted) boolean query and the belonging result set in a crystal structure. The Lyberworld model (Hemmje et al. 1994) and the visualization components of the SENTINEL Model (Fox et al. 1999) are representing documents in an abstract keyword space.

An approach to visualize the results of a set of queries was presented in Havre et al. (2001). Here, retrieved documents are arranged according to their similarity to a query on straight lines. These lines are arranged in a circle around a common center, i.e. every query is represented by a single line. If several documents are placed on the same (discrete) position, they are arranged in the same distance to the circle, but with a slight offset. Thus, clusters occur that represent the distribution of documents for the belonging query.

3.4.2 Visualizing Document Collections

For the visualization of document collections usually two-dimensional projections are used, i.e. the high dimensional document space is mapped on a two-dimensional surface. In order to depict individual documents or groups of documents usually text flags are used, which represent either a keyword or the document category. Colors are frequently used to visualize the density, e.g. the number of documents in this area, or the difference to neighboring documents, e.g. in order to emphasize borders between different categories. If three-dimensional projections are used, for example, the number of documents assigned to a specific area can be represented by the z-coordinate.

An Example: Visualization Using Self-Organizing Maps Visualization of document collections requires methods that are able to group documents based on their similarity and furthermore that visualize the similarity between discovered groups of documents. Clustering approaches that are frequently used to find groups of documents with similar content (Steinbach et al. 2000) – see also section 3.2 – usually do not consider the neighborhood relations between the obtained cluster centers. Self-organizing maps, as discussed above, are an

alternative approach which is frequently used in data analysis to cluster high dimensional data. The resulting clusters are arranged in a low-dimensional topology that preserves the neighborhood relations of the corresponding high dimensional data vectors and thus not only objects that are assigned to one cluster are similar to each other, but also objects of nearby clusters are expected to be more similar than objects in more distant clusters.

Usually, two-dimensional arrangements of squares or hexagons are used for the definition of the neighborhood relations. Although other topologies are possible for self-organizing maps, two-dimensional maps have the advantage of intuitive visualization and thus good exploration possibilities. In document retrieval, self-organizing maps can be used to arrange documents based on their similarity. This approach opens up several appealing navigation possibilities. Most important, the surrounding grid cells of documents known to be interesting can be scanned for further similar documents. Furthermore, the distribution of keyword search results can be visualized by coloring the grid cells of the map with respect to the number of hits. This allows a user to judge e.g. whether the search results are assigned to a small number of (neighboring) grid cells of the map, or whether the search hits are spread widely over the map and thus the search was – most likely – too unspecific.

A first application of self-organizing maps in information retrieval was presented in Lin et al. (1991). It provided a simple two-dimensional cluster representation (categorization) of a small document collection. A refined model, the WEBSOM approach, extended this idea to a web based interface applied to newsgroup data that provides simple zooming techniques and coloring methods (Honkela et al. 1996; Honkela 1997; Kohonen et al. 2000). Further extensions introduced hierarchies (Merkl 1998), supported the visualization of search results (Roussinov & Chen 2001) and combined search, navigation and visualization techniques in an integrated tool (Nürnberger 2001). A screenshot of the prototype discussed in Nürnberger (2001) is depicted in Fig. 4.

3.4.3 Other Techniques

Besides methods based on self-organizing maps several other techniques have been successfully applied to visualize document collections. For example, the tool VxInsight (Boyack et al. 2002) realizes a partially interactive mapping by an energy minimization approach similar to simulated annealing to construct a three dimensional landscape of the document collection. As input either a vector space description of the documents or a list of directional edges, e.g. defined based on citations of links, can be used. The tool SPIRE (Wise et al. 1995)

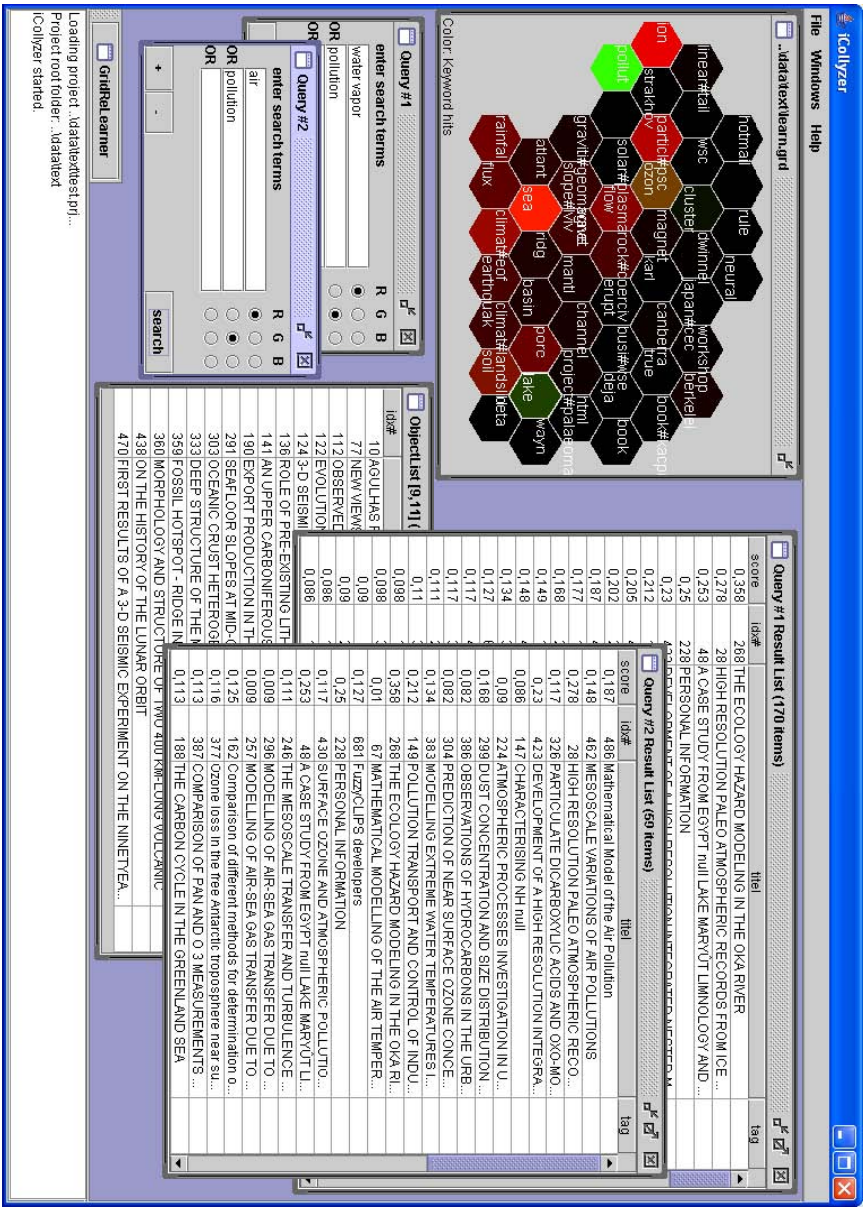

Figure 4: A Prototypical Retrieval System Based on Self-Organizing Maps

applies a three step approach: It first clusters documents in document space, than projects the discovered cluster centers onto a two dimensional surface and finally maps the documents relative to the projected cluster centers. SPIRE offers a scatter plot like projection as well as a three dimensional visualization. The visualization tool SCI-Map (Small 1999) applies an iterative clustering approach to create a network using, e.g., references of scientific publications. The tools visualizes the structure by a map hierarchy with an increasing number of details.

One major problem of most existing visualization approaches is that they create their output only by use of data inherent information, i.e. the distribution of the documents in document space. User specific information can not be integrated in order to obtain, e.g., an improved separation of the documents with respect to user defined criteria like keywords or phrases. Furthermore, the possibilities for a user to interact with the system in order to navigate or search are usually very limited, e.g., to boolean keyword searches and simple result lists.

3.5 Further Application Areas

Further major applications of text mining methods consider the detection of topics in text streams and text summarization.

Topic detection studies the problem of detecting new and upcoming topics in time-ordered document collections. The methods are frequently used in order to detect and monitor (*topic tracking*) news tickers or news broadcasts. An introduction and overview of current approaches can be found in Allan (2002).

Text summarization aims at the creation of a condensed version of a document or a document collection (multidocument summarization) that should contain its most important topics. Most approaches still focus on the idea to extract individual informative sentences from a text. The summary consists then simply of a collection of these sentences. However, recently refined approaches try to extract semantic information from documents and create summaries based on this information (cf. Leskovec et al. (2004)). For an overview see Mani & Maybury (1999) and Radev et al. (2002).

4 Applications

In this section we briefly discuss successful applications of text mining methods in quite diverse areas as patent analysis, text classification in news agencies, bioinformatics and spam filtering. Each of the applications has specific char-

acteristics that had to be considered while selecting appropriate text mining methods.

4.1 Patent Analysis

In recent years the analysis of patents developed to a large application area. The reasons for this are on the one hand the increased number of patent applications and on the other hand the progress that had been made in text classification, which allows to use these techniques in this due to the commercial impact quite sensitive area. Meanwhile, supervised and unsupervised techniques are applied to analyze patent documents and to support companies and also the European patent office in their work. The challenges in patent analysis consists of the length of the documents, which are larger then documents usually used in text classification, and the large number of available documents in a corpus (Koster et al. 2001). Usually every document consist of 5,000 words in average. More than 140,000 documents have to be handled by the European patent office (EPO) per year. They are processed by 2,500 patent examiners in three locations.

In several studies the classification quality of state-of-the-art methods was analyzed. Koster et al. (2001) reported very good result with an 3% error rate for 16,000 full text documents to be classified in 16 classes (mono-classification) and a 6% error rate in the same setting for abstracts only by using the Winnow (Littlestone 1988) and the Rocchio algorithm (Rocchio 1971). These results are possible due to the large amount of available training documents. Good results are also reported in (Krier & Zacca 2002) for an internal EPO text classification application with a precision of 81 % and an recall of 78 %.

Text clustering techniques for patent analysis are often applied to support the analysis of patents in large companies by structuring and visualizing the investigated corpus. Thus, these methods find their way in a lot of commercial products but are still also of interest for research, since there is still a need for improved performance. Companies like IBM offer products to support the analysis of patent text documents. Dorre describes in (Dörre et al. 1999) the IBM Intelligent Miner for text in a scenario applied to patent text and compares it also to data mining and text mining. Coupet & Hehenberger (1998) do not only apply clustering but also give some nice visualization. A similar scenario on the basis of SOM is given in (Lamirel et al. 2003).

4.2 Text Classification for News Agencies

In publishing houses a large number of news stories arrive each day. The users like to have these stories tagged with categories and the names of important persons, organizations and places. To automate this process the Deutsche Presse-Agentur (dpa) and a group of leading German broadcasters (PAN) wanted to select a commercial text classification system to support the annotation of news articles. Seven systems were tested with a two given test corpora of about half a million news stories and different categorical hierarchies of about 800 and 2,300 categories (Paaß & deVries 2005). Due to confidentiality the results can be published only in anonymized form.

For the corpus with 2,300 categories the best system achieved at an F1-value of 39%, while for the corpus with 800 categories an F1-value of 79% was reached. In the latter case a partially automatic assignment based on the reliability score was possible for about half the documents, while otherwise the systems could only deliver proposals for human categorizers. Especially good are the results for recovering persons and geographic locations with about 80% F1-value. In general there were great variations between the performances of the systems.

In a usability experiment with human annotators the formal evaluation results were confirmed leading to faster and more consistent annotation. It turned out, that with respect to categories the human annotators exhibit a relative large disagreement and a lower consistency than text mining systems. Hence the support of human annotators by text mining systems offers more consistent annotations in addition to faster annotation. The Deutsche Presse-Agentur now is routinely using a text mining system in its news production workflow.

4.3 Bioinformatics

Bio-entity recognition aims to identify and classify technical terms in the domain of molecular biology that correspond to instances of concepts that are of interest to biologists. Examples of such entities include the names of proteins, genes and their locations of activity such as cells or organism names. Entity recognition is becoming increasingly important with the massive increase in reported results due to high throughput experimental methods. It can be used in several higher level information access tasks such as relation extraction, summarization and question answering.

Recently the GENIA corpus was provided as a benchmark data set to compare different entity extraction approaches (Kim et al. 2004). It contains 2,000 abstracts from the MEDLINE database which were hand annotated with 36

types of biological entities. The following sentence is an example: "We have shown that <*protein*> interleukin-1 </*protein*> (<*protein*> IL-1 </*protein*>) and <*protein*> IL-2 </*protein*> control <*DNA*> IL-2 receptor alpha (IL-2R alpha) gene </*DNA*> transcription in <*cell_line*> CD4-CD8- murine T lymphocyte precursors </*cell_line*>".

In the 2004 evaluation four types of extraction models were used: Support Vector Machines (SVMs), Hidden Markov Models (HMMs), Conditional Random Fields (CRFs) and the related Maximum Entropy Markov Models (MEMMs). Varying types of input features were employed: lexical features (words), n-grams, orthographic information, word lists, part-of-speech tags, noun phrase tags, etc. The evaluation shows that the best five systems yield an F1-value of about 70% (Kim et al. 2004). They use SVMs in combination with Markov models (72.6%), MEMMs (70.1%), CRFs (69.8%), CRFs together with SVMs (66.3%), and HMMs (64.8%). For practical applications the current accuracy levels are not yet satisfactory and research currently aims at including a sophisticated mix of external resources such as keyword lists and ontologies which provide terminological resources.

4.4 Anti-Spam Filtering of Emails

The explosive growth of unsolicited e-mail, more commonly known as spam, over the last years has been undermining constantly the usability of e-mail. One solution is offered by anti-spam filters. Most commercially available filters use black-lists and hand-crafted rules. On the other hand, the success of machine learning methods in text classification offers the possibility to arrive at anti-spam filters that quickly may be adapted to new types of spam.

There is a growing number of learning spam filters mostly using naive Bayes classifiers. A prominent example is Mozilla's e-mail client. Michelakis et al. (2004) compare different classifier methods and investigate different costs of classifying a proper mail as spam. They find that for their benchmark corpora the SVM nearly always yields best results.

To explore how well a learning-based filter performs in real life, they used an SVM-based procedure for seven months without retraining. They achieved a precision of 96.5% and a recall of 89.3%. They conclude that these good results may be improved by careful preprocessing and the extension of filtering to different languages.

5 Conclusion

In this article, we tried to give a brief introduction to the broad field of text mining. Therefore, we motivated this field of research, gave a more formal definition of the terms used herein and presented a brief overview of currently available text mining methods, their properties and their application to specific problems. Even though, it was impossible to describe all algorithms and applications in detail within the (size) limits of an article, we think that the ideas discussed and the provided references should give the interested reader a rough overview of this field and several starting points for further studies.

References

Abney, S. P. (1991). Parsing by chunks. In R. C. Berwick, S. P. Abney, & C. Tenny (Eds.), *Principle-Based Parsing: Computation and Psycholinguistics* (pp. 257–278). Boston: Kluwer Academic Publishers.

Allan, J. (Ed.). (2002). *Topic Detection and Tracking*. Norwell, MA: Kluwer Academic Publishers.

Baeza-Yates, R. & Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Addison Wesley Longman.

Berthold, M. & Hand, D. J. (Eds.). (1999). *Intelligent data analysis*. Springer-Verlag New York, Inc.

Bezdek, J. C. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms*. New York: Plenum Press.

Bikel, D., Schwartz, R., & Weischedel, R. (1999). An algorithm that learns what's in a name. *Machine learning*, *34*, 211–231.

Bloehdorn, S. & Hotho, A. (2004). Text classification by boosting weak learners based on terms and concepts. In *Proc. IEEE Int. Conf. on Data Mining (ICDM 04)*, (pp. 331–334). IEEE Computer Society Press.

Borgelt, C. & Nürnberger, A. (2004). Fast fuzzy clustering of web page collections. In *Proc. of PKDD Workshop on Statistical Approaches for Web Mining (SAWM)*, Pisa, Italy.

Boyack, K. W., Wylie, B. N., & Davidson, G. S. (2002). Domain visualization using vxinsight for science and technology management. *Journal of the American Society for Information Science and Technologie*, *53*(9), 764–774.

Cheeseman, P. & Stutz, J. (1996). Bayesian classification (AutoClass): Theory and results. In U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, & R. Uthurusamy (Eds.), *Advances in Knowledge Discovery and Data Mining* (pp. 153–180). AAAI/MIT Press.

Chen, M.-S., Han, J., & Yu, P. S. (1996). Data mining: an overview from a database perspective. *IEEE Transaction on Knowledge and Data Engineering*, *8*(6), 866–883.

Chinchor, N. (1997). Muc-7 named entity task definition version 3.5. Technical report, NIST, ftp.muc.saic.com/pub/MUC/MUC7-guidelines.

Coupet, P. & Hehenberger, M. (1998). Text mining applied to patent analysis. In *Annual Meeting of American Intellectual Property Law Association (AIPLA) Airlington*.

crispdm and CRISP99 (1999). Cross industry standard process for data mining. `http://www.crisp-dm.org/`.

Deerwester, S., Dumais, S., Furnas, G., & Landauer, T. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Sciences*, *41*, 391–407.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistic Society, Series B*, *39*(1), 1–38.

Dhillon, I., Mallela, S., & Modha, D. (2003). Information-theoretic co-clustering. In *Proc. of the ninth ACM SIGKDD int. conf. on Knowledge Discovery and Data Mining*, (pp. 89–98). ACM Press.

Dörre, J., Gerstl, P., & Seiffert, R. (1999). Text mining: finding nuggets in mountains of textual data. In *Proc. 5th ACM Int. Conf. on Knowledge Discovery and Data Mining (KDD-99)*, (pp. 398–401)., San Diego, US. ACM Press, New York, US.

Duda, R. O. & Hart, P. E. (1973). *Pattern Classification and Scene Analysis*. New York, NY, USA: J. Wiley & Sons.

Dumais, S., Platt, J., Heckerman, D., & Sahami, M. (1998). Inductive learning algorithms and representations for text categorization. In *7th Int. Conf. on Information and Knowledge Managment*.

Fayyad, U. M., Piatetsky-Shapiro, G., & Smyth, P. (1996). Knowledge discovery and data mining: Towards a unifying framework. In *Knowledge Discovery and Data Mining*, (pp. 82–88).

Feldman, R. & Dagan, I. (1995). Kdt - knowledge discovery in texts. In *Proc. of the First Int. Conf. on Knowledge Discovery (KDD)*, (pp. 112–117).

Fickel, N. (1997). Clusteranalyse mit gemischt-skalierten merkmalen: Abstrahierung vom skalenniveau. *Allg. Statistisches Archiv*, *81*(3), 249–265.

Forgy, E. (1965). Cluster analysis of multivariate data: Efficiency versus interpretability of classification. *Biometrics*, *21*(3), 768–769.

Fox, K. L., Frieder, O., Knepper, M. M., & Snowberg, E. J. (1999). Sentinel: A multiple engine information retrieval and visualization system. *Journal of the American Society of Information Science*, *50*(7), 616–625.

Frakes, W. B. & Baeza-Yates, R. (1992). *Information Retrieval: Data Structures & Algorithms*. New Jersey: Prentice Hall.

Gaizauskas, R. (2003). An information extraction perspective on text mining: Tasks, technologies and prototype applications. `http://www.itri.bton.ac.uk/projects/euromap/TextMiningEvent/Rob_Gaizauskas.pdf`.

Gersho, A. & Gray, R. M. (1992). *Vector quantization and signal compression*. Kluwer Academic Publishers.

Good, I. J. (1965). *The Estimation of Probabilities: An Essay on Modern Bayesian Methods*. Cambridge, MA: MIT Press.

Greiff, W. R. (1998). A theory of term weighting based on exploratory data analysis. In *21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY. ACM.

Hartigan, J. (1975). *Clustering Algorithms*. John Wiley and Sons, New York.

Hastie, T., Tibshirani, R., & Friedman, J. (2001). *The Elements of Statistical Learning*. Springer.

Havre, S., Hetzler, E., Perrine, K., Jurrus, E., & Miller, N. (2001). Interactive visualization of multiple query result. In *Proc. of IEEE Symposium on Information Visualization 2001*, (pp. 105 –112). IEEE.

Hearst, M. (1999). Untangling text data mining. In *Proc. of ACL'99 the 37th Annual Meeting of the Association for Computational Linguistics*.

Hearst, M. A. & Karadi, C. (1997). Cat-a-cone: An interactive interface for specifying searches and viewing retrieval results using a large category hierarchie. In *Proc. of the 20th Annual Int. ACM SIGIR Conference*, (pp. 246–255). ACM.

Hemmje, M., Kunkel, C., & Willett, A. (1994). Lyberworld - a visualization user interface supporting fulltext retrieval. In *Proc. of ACM SIGIR 94*, (pp. 254–259). ACM.

Hidalgo, J. (2002). Tutorial on text mining and internet content filtering. Tutorial Notes Online: `http://ecmlpkdd.cs.helsinki.fi/pdf/hidalgo.pdf`.

Hofmann, T. (2001). Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning Journal*, *41*(1), 177–196.

Honkela, T. (1997). *Self-Organizing Maps in Natural Language Processing*. PhD thesis, Helsinki Univ. of Technology, Neural Networks Research Center, Espoo, Finland.

Honkela, T., Kaski, S., Lagus, K., & Kohonen, T. (1996). Newsgroup exploration with the websom method and browsing interface, technical report. Technical report, Helsinki University of Technology, Neural Networks Research Center, Espoo, Finland.

Hotho, A., Staab, S., & Stumme, G. (2003). Ontologies improve text document clustering. In *Proc. IEEE Int. Conf. on Data Mining (ICDM 03)*, (pp. 541–544).

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In Nedellec, C. & Rouveirol, C. (Eds.), *European Conf. on Machine Learning (ECML)*.

Kaufman, L. & Rousseeuw, P. (1990). *Finding groups in data: an introduction to cluster analysis*. New York: Wiley.

Keim, D. A. (2002). Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, *7*(2), 100–107.

Kim, J., Ohta, T., Tsuruoka, Y., Tateisi, Y., & Collier, N. (2004). Introduction to the bio-entity task at jnlpba. In Collier, N., Ruch, P., & Nazarenko, A. (Eds.), *Proc. Workshop on Natural Language Processing in Biomedicine and its Applications*, (pp. 70–76).

Kodratoff, Y. (1999). Knowledge discovery in texts: A definition and applications. *Lecture Notes in Computer Science*, *1609*, 16–29.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, *43*, 59–69.

Kohonen, T. (1984). *Self-Organization and Associative Memory*. Berlin: Springer-Verlag.

Kohonen, T., Kaski, S., Lagus, K., Salojärvi, J., Honkela, J., Paattero, V., & Saarela, A. (2000). Self organization of a massive document collection. *IEEE Transactions on Neural Networks*, *11*(3), 574–585.

Koster, C., Seutter, M., & Beney, J. (2001). Classifying patent applications with winnow. In *Proceedings Benelearn*, Antwerpen.

Krier, M. & Zacca, F. (2002). Automatic categorisation applications at the european patent office. *World Patent Information*, *24*(3), 187–196.

Kumar, V. & Joshi, M. (2003). What is data mining? `http://www-users.cs.umn.edu/~mjoshi/hpdmtut/sldo04.htm`.

Lafferty, J., McCallum, A., & Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. ICML*.

Lamirel, J.-C., Al Shehabi, S., Hoffmann, M., & Francois, C. (2003). Intelligent patent analysis through the use of a neural network: Experiment of multi-viewpoint analysis with the multisom model. In *ACL-2003 Workshop on Patent Corpus Processing*.

Leopold, E. & Kindermann, J. (2002). Text categorization with support vector machines. How to represent texts in input space? *Machine Learning*, *46*, 423 – 444.

Leskovec, J., Grobelnik, M., & Milic-Frayling, N. (2004). Learning sub-structures of document semantic graphs for document summarization. In *KDD 2004 Workshop on Link Analysis and Group Detection (LinkKDD)*, Seattle, Washington.

Lin, X., Marchionini, G., & Soergel, D. (1991). A selforganizing semantic map for information retrieval. In *Proc. of the 14th International ACM/SIGIR Conference on Research and Development in Information Retrieval*, (pp. 262–269)., New York. ACM Press.

Littlestone, N. (1988). Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, *2*(4), 285–318.

Lochbaum, K. E. & Streeter, L. A. (1989). Combining and comparing the effectiveness of latent semantic indexing and the ordinary vector space model for information retrieval. *Information Processing and Management*, *25*(6), 665–676.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In Le Cam, L. & Neyman, J. (Eds.), *Proc. of the fifth Berkeley Symposium*

on Mathematical Statistics and Probability, volume 1, (pp. 281–297). University of California Press.

Maitra, R. (2002). A statistical perspective on data mining. *J. Ind. Soc. Prob. Statist.*

Mani, I. & Maybury, M. T. (Eds.). (1999). *Advances in Automatic Text Summarization*. MIT Press.

Manning, C. D. & Schütze, H. (2001). *Foundations of Statistical Natural Language Processing*. Cambridge, MA: MIT Press.

McCallum, A. (2003). Efficiently inducing features of conditional random fields. In *Proc. Conf. on Uncertainty in Articifical Intelligence (UAI), 2003.*

Mendes, M. E. & Sacks, L. (2001). Dynamic knowledge representation for e-learning applications. In *Proc. of BISC International Workshop on Fuzzy Logic and the Internet (FLINT 2001)*, (pp. 176–181)., Berkeley, USA. ERL, College of Engineering, University of California.

Merkl, D. (1998). Text classification with self-organizing maps: Some lessons learned. *Neurocomputing*, *21*, 61–77.

Michelakis, E., Androutsopoulos, I., Paliouras, G., Sakkis, G., & Stamatopoulos, P. (2004). Filtron: A learning-based anti-spam filter. In *Proc. 1st Conf. on Email and Anti-Spam (CEAS 2004)*, Mountain View, CA, USA.

Mitchell, T. (1997). *Machine Learning*. McGraw-Hill.

Nahm, U. & Mooney, R. (2002). Text mining with information extraction. In *Proceedings of the AAAI 2002 Spring Symposium on Mining Answers from Texts and Knowledge Bases*.

Nigam, K., McCallum, A., Thrun, S., & Mitchell, T. (2000). Text classification from labeled and unlabeled documents using em. *Machine Learning*, *39*, 103–134.

Nürnberger, A. (2001). Interactive text retrieval supported by growing self-organizing maps. In Ojala, T. (Ed.), *Proc. of the International Workshop on Information Retrieval (IR 2001)*, (pp. 61–70)., Oulu, Finland. Infotech.

Paaß G. & deVries, H. (2005). Evaluating the performance of text mining systems on real-world press archives. In *Proc. 29th Annual Conference of the German Classification Society (GfKl 2005)*. Springer.

Pantel, P. & Lin, D. (2002). Document clustering with committees. In *Proc. of SIGIR'02, Tampere, Finland*.

Porter, M. (1980). An algorithm for suffix stripping. *Program*, 130–137.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*, 81–106.

Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proc. of IEEE*, *77*(2), 257–286.

Radev, D., Hovy, E., & McKeown, K. (2002). Introduction to the special issue on summarization. *Computational Linguistics*, *28*(4), 399–408.

Robertson, S. E. (1977). The probability ranking principle. *Journal of Documentation*, *33*, 294–304.

Rocchio, J. J. (1971). Relevance feedback in information retrieval. In G. Salton (Ed.), *The SMART Retrieval System* (pp. 313–323). Englewood Cliffs, NJ: Prentice Hall.

Roussinov, D. G. & Chen, H. (2001). Information navigation on the web by clustering and summarizing query results. *Information Processing & Management*, *37*(6), 789–816.

Salton, G., Allan, J., & Buckley, C. (1994). Automatic structuring and retrieval of large text files. *Communications of the ACM*, *37*(2), 97–108.

Salton, G. & Buckley, C. (1988). Term weighting approaches in automatic text retrieval. *Information Processing & Management*, *24*(5), 513–523.

Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, *18*(11), 613–620. (see also TR74-218, Cornell University, NY, USA).

Schapire, R. E. & Singer, Y. (1999). Improved boosting using confidence-rated predictions. *Machine Learning*, *37*(3), 297–336.

Schapire, R. E. & Singer, Y. (2000). BoosTexter: A boosting-based system for text categorization. *Machine Learning*, *39*(2/3), 135–168.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, *34*, 1–47.

Sha, F. & Pereira, F. (2003). Shallow parsing with conditional random fields. In *Proc. Human Language Technology NAACL*.

Small, H. (1999). Visualizing science by citation mapping. *Journal of the American Society for Information Science*, *50*(9), 799–813.

Sparck-Jones, K. & Willett, P. (Eds.). (1997). *Readings in Information Retrieval*. Morgan Kaufmann.

Spoerri, A. (1995). *InfoCrystal: A Visual Tool for Information Retrieval*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA.

Steinbach, M., Ertoz, L., & Kumar, V. (2003). Challenges of clustering high dimensional data. In Wille, L. T. (Ed.), *New Vistas in Statistical Physics – Applications in Econophysics, Bioinformatics, and Pattern Recognition*. Springer-Verlag.

Steinbach, M., Karypis, G., & Kumara, V. (2000). A comparison of document clustering techniques. In *KDD Workshop on Text Mining*. (see also TR 00-034, University of Minnesota, MN).

Takeuchi, K. & Collier, N. (2002). Use of support vector machines in extended named entity recognition. In *6th Conf. on Natural Language Learning (CoNLL-02)*, (pp. 119–125).

TMS05 (2005). Text mining summit conference brochure. `http://www.textminingnews.com/`.

U. Fayyad, G. Grinstein, A. W. (2001). *Information Visualization in Data Mining and Knowledge Discovery*. Morgan Kaufmann.

van Rijsbergen, C. J. (1986). A non-classical logic for information retrieval. *The Computer Journal*, *29*(6), 481–485.

Wilks, Y. (1997). Information extraction as a core language technology. In M.-T. Pazienza (Ed.), *Information Extraction*. Springer, Berlin.

Wise, J. A., Thomas, J. J., Pennock, K., Lantrip, D., Pottier, M., Schur, A., & Crow, V. (1995). Visualizing the non-visual: Spatial analysis and interaction with information from text documents. In *Proc. of IEEE Symposium on Information Visualization '95*, (pp. 51–58). IEEE Computer Society Press.

Witten, I. H., Moffat, A., & Bell, T. C. (1999). *Managing Gigabytes: Compressing and Indexing Documents and Images*. San Francisco: Morgan Kaufmann Publishers.

Edda Leopold

On Semantic Spaces

1 Introduction

This contribution gives an overview about different approaches to semantic spaces. It is not a exhaustive survey, but rather a personal view on different approaches which use metric spaces for the representation of meanings of linguistic units. The aim is to demonstrate the similarities of apparently different approaches and to inspire the generalisation of semantic spaces tailored to the representation of texts to arbitrary semiotic artefacts.

I assume that the primary purpose of a semiotic system is communication. A semiotic system $\tilde{S}$ consists of signs s. Signs fulfil a communicative function $f(s)$ within the semiotic system in order to meet the communicative requirements of system's user. There are different similarity relations between functions of signs. In its most general form a semantic space can be defined as follows:

Definition 1.1 *Let $\tilde{S}$ be a semiotic system, (S,d) a metric space and $r : \tilde{S} \to S$ a mapping from $\tilde{S}$ to S. A semantic space (S,d) is a metric space whose elements are representations of signs of a semiotic system, i.e. for each $x \in S$ there is a $s \in \tilde{S}$ such that $r(s) = x$. The inverse metric $(d(x,y))^{-1}$ quantifies some functional similarity of the signs $r^{-1}(x)$ and $r^{-1}(y)$ in $\tilde{S}$.*

Semantic spaces can quantify functional similarities in different respects. If the semiotic system is a natural language, the represented units are usually words or texts — but semantic spaces can also be constructed from other linguistic units like syllables or sentences. The constructions of Semantic spaces leads to a notion of semantic distance, which often cannot easily be made explicit. Some constructions (like the one described in section 6) yield semantically transparent dimensions.

The definition of a semantic space is not confined to linguistic units. Anything that fulfils a function in a semiotic system can be represented in a semantic space. The calculation of a semantic space often involves a reduction of dimensionality and the spaces described in this paper will be ordered with decreasing dimensionality and increasing semantic transparency. In the following section the basic notations will be introduced, that are used in the subsequent sections.

Section 3 roughly outlines the fuzzy linguistic paradigm. Sections 4 and 5 describe shortly the methods of latent semantic indexing and probabilistic latent semantic indexing. In section 6 I show how previously trained classifiers can be used in order to construct semantic spaces.

2 Notations

In order to harmonise the presentation of the different approaches I will use the following notations: A text corpus $\mathcal{C}$ consists of a number of D different textual units referred to as *documents* $d_j, j = 1, \ldots, D$. Documents can be complete texts, such as articles in a newspaper, short news as e.g. in the Reuters newswire corpus, or even short text fragments like paragraphs or text blocks of a constant length.

Each document consists of a (possibly huge) number of *terms*. The entire number of different term-types in $\mathcal{C}$ (i.e. the size of the vocabulary of $\mathcal{C}$) is denoted by W and the number of occurrences of a given term w_i in a given document d_j is denoted by $f(w_i, d_j)$. The definition of what is considered as a term may vary, terms can be lemmas, words as they occur in the running text (i.e. strings separated by blanks), tagged words as for instance in Leopold & Kindermann (2002), strings of syllables as in Paaß et al. (2002), or even a mixture of lemmas and phrases as in Neumann & Schmeier (2002). The methods described below are independent from what is considered as a term in a particular application. It is merely assumed that a corpus consists of a set of documents and each of these documents consist of a set of terms[1]. The *term-document matrix* A of $\mathcal{C}$ is a $W \times D$ matrix with W rows and D columns, which is defined as

$$A \quad = \quad (f(w_i, d_j))_{i=1,\ldots,W, j=,\ldots,D}$$

or more explicitly

$$A \quad = \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1D} \\ a_{21} & a_{22} & \ldots & a_{2D} \\ \vdots & & \ddots & \vdots \\ a_{W1} & a_{W2} & \ldots & a_{WD} \end{pmatrix}, \qquad \text{where } a_{ij} := f(w_i, d_j) \tag{1}$$

1 Actually the assumption is even weaker: the methods simply focus on the co-occurrences of documents and terms, no matter if one is contained in the other.

The entry in the ith row and the jth column of the term-document matrix indicates how often term w_i appears in document[2] d_j. The rows of A represent terms and its columns represent documents. In the so-called bag-of-words representation, document d_j is represented by the jth column of A, which is also called the word-frequency vector of document d_j and denoted by $\vec{x}_j$. The sum of the frequencies in the j-th row of A is denoted by $f(d_j)$, which is also called the *length* of document d_j. The length of corpus $\mathcal{C}$ is denoted by L. Clearly

$$f(d_j) = \sum_{i=1}^{W} f(w_i, d_j) \quad \text{and} \quad L = \sum_{j=1}^{D} f(d_j) \tag{2}$$

The ith row of A indicates how the term w_i is spread over the documents in the corpus. The rows of A are linked to the notion of *polytexty*, which was defined by Köhler (1986) as the number of contexts in which a given term w_i occurs. Köhler noted that polytexty can be operationalised by the number of *texts* the term occurs in i.e. the number of non-zero entries of the i-th row. The ith column of A is therefore called *vector of polytexty* of term w_i and the vector of the respective relative frequencies is named *distribution of polytexty*. The sum over the frequencies in the ith column, i.e. the total number of occurrences of term w_i in the corpus $\mathcal{C}$, is denoted by

$$f(w_i) = \sum_{j=1}^{D} f(w_i, d_j).$$

The polytexty measured in terms of non-zero entries in a row of the term-document matrix is also called *document-frequency* denoted as df. The so-called inverse document frequency, which was defined by Salton & McGill (1983) as $idf = (\log df)^{-1}$, is widely used in the literature on automatic text processing in order to tune term-frequencies according to the thematic relevance of a term. Other term weighting schemes like e.g. the redundancy used by Leopold & Kindermann (2002) consider the entire vector of polytexty rather than solely the number of non-zero elements. An overview about different weighting schemes is given in Manning & Schütze (1999).

Matrix transposition, subsequently indicated by a superscript $\cdot^T$, exchanges columns and rows of a matrix. So the transposed term-document matrix is

2 It should be noticed here that in many cases the term-document matrix does not contain the term-frequencies $f(w, d)$ themselves but a transformation of them like e.g. $\log f(w, d)$ or tfdif.

defined as

$$A^T = (f(w_j, d_i))_{i=1,\ldots,D, j=1,\ldots,W} = \begin{pmatrix} a^t_{11} & a^t_{12} & \cdots & a^t_{1W} \\ a^t_{21} & a^t_{22} & \cdots & a^t_{2W} \\ \vdots & & \ddots & \vdots \\ a^t_{D1} & a^t_{W2} & \cdots & a^t_{DW} \end{pmatrix},$$

$$\text{where } a^t_{ij} := f(w_j, d_i)$$

It is easy to see that the matrix transposition is inverse to itself, i.e. $(A^T)^T = A$. All algorithms presented below are symmetric in documents and terms, i.e. they can be used to estimate semantic similarity of terms as well as of documents depending on whether A or A^T is considered.

There are various measures for judging the similarity of documents. Some measures — the so-called association measures — disregard the term frequencies and just perform set-theoretical operations on the document's term sets. An example for an association measure is the *matching coefficient*, which simply counts the number of terms that two documents have in common (van Rijsbergen 1975).

Other measures take advantage from the vector space model and consider the entire term-frequency vectors of the respective documents. One of the most often used similarity measure, which is also mathematically convenient, is the cosine measure (Manning & Schütze 1999; Salton & McGill 1983) defined as

$$\cos(\vec{x}_i, \vec{x}_j) = \frac{\sum_k^W f(w_k, d_i) f(w_k, d_j)}{\sqrt{\sum_k^W f(w_k, d_i)^2 \sum_k^W f(w_k, d_j)^2}} = \frac{\vec{x}_i \cdot \vec{x}_j}{\|\vec{x}_i\| \|\vec{x}_j\|}, \quad (3)$$

which can also be interpreted as the angle between the vectors $\vec{x}_i$ and $\vec{x}_j$ or, up to centering, as the correlation between the respective discrete probability distributions.

3 Fuzzy Linguistics

> [...] the investigation of linguistic problems in general, and that of word-semantics in particular, should start with more or less pre-theoretical working hypotheses, formulated and re-formulated for continuous estimation and/or testing against observable data, then proceed to incorporate its findings tentatively in some preliminary

> theoretical set up which finally may perhaps get formalised to become part of an encompassing abstract theory. Our objective being natural language meaning, this operational approach would have to be what I would like to call *semiotic*. (Rieger 1981)

Fuzzy Linguistics (Rieger & Thiopoulos 1989; Rieger 1981, 1999) aims at a spatial representation of word meanings. I.e. the units represented in the semantic space are *words* as opposed to documents in the other approaches. However from a mathematical point of view there is no formal difference between semantic spaces that are constructed to represent documents and those which are intended to represent terms. One can transform one problem into the other by simply transposing the term-document matrix i.e. by considering A^T instead of A.

Rieger has calculated a semantic space of word meanings in two steps of abstraction, which are also implicitly incorporated in the other constructions of semantic spaces described in the sections (4) to (6). The first step of abstraction is the *α-abstraction* or more explicitly *syntagmatic abstraction* which reflects a term's usage regularities in terms of its vector of polytexty. The second abstraction step is the δ-abstraction or *paradigmatic abstraction*, which represents a word's relation to all other words in the corpus.

3.1 The Syntagmatic Abstraction

For each term w_i a vector of length W is calculated, which contains the correlations of a term's vector of polytexty with all other terms in the corpus.

$$\alpha_{i,j} = \frac{\sum_{k=1}^{D}(f(w_i,d_k) - E(f(w_i) \mid d_k))(f(w_j,d_k) - E(f(w_j) \mid d_k)}{\sqrt{\sum_{k=1}^{D}(f(w_i,d_k) - E(f(w_i) \mid d_k))^2 \sum_{k=1}^{D}(f(w_j,d_k) - E(f(w_j) \mid d_k))^2}} \tag{4}$$

where $E(f(w_i) \mid d_k) = f(w_i)\frac{f(d_k)}{L}$ is an estimator of the conditioned expectation of the frequency of term w_i in document d_j, based on all documents in the corpus. The coefficient $\alpha_{i,j}$ measures the mutual affinity ($\alpha_{i,j} > 0$) or repugnancy ($\alpha_{i,j} < 0$) of pairs of terms in the corpus (Rieger & Thiopoulos 1989).

Substituting $y_{i,j} = f(w_i,d_k) - E(f(w_i) \mid d_k)$ the centralised vector of polytexty of term w_i is defined as $\vec{y}_i = (y_{i,1},\ldots,y_{i,D})^T$. Using this definition equation (4) can be rewritten as

$$\alpha_{i,j} = \frac{\sum_k^D y_{i,k} y_{j,k}}{\sqrt{\sum_k^D y_{i,k}^2 \sum_k^D y_{j,k}^2}} = \frac{\vec{y}_i \cdot \vec{y}_j}{\|\vec{y}_i\| \|\vec{y}_j\|}, \quad (5)$$

which is the definition of the cosine distance as defined in equation (3). The difference between the α-abstraction and the cosine distance is merely that in equation (4) the centralised vector of polytexty is considered instead of the word-frequency vector in (3). Using the notion of polytexty one might say more abstractly that $\alpha_{i,j}$ is the correlation coefficient of the polytexty distributions of the types w_i and w_j on the texts in the corpus.

Syntagmatic abstraction realised by equation (4) refers to usage regularities in terms of co-occurrences in the same document. Documents in Rieger's works were in general short texts, like e.g. newspaper texts (Rieger 1981; Rieger & Thiopoulos 1989) or small textual fragments (Rieger 2002). This means that the syntagmatic abstraction solely relies on the distribution of polytexty of the respective terms.

In principle however the approach can be generalised regarding various types of generalised syntagmatic relations. Note that documents were defined as arbitrary disjoint subsets of a corpus. The underlying formal assumption was simply that there is a co-occurrence structure of documents and terms, which is represented in the term-document matrix. Consider for instance a syntactically tagged corpus. In such a corpus documents might be defined e.g. as a set of terms that all carry the same tag. The corresponding "distributions of polytexty" would describe how a term is used in different parts-of-speech and the syntagmatic abstraction $\alpha_{i,j}$ would measure the similarity of w_i and w_j in terms of part-of-speech membership.

3.2 The Paradigmatic Abstraction

The α-abstraction measures the similarities of the distribution of polytexty over all terms in the corpus. The absolute value of the similarities, however, is not solely a property of the terms themselves, but also of the corpus as a whole. That is if the corpus is confined to a small thematic domain, the documents will be more similar than in the case of a corpus that covers a wide range of themes. In order to attain a paradigmatic abstraction, which abstracts away from the thematic coverage of the corpus, the Euclidean distances to all words in the corpus are summed. This is the δ-abstraction (Rieger 1981; Rieger & Thiopoulos 1989) given by:

$$\delta(y_i, y_j) = \sqrt{\sum_{n=1}^{W} (\alpha_{i,n} - \alpha_{j,n})^2}; \qquad \delta \in [0; 2\sqrt{W}] \tag{6}$$

The δ-abstraction compensates the effect of the corpus' coverage on α. The similarity vector of each term is related to the similarity vectors of all other terms in the corpus. In this way the paradigmatic structure in the corpus is evaluated in the sense that every term is paradigmatically related to each other since every term can equally be engaged in a *occurs-in-document* relation.

So the vector y_i, is mapped to a vector $(\delta(i,1) \ldots \delta(i,W))$, which contains the Euclidean distance of x_i's α to all other αs generated by the corpus and is interpreted as meaning point in a semantic space (Rieger 1988). Rieger concludes that in this way a semantic representation is attained that represents the numerically specified generalised paradigmatic structure that has been derived for each abstract syntagmatic usage regularity against all other in the corpus (Rieger 1999).

Goebl (1991) uses another measurement to anchor similarity measurements of linguistic units (in his case dialectometric data sets) for the completely different purpose of estimating the centrality of dialects in a dialectal network. Let $\alpha_{i,j}$ denote the similarity of dialect x_i and x_j, and let W denote the number of dialects in the network. The centrality of x_i is given by:

$$\gamma(x_i) = \sum_{n=1}^{W} \left(\alpha_{i,n} - \frac{1}{W} \sum_{k=1}^{W} \alpha_{i,k} \right)^3 \tag{7}$$

He argues

> The skewness of a similarity distribution has a particular *linguistic* meaning. The more symmetric a similarity distribution is, the greater the centrality of the particular local dialect in the whole network.(Goebl 1991)

Goebl uses (7) in order to calculate the centrality of a local dialect from the matrix $(\alpha_{i,j})_{i,j}$ of similarity measures between pairs of dialects in the network. These centrality measures are employed to draw a choropleth map of the dialectal network. Substituting the delta abstraction in (6) by the skewness in (7) would result in a measure for the centrality of a term in a term-document network: the more typical a term's usage in the corpus the larger the value of γ. Such a measure could be used as a term-weighting scheme.

Rieger's construction of a semantic space does *not* lead to a reduction of dimensionality. This was not his aim. The meaning of a term is represented by a high-dimensional vector and thus demonstrates the complexity of meaning structures in natural language. Rieger's idea to compute semantic relations from a term-document matrix and represent semantic similarities as distances in a metric space has aspects in common with pragmatically oriented approaches like e.g. latent semantic analysis. The measures of the $\alpha_{i,j}$ can be written in a more condensed way as

$$B^* = A^*(A^*)^T = (\alpha_{i,j})_{i,j=1,...W} \tag{8}$$

B^* is a $W \times W$-matrix which represents the similarity of the words w_i and w_j in terms of their distribution of polytexty. The semantic similarity between words is calculated here in a way similar to the semantic similarity between words in latent semantic indexing, which is described in the next section. The *similarity matrix* $B^* = A^*(A^*)^T$ however is calculated in a slightly different way. The entries of A^* are $y_{i,j} = f(w_i, d_k) - E(f(w_i) \mid d_k)$ rather than the term frequencies $f(w_i, d_j)$ themselves, as can be seen from equation (4).

More advanced techniques within the fuzzy linguistic paradigm (Mehler 2002) extend the concept of the semantic space to the representation of texts. The respective computations, however, are complicated and exceed the scope of this paper.

Fuzzy linguistics aims at a numerical representation of the meaning of terms. Thus the paradigmatic abstraction in equation (6) does not involve a reduction of dimensionality, in contrast to the principal component analysis that is performed in the paradigmatic abstraction step in latent semantic analysis. There is however a close formal relationship.

4 Latent Semantic Analysis

> In essence, and in detail, it [latent semantic analysis] assumes that the psychological similarity between any two words is reflected in the way they co-occur in small subsamples of language. (Landauer & Dumais (1997); Words in square brackets added by the author.)

In contrast to fuzzy linguistics latent semantic analysis (LSA) is interested in the semantic nearness of *documents* rather than of words. The method however is symmetric and can be applied to the similarity of words as well.

LSA projects Document frequency vectors into a low dimensional space calculated using the frequencies of word occurrence in each document. The relative distances between these points are interpreted as distances between the topics of the documents and can be used to find related documents, or documents matching some specified query (Berry et al. 1995). The underlying technique of LSA was chosen to fulfil the following criteria:

1. To represent the underlying semantic structure a model with sufficient power is needed. Since the right kind of alternative is unknown the power of the model should be variable.

2. Terms and documents should both be explicitly represented in the model.

3. The method should be computationally tractable for large data sets. Deerwester et al. concluded that the only model which satisfied all these three criteria was the singular value decomposition (SVD), which is a well known technique in linear algebra (Deerwester et al. 1990).

4.1 Singular Value Decomposition

Let A be a term-document matrix as defined in section (2) with rank[3] r. The singular value decomposition of A is given by

$$A = U\Sigma V, \tag{9}$$

where $\Sigma = diag(\sigma_1, \ldots, \sigma_r)$ is a diagonal matrix with ordered diagonal elements $\sigma_1 > \ldots > \sigma_r$,

$$U = \begin{pmatrix} u_{11} & u_{12} & \ldots & u_{1r} \\ u_{21} & u_{22} & \ldots & u_{2r} \\ \vdots & & \ddots & \vdots \\ u_{W1} & u_{W2} & \ldots & u_{Wr} \end{pmatrix}$$

is a $W \times r$-matrix with orthonormal columns and

$$V = \begin{pmatrix} v_{11} & v_{12} & \ldots & v_{1r} \\ v_{21} & v_{22} & \ldots & v_{2r} \\ \vdots & & \ddots & \vdots \\ v_{r1} & v_{r2} & \ldots & v_{rr} \end{pmatrix}$$

3 In practice one can assume $r = D$, since it is very unlikely that there are two documents in the corpus with linear dependent term-frequency vectors

is a $r \times r$-matrix with orthonormal rows. The diagonal elements $\sigma_1, \ldots, \sigma_r$ of the matrix Σ are singular values of A. The singular value decomposition can equivalently be written as an eigen-value decomposition of the similarity matrix

$$B = AA^T \tag{10}$$

Note that U and V are orthonormal matrices therefore $UU^T = I$ and $VV^T = I$, where I is the neutral element of matrix-multiplication. According to (9) the singular value decomposition of the transposed term-document matrix A^T is obtained as $A^T = V^T \Sigma U^T$. Hence $AA^T = U\Sigma VV^T \Sigma U^T = U\Sigma^2 U^T$ which is the eigen-value decomposition of AA^T with eigen-values $\sigma_1^2, \ldots, \sigma_r^2$. Term frequency vectors are mapped to the latent space of artificial concepts by multiplication with $U\Sigma$, i.e. $\vec{x} \rightarrow \vec{x}^T U\Sigma$. Each of the r dimensions of the latent space may be thought of as an artificial concept, which represents common meaning components of different words and documents.

4.2 Deleting the Smallest Singular Values

A reduction of dimensionality is achieved by deleting the smallest singular values corresponding to the less important concepts in the corpus. In so doing latent semantic analysis reduces the matrix A to a smaller K-dimensional ($K < r$) matrix

$$A_K = U_K \Sigma_K V_K, \tag{11}$$

where U_K and V_K are obtained from U and V in equation (9) by deleting respectively columns and/or rows $K+1$ to r and the diagonal matrix is reduced to $\Sigma_K = diag(\sigma_1, \ldots, \sigma_K)$. The mapping of a term-frequency vector to the reduced latent space is now performed by $\vec{x} \rightarrow \vec{x}^T U_K \Sigma_K$. It has been found that $K \approx 100$ is a good value to chose for K (Landauer & Dumais 1997).

LSA leads to vectors with few zero entries and to a reduction of dimensionality (k instead of W) which results in a better geometric interpretability. This implies that it is possible to compute meaningful association values between pairs of documents, even if the documents do not have any terms in common.

4.3 SVD Minimises Euclidean Distance

Truncating the singular value decomposition as described in equation (11) projects the data onto the best-fitting affine subspace of a specified dimension K. It is a well-known theoretical result in linear algebra, that there is no matrix X

with $rank(X) < K$ that has a smaller Frobenius distance to the original matrix A i.e. A_K minimises

$$\|A - A_K\|_F = \sum_{i,j}^{K} (a_{i,j} - a_{i,j}^K)^2. \tag{12}$$

Interestingly Rieger's δ-abstraction in equation (6) yields a nice interpretation of this optimality statement. The reduction of dimensionality performed by latent semantic analysis is achieved in such a way that it optimally preserves the inherent meaning (i.e. the sum of the $\delta(x_i, x_j)$). That is the meaning points in the Rieger's δ-space are changed to the minimal possible extent. Another parallel between fuzzy linguistics and LSA is that equation (4) and the corresponding matrix notation of $\alpha_{i,j}$ in equation (8) coincide withe the similarity matrix in equation (10). The only difference is that the entries of A and A^* are defined in a different way. Using Rieger's terminology one may call equation (10) a syntagmatic abstraction, because it reflects the usage regularities in the corpus. The singular value decomposition is then the paradigmatic abstraction, since it abstracts away from the paradigmatic structure of the language's vocabulary which consists of synonymy and polysemy relationships.

One objection to latent semantic indexing is that, along with all other least-square methods, the property of minimising the Frobenius distance makes it suited for normally distributed data. The normal distribution however is unsuitable to model term frequency counts. Other distributions like Poisson or negative binomial are more appropriate for this purpose (Manning & Schütze 1999).

Alternative methods have therefore been developed (Gous 1998), which assume that the term frequency vectors are multinomially distributed and therefore agree with well corroborated models on word frequency distribution developed by Chitashvili and Baayen (Chitashvili & Baayen 1993). Probabilistic Latent Semantic Analysis has advanced further in this direction.

5 Probabilistic Latent Semantic Analysis

Whereas latent semantic analysis is based on counts of co-occurrences and uses the singular value decomposition to calculate the mapping of term-frequency vectors to a low-dimensional space, probabilistic latent semantic analysis (see Hofmann & Puzicha (1998); Hofmann (2001)) is based on a probabilistic framework and uses the maximum likelyhood principle. This results in a better lin-

guistic interpretability and makes probabilistic latent semantic analysis (PLSA) compatible with the well-corroborated multinomial model of word frequency distributions.

5.1 The Multinomial Model

The assumption that the occurrences of different terms in the corpus are stochastically independent allows to calculate the probability of a given term frequency vector $\vec{x}_j = (f(w_1, d_j), \ldots, f(w_W, d_j))$ according to the multinomial distribution (see Chitashvili & Baayen (1993); Baayen (2001)):

$$p(\vec{x}_j) = \frac{f(d_j)}{\prod_{i=1}^{W} f(w_i, d_j)!} \prod_{i=1}^{W} p(w_i, d_j)^{f(w_i, d_j)}$$

If it is further assumed that the term-frequency vectors of the documents in the corpus are stochastically independent, the probability to observe a given term-document matrix is

$$p(A) = \prod_{j=1}^{D} \frac{f(d_j)}{\prod_{i=1}^{W} f(w_i, d_j)!} \prod_{i=1}^{W} p(w_i, d_j)^{f(w_i, d_j)} \tag{13}$$

5.2 The Aspect Model

In order to map high-dimensional term-frequency vectors to a limited number of dimensions PLSA uses a probabilistic framework, called aspect model. The aspect model is a latent variable model which associates an unobserved class variable $z_k, k = 1, \ldots, K$, with each observation an observation being the occurrence of a word in a particular document. The latent variables z_k can be thought of as artificial concepts like the latent dimensions in LSA. Like in LSA the number of artificial concepts K has to be chosen by the experimenter. The following probabilities are introduced: $p(d_j)$ denotes the probability that a word occurrence will be observed in a particular document d_i, $p(w_i \mid z_k)$ denotes the conditional probability of a specific term conditioned on the latent variable z_k (i.e. the probability of term w_i given the thematic domain z_k), and finally $p(z_k \mid d_j)$ denotes a document-specific distribution over the latent variable space i.e. the distribution of artificial concepts in document d_j. A generative model for word/document co-occurrences is defined as follows:

(1) select a document d_j with probability $p(d_j)$,

(2) pick a latent class z_k with probability $p(z_k|d_j)$, and

(3) generate word w_j with probability $p(w_i|z_k)$ (Hofmann 2001).

Since the aspects are latent variables which cannot be observed directly, the conditioned probability $p(w_i \mid d_j)$ has to be calculated as the sum of the possible aspects:

$$p(w_i|d_j) \quad = \quad \sum_{k=1}^{K} p(w_i|z_k)p(z_k|d_j) \tag{14}$$

This implies the assumption, that the conditioned probability of occurrence of aspect z_k in document d_j is independent from the conditioned probability that term w_i is used given that aspect z_k is present (Hofmann 2001).

In order to find the optimal probabilities $p(w_i|z_k)$ and $p(z_k|d_j)$, maximizing the probability of observing a given term-document matrix, the maximum likelihood principle is applied. The multinomial coefficient in equation (13) remains constant when the probabilities $p(w_i, d_j)$ are varied. It can therefore be omitted for the calculation of the likelihood function, which is then given as

$$\mathcal{L} = \sum_{j=1}^{D} \sum_{i=1}^{W} f(w_i, d_j) \log p(w_i, d_j)$$

Using the definition of the conditioned probabilities $p(w_i, d_j) = p(d_j)p(w_i \mid d_j)$ and inserting equation (14) yields

$$\mathcal{L} = \sum_{j=1}^{D} \sum_{i=1}^{W} \left(f(w_i, d_j) \log \Big(p(d_j) \cdot \sum_{k=1}^{K} p(w_i \mid z_k) p(z_k \mid d_j) \Big) \right)$$

Using the additivity of the logarithm and factoring in $f(w_i, d_j)$ gives

$$\mathcal{L} = \sum_{j=1}^{D} \left(\sum_{i=1}^{W} f(w_i, d_j) \log p(d_j) + \sum_{i=1}^{W} f(w_i, d_j) \log \sum_{k=1}^{K} p(w_i \mid z_k) p(z_k \mid d_j) \right)$$

Since $\sum_i f(w_i, d_j) = f(d_j)$ factoring out $f(d_j)$ finally leads to the likelihood function

$$\mathcal{L} = \sum_{j=1}^{D} f(d_j) \left(\log p(d_j) + \sum_{i=1}^{W} \frac{f(w_i, d_j)}{f(d_j)} \log \sum_{k=1}^{K} p(w_i \mid z_k) p(z_k \mid d_j) \right) \tag{15}$$

which has to be maximised with respect to the conditional probabilities involving the latent aspects z_k. Maximisation of (15) can be achieved using the EM-algorithm, which is a standard procedure for maximum likelihood estimation in latent variable models (Dempster et al. 1977). The EM-algorithm works in two steps that are iteratively repeated (see e.g. Mitchell (1997) for details).

Step 1 In the first step (the expectation step) the expected value $E(z_k)$ of the latent variables is calculated, assuming that the current hypothesis h_1 holds.

Step 2 In a second step (the maximisation step) a new maximum likelihood hypothesis h_2 is calculated assuming that the latent variables z_k equal their expected values $E(z_k)$ that have been calculated in the expectation step. Then h_1 is substituted by h_2 and the algorithm is iterated.

In the case of PLSA the the EM-algorithm is employed as follows (see Hofmann (2001) for details): To initialise the algorithm generate $W \cdot K$ random values for the probabilities $p(w_i \mid z_k)$ and $D \cdot K$ random values for the probabilities $p(z_k \mid d_j)$ such that all probabilities are larger than zero and fulfil the conditions $\sum_{i,k} p(w_i \mid z_k) = 1$ and $\sum_{j,k} p(z_k \mid d_j) = 1$ respectively. The expectation step can be obtained from equation (15) by applying Bayes' formula:

$$p(z_k \mid w_i, d_j) = \frac{p(w_i \mid z_k) p(z_k \mid d_j)}{\sum_{k=1}^{K} p(w_i \mid z_k) p(z_k \mid d_j)} \tag{16}$$

In the maximization step the probability $p(z_k \mid w_i, d_j)$ is used to calculate the new conditioned probabilities

$$p(w_i \mid z_k) = \frac{\sum_{j=1}^{N} f(w_i, d_j) p(z_k \mid w_i, d_j)}{\sum_{k=1}^{K} \sum_{j=1}^{D} f(w_i, d_j)(z_k \mid w_i, d_j)} \tag{17}$$

and

$$p(z_k \mid d_j) = \frac{\sum_{i=1}^{W} f(w_i, d_j) p(z_k \mid w_i, d_j)}{f(d_j)}, \tag{18}$$

Then the conditioned probabilities $p(z_k|d_j)$ and $p(w_i|z_k)$ calculated from equation (17) and (18) are inserted into equation (16) to perform the next iteration. The iteration is stopped when a stationary point of the likelihood function is achieved. The probabilities $p(z_k \mid d_j), k = 1, \ldots, K$, uniquely define for each document a $K-1$-dimensional point in continuous latent space.

It is reported that PLSA outperforms LSA in terms of perplexity reduction. Notably PLSA allows to train latent spaces with a continuous increase in performance, in contrast to LSA where the model perplexity increases when a certain number of latent dimensions is exceeded. In PLSA the number of latent dimensions may even exceed the rank of the term-document matrix (Hofmann 2001).

The main difference between LSA and PLSA is the optimisation criterion for the mapping to the latent space, which is defined by $U\Sigma$ and $p(z_k \mid d_j)$ respectively. LSA minimises the least square criterion in equation (12) and thus implicitly assumes an additive Gaussian noise on the term-frequency data. PLSA in contrast assumes multinomially distributed term-frequency vectors and maximises the likelihood of the aspect model. It is therefore in accordance with linguistic word frequency models. One disadvantage of PLSA is, that the EM-algorithm like most iterative algorithms converges only locally. Therefore the solution need not be a global optimum, in contrast to LSA which uses an algebraic solution and ensures global optimality.

6 Classifier Induced Semantic Spaces

> [...] problems, in which the task is to classify examples into one of a discrete set of possible categories, are often referred to as *classification problems*.(Mitchell 1997)

The main problem in PLSA approach was to find the latent aspect variables z_k and calculate the corresponding conditioned probabilities $p(w_i|z_k)$ and $p(z_k|d_j)$. It was assumed that the latent variables correspond to some artificial concepts. It was impossible however to specify these concepts explicitly. In the approach described below, the aspect variables can be interpreted semantically. Prerequisite for such a construction of a semantic space is a semantically annotated *training corpus*. Such annotations are usually done manually according to explicitly

defined annotation rules. An example of such a corpus is e.g. the news data of the German Press Agency (dpa) which is annotated according to the categories of the International Press Telecommunications Council (IPTC). These annotations inductively define the concepts z_k, or the dimensions, of the semantic space. A *classifier induced semantic space* (CISS) is generated in two steps: In the *training step* classification rules $\vec{x}_j \rightarrow z_k$ are inferred from the training data. In the *classification step* these decision rules are applied to possibly unannotated documents.

This construction of a semantic space is especially useful for practical applications because (1) the space is low-dimensional (up to dozens of dimensions) and thus can easily be visualised, (2) the space's dimension possesses a well defined semantic interpretation, and (3) the space can be tailored to the special requirements of a specific application. The disadvantage of classifier induced semantic spaces (CISS) is that they rely on *supervised* classifiers. Therefore manually annotated training data is required.

Classification algorithms often use an internal representation of degree of membership. They internally calculate how much a given input vector $\vec{x}$ belongs to a given class z_k. This internal representation of degree of membership can be exploited to generate a semantic space.

A Support Vector Machine (SVM) is a supervised classification algorithm that recently has been applied successfully to text classification tasks. SVMs have proven to be an efficient and accurate text classification technique (Dumais et al. 1998; Drucker et al. 1999; Joachims 1998; Leopold & Kindermann 2002). Therefore Support Vector Machines appears to be the best choice for the construction of a semantic space for textual documents.

6.1 Using an SVM to Quantify the Degree of Membership

Like other supervised machine learning algorithms, an SVM works in two steps. In the first step — the *training* step — it learns a decision boundary in input space from preclassified training data. In the second step — the *classification* step — it classifies input vectors according to the previously learned decision boundary. A *single* support vector machine can only separate *two* classes — a positive class ($y = +1$) and a negative class ($y = -1$). This means that for each of the K classes z_k a new SVM has to be trained separating z_k from all other classes.

In the training step the following problem is solved: Given is a set of training examples $S_\ell = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_\ell, y_\ell)\}$ of size $\ell \leq W$ from a fixed but unknown distribution $p(\vec{x}, y)$ describing the learning task. The term-frequency

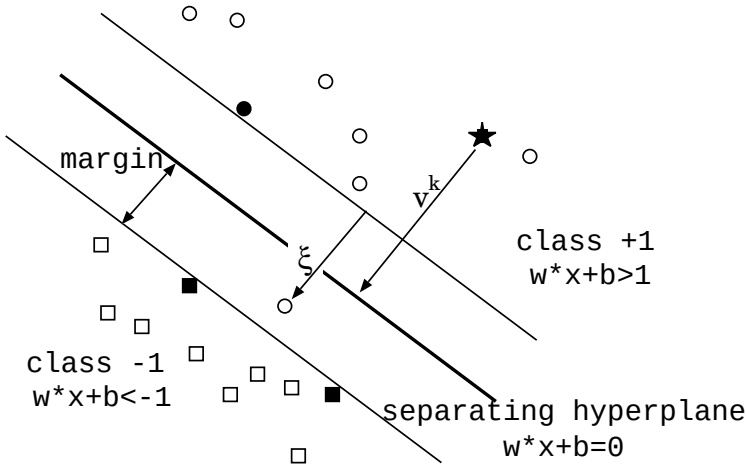

Figure 1: Generating a CISS with a support vector machine. The SVM algorithm seeks to maximise the margin around a hyperplane that separate a positive class (marked by circles) from a negative class (marked by squares). Once an SVM is trained, $v^k = \vec{w}^k\vec{x} + b$ is calculated in the classification step. The quantity v^k measures the rectangular distance between the point marked by a star and the hyperplane. It can be used to generate a CISS.

vectors $\vec{x}_i$ represent documents and $y_i \in \{-1, +1\}$ indicates whether a document has been annotated as belonging to the positive class or not. The SVM aims to find a decision rule $h_{\mathcal{L}} : \vec{x} \to \{-1, +1\}$ based on S_ℓ that classifies documents as accurately as possible.

The hypothesis space is given by the functions $f(\vec{x}) = \text{sgn}(\vec{w}\vec{x} + b)$, where $\vec{w}$ and b are parameters that are learned in the training step and which determine the class separating hyperplane. Computing this hyperplane is equivalent to solving the following optimisation problem (Vapnik 1998; Joachims 2002):

$$\begin{aligned} \text{minimise:} \quad & V(\vec{w}, b, \vec{\xi}) = \frac{1}{2}\vec{w}\vec{w} + C\sum_{i=1}^{\ell}\xi_i \\ \text{subject to:} \quad & \forall_{i=1}^{\ell} : y_i(\vec{w}\vec{x} + b) \geq 1 - \xi_i \\ & \forall_{i=1}^{\ell} : \xi_i \geq 0 \end{aligned}$$

The constraints require that all training examples are classified correctly allowing for some outliers, symbolised by the slack variables ξ_i. If a training example lies on the wrong side of the hyperplane, the corresponding ξ_i is greater or equal to 0. The factor C is a parameter that allows one to trade off training error against model complexity. Instead of solving the above optimization problem directly, it is easier to solve the following dual optimisation problem (Vapnik 1998; Joachims 2002).

$$\begin{aligned} \text{minimise:} \quad & W(\vec{\alpha}) = -\sum_{i=1}^{\ell} \alpha_i + \frac{1}{2} \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} y_i y_j \alpha_i \alpha_j \vec{x}_i \vec{x}_j \\ \text{subject to:} \quad & \sum_{i=1 \underline{\leq} \alpha_i \leq C}^{\ell} y_i \alpha_i = 0 \end{aligned} \tag{19}$$

All training examples with $\alpha_i > 0$ at the solution are called support vectors. The support vectors are situated right at the margin (see the solid squares and the circle in figure (1)) and define the hyperplane. The definition of a hyperplane by the support vectors is especially advantageous in high dimensional feature spaces because a comparatively small number of parameters — the αs in the sum of equation (19) — is required.

In the classification step an unlabeled term-frequency vector is estimated to belong to the class

$$\hat{y} = \text{sgn}(\vec{w}\vec{x} + b) \tag{20}$$

Heuristically the estimated class membership $\hat{y}$ corresponds to whether $\vec{x}$ belongs on the lower or upper side of the decision hyperplane. Thus estimating the class membership by equation (20) consists of a loss of information since only the algebraic sign of right-hand term is evaluated. However the value of $v = \vec{w}\vec{x} + b$ is a real number and can be used in order to create a real valued semantic space, rather than just to estimate if $\vec{x}$ belongs to a given class or not.

6.2 Using Several Classes to Construct a Semantic Space

Suppose there are several, say K, classes of documents. Each document is represented by an input vector $\vec{x}_j$. For each document the variable $y_j^k \in \{-1, +1\}$ indicates whether $\vec{x}_j$ belongs to the k-th class ($k = 1, \ldots, K$) or not. For each class $k = 1, \ldots, K$ an SVM can be learned which yields the parameters $\vec{w}^k$ and

b^k. After the SVMs have been learned, the classification step (equation (20)) can be applied to a (possibly unlabeled) document represented by $\vec{x}$ resulting in a K-dimensional vector $\vec{v}$, whose kth component is given by $v^k = \vec{w}^k \cdot \vec{x} + b^k$ The component v^k quantifies how much a document belongs to class k. Thus the document represented by the term frequency vector $\vec{x_j}$ is mapped to the K-dimensional vector in the classifier induced semantic space. Each dimension in this space can be interpreted as the membership degree of the document to each of the K classes.

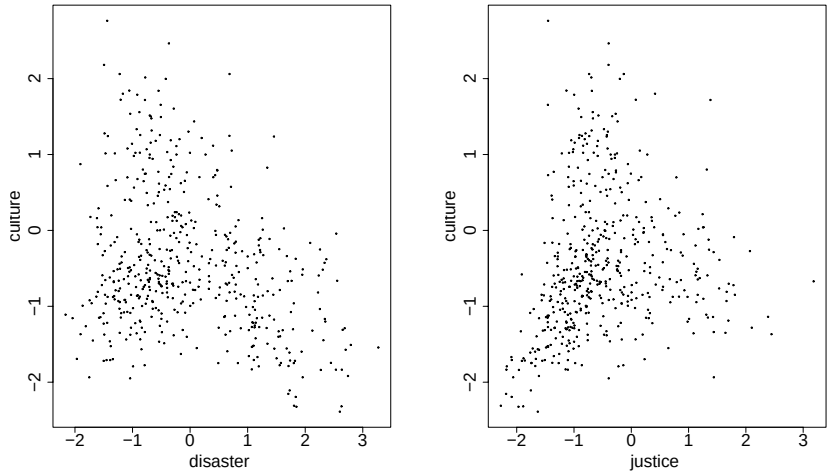

Figure 2: A classifier induced semantic space. 17 classifiers have been trained according to the highest level of the IPTC classification scheme. The projection to two dimensions "culture" and "disaster" is displayed on the right, and the projection to "culture" and "justice" on the left. The calculation is based on 68778 documents from the "Basisdienst" of the German Press Agency (dpa) July-October 2000.

The relation between PLSA and CISS is given by the latent variable z_k. In the context of CISS the latent variable z_k is interpreted as the thematic domain, in accordance with semantic annotations in the corpus. Statistical learning theory assumes, that each class k is learnable because there is an underlying conditional

distribution $p(\vec{x}_j \mid z_k)$, which reflects the special characteristics of the class z_k. The classification rules that are learned from the training data minimise the expected error. In PLSA the aspect variables are not previously defined. The conditioned probabilities $p(w_i \mid z_k)$ and $p(z_k \mid \vec{x}_j)$ are chosen in such a way that they maximise the likelihood of the multinomial model.

6.3 Graphical Representation of a CISS

Self-organising Maps (SOM) were invented in the early 80s (Kohonen 1980). They use a specific neural network architecture to perform a recursive regression leading to a reduction of the dimension of the data. For practical applications SOMs can be considered as a distance preserving mapping from a more than three-dimensional space to two-dimensions. A description of the SOM algorithm and a thorough discussion of the topic is given by Kohonen (1995).

Figure 3 shows an example of a SOM visualising the semantic relations of news messages. SVMs for the four classes 'culture', 'economy', 'politics', and 'sports' were trained by news messages from the 'Basisdienst' of the German Press Agency (dpa) April 2000. Classification and generation of the SOM was performed for the news messages of the first 10 days of April. 50 messages were selected at random and displayed as white crosses. The categories are indicated by different grey tone. Then the SOM algorithm is applied (with 100×100 nodes using Euclidean metric) in order to map the four-dimensional document representations to two dimensions admitting a minimum distortion of the distances. The grey tone indicates the topic category. Shadings within the categories indicate the confidence of the estimated class membership (dark = low confidence, bright = high confidence).

It can be seen that the change from sports (15) to economy (04) is filled by documents which cannot be assigned confidently to either classes. The area between politics (11) and economy (04), however, contains documents, which definitely belong to both classes. Note that classifier induced semantic spaces go beyond a mere extrapolation of the annotations found in the training corpus. It gives an insight into how typical a certain document is for each of the classes. Furthermore Classifier induced semantic spaces allow one to reveal previously unseen relationships between classes. The bright islands in area 11 on Figure 3 show, for example, that there are messages classified as economy which surely belong to politics.

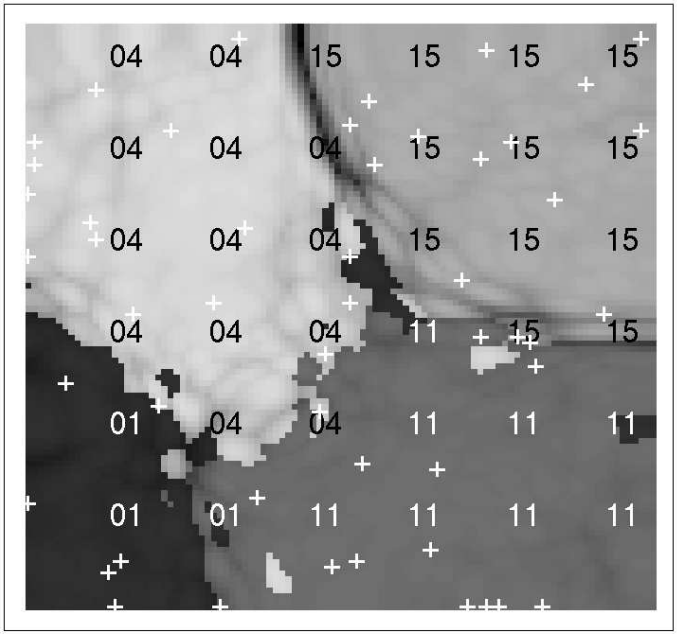

Figure 3: Self-organising map of a classifier induced semantic space. 4 classifiers have been trained according to the highest level of the IPTC classification scheme. The shadings and numbers indicate the "true" topic annotations of the news messages. 01: culture, 04: economy, 11: politics, 15: sports. (The figure was taken from Leopold et al. (2004)).

7 Conclusion

Fuzzy Linguistics, LSA, PLSA, and CISS map documents to the semantic space in a different manner. Fuzzy Lintuistics computes a vector for each word which consists of the cosine distances to every other word in the corpus. Then it calculates the Euclidean Dinstances between the vectors which gives the meaning point. Documents are represented by summing up the meaning points of the document's words.

In the case of LSA the representation of the document in the semantic space is achieved by matrix multiplication: $d_j \rightarrow \vec{x}_j^T U_K \Sigma_K$. The dimensions of the semantic space correspond to the K largest eigen-values of the similarity matrix AA^T. The projection employed by LSA always leads to a global optimum in terms of the Euclidean distance between A and A_k.

PLSA maps a document to the vector of the conditional probabilities, which indicate how probable aspect z_k is, when document d_j is selected: $d_j \rightarrow (p(z_1 \mid d_j), \ldots, p(z_K \mid d_j))$. The probabilities are derived from the aspect model using the maximum likelihood principle and the assumption of multinomially distributed word frequency distributions. The the likelihood function is maximised using the EM-algorithm, which is an iterative algorithm that leads only to a local optimum.

CISS requires a training corpus of documents annotated according to their membership of classes z_k. The classes have to be explicitly defined by the human annotation rules. For each class z_k a classifier is trained, i.e. parameters $\vec{w}^k$ and b^k are calculated from the training data. For each document d_j the quantities $v^k = \vec{w}^k \cdot \vec{x} + b^k$ are calculated, which indicate how much d_j belongs the previously learned classes z_k. The mapping of document d_j to the semantic space is defined as $d_j \rightarrow (v_1, \ldots v_K)$. The dimensions can be interpreted according to the annotation rules.

8 Acknowledgements

This study is part of the project InDiGo which is funded by the German ministry for research and technology (BMFT) grant number 01 AK 915 A.

References

Baayen, H. (2001). *Word Frequency Distributions*. Dordrecht: Kluwer.

Berry, M. W., Dumais, S. T., & O'Brien, G. W. (1995). Using linear algebra for intelligent information retrieval. *SIAM Review*, *37*(4), 573–595.

Chitashvili, R. J. & Baayen, R. H. (1993). Word frequency distributions. In G. Altmann & L. Hřebíček (Eds.), *Quantitative Text Analysis* (pp. 54–135). Trier: wvt.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshmann, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, *41*(6), 391–407.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, B*, *39*, 1–38.

Drucker, H., Wu, D., & Vapnik, V. (1999). Support vector machines for spam categorization. *IEEE Transactions on Neural Networks*, *10*, 1048–1054.

Dumais, S., Platt, J., Heckerman, D., & Sahami, M. (1998). Inductive learning algorithms and representations for text categorization. In *Proceedings of the ACM-CIKM*, (pp. 148–155).

Goebl, H. (1991). Dialectrometry: A short overview of the principles and practice of quantitative classification of linguistic atlas data. In Köhler, R. & Rieger, B. B. (Eds.), *Contributions to quantitative linguistics, Proceedings of the first international conference on quantitative linguistics*, (pp. 277–315)., Dordrecht. Kluwer.

Gous, A. (1998). *Exponential and Spherical Subfamily Models*. PhD thesis, Stanford University.

Hofmann, T. (2001). Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, *42*, 177–196.

Hofmann, T. & Puzicha, J. (1998). Statistical models for co-occurrence data. A.I. Memo No. 1625., Massachusetts Institute of Technology.

Joachims, T. (1998). Text categorization with support vector machines: learning with many relevant features. In *Proceedings of the Tenth European Conference on Machine Learning (ECML 1998)*, (pp. 137–142)., Berlin. Springer.

Joachims, T. (2002). *Learning to classify text using support vector machines*. Boston: Kluwer.

Köhler, R. (1986). *Zur linguistischen Synergetik: Struktur und Dynamik der Lexik*. Bochum: Brockmeyer.

Kohonen, T. (1980). *Content-addressable Memories*. Berlin: Springer.

Kohonen, T. (1995). *Self-Organizing Maps*. Berlin: Springer.

Landauer, T. K. & Dumais, S. T. (1997). A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, *104*(2), 211–240.

Leopold, E. & Kindermann, J. (2002). Text categorization with support vector machines. How to represent texts in input space? *Machine Learning*, *46*, 423–444.

Leopold, E., May, M., & Paaß, G. (2004). Data mining and text mining for science and technology research. In H. F. Moed, W. Glänzel, & U. Schmoch (Eds.), *Handbook of Quantitative Science and Technology Research* (pp. 187–214). Dordrecht: Kluwer.

Manning, C. D. & Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. Cambridge, Massachusetts: MIT Press.

Mehler, A. (2002). Hierarchical orderings of textual units. In *Proceedings of the 19th International Conference on Computational Linguistics, COLING'02, Taipei*, (pp. 646–652)., San Francisco. Morgan Kaufmann.

Mitchell, T. M. (1997). *Machine Learning*. New York: McGraw-Hill.

Neumann, G. & Schmeier, S. (2002). Shallow natural language technology and text mining. *Künstliche Intelligenz*, *2(2)*, 23–26.

Paaß, G., Leopold, E., Larson, M., Kindermann, J., & Eickeler, S. (2002). SVM classification using sequences of phonemes and syllables. In *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD), Helsinki*, (pp. 373–384)., Berlin. Springer.

Rieger, B. B. (1981). Feasible fuzzy semantics. On some problems of how to handle word meaning empirically. In H. Eikmeyer & H. Rieser (Eds.), *Words, Worlds, and Contexts. New Approaches in Word Semantics (Research in Text Theory 6)* (pp. 193–209). Berlin: de Gruyter.

Rieger, B. B. (1988). Definition of terms, word meaning, and knowledge structure. On some problems of semantics from a computational view of linguistics. In Czap, H. & Galinski, C. (Eds.), *Terminology and Knowledge Engineering. Proceedings International Congress on Terminology and Knowledge Engineering (Volume 2)*, (pp. 25–41)., Frankfurt a. M. Indeks.

Rieger, B. B. (1999). Computing fuzzy semantic granules from natural language texts. A computational semiotics approach to understanding word meanings. In Hamza, M. H. (Ed.), *Artificial Intelligence and Soft Computing, Proceedings of the IASTED International Conference, Anaheim/Calgary/Zürich*, (pp. 475–479). IASTED/Acta Press.

Rieger, B. B. (2002). Perception based processing of NL texts. Discourse understanding as visualized meaning constitution in scip systems. In Lotfi, A., John, B., & Garibaldi, J. (Eds.), *Recent Advances in Soft Computing (RASC-2002 Proceedings), Nottingham (Nottingham Trent UP)*, (pp. 506–511).

Rieger, B. B. & Thiopoulos, C. (1989). Situations, topoi, and dispositions: on the phenomenological modeling of meaning. In Retti, J. & Leidlmair, K. (Eds.), *5th Austrian Artificial Intelligence Conference, ÖGAI '89, Innsbruck, KI-Informatik-Fachberichte 208*, (pp. 365–375)., Berlin. Springer.

Salton, G. & McGill, M. J. (1983). *Introduction to Modern Information Retrieval*. New York: McGraw Hill.

van Rijsbergen, C. J. (1975). *Information Retrieval*. London, Boston: Butterworths.

Vapnik, V. N. (1998). *Statistical Learning Theory*. New York: Wiley & Sons.

Stephan Bloehdorn, Philipp Cimiano, Andreas Hotho, and Steffen Staab

An Ontology-based Framework for Text Mining

Structuring of text document knowledge frequently appears either by ontologies and metadata or by automatic (un-)unsupervised text categorization. This paper describes our integrated framework OTTO (OnTology-based Text mining framewOrk). OTTO uses text mining to learn the target ontology from text documents and uses then the same target ontology in order to improve the effectiveness of both supervised and unsupervised text categorization approaches.

1 Introduction

Most information resources available in the internet as well as within intranets are natural language text documents. It is often a prerequisite that these knowledge sources are structured in order to query for and retrieve them in a straightforward way. Speaking in very broad terms we recognize ongoing efforts for this purpose in two major directions.

First, researchers and practitioners working in the areas of information retrieval and text mining seek to find categories of textual resources by various fully automatic methods. The approaches either *(i)* predefine a metric on a document space in order to cluster 'nearby' documents into meaningful groups of documents (called 'unsupervised categorization' or 'text clustering'; Salton (1989)) or *(ii)* they adapt a metric on a document space to a manually predefined sample of documents assigned to a list of target categories such that new documents may be assigned to labels from the target list of categories, too ('supervised categorization' or 'text classification'; Sebastiani (2002)).

Second, researchers and practitioners working mainly in the areas of thesauri (Foskett 1997) and ontologies (Staab & Studer 2004) predefine conceptual structures and assign metadata to the documents that confirm to these conceptual structures.

Thereby, each of the two directions exhibits its advantages and problems. On the one hand the categorization of documents is (comparatively) cheap[1], but the

1 Automatic approaches are comparatively cheap even though the provisioning of sample data for supervised categorization may imply considerable, and sometimes even unbearable, costs.

quality of its document categorization for larger sets of target categories as well as the understandability of its results are often quite low. On the other hand, the quality of manual metadata may be very good, but the cost of building an ontology and adding manual metadata typically are one or several orders of magnitude higher than for automatic approaches.

To gain both advantages, while diminishing both their drawbacks at once, we here propose an approach of integrated ontology learning and text mining framework, viz. OTTO (OnTology-based Text mining framewOrk). Our implementation of OTTO includes a number of methods for (semi-)automatic ontology construction (also called *ontology learning*; Maedche & Staab (2004)) in order to provide for rich conceptual structures. Then, OTTO allows for exploitation of ontologies learned in this way by supervised or unsupervised text categorization.

We have shown in multiple contributions that ontology learning may be performed effectively (Maedche & Staab 2004; Cimiano et al. 2004b) and that text categorization may profit from ontologies (Bloehdorn & Hotho 2004; Hotho et al. 2003b, a). The integration we propose here allows for a tight integration of the two approaches combining their advantages.

The structure of the remainder of the paper is as follows: in Section 2 we introduce the overall OTTO text mining framework. In Section 3 we first present the TextToOnto system, which is designed to support the ontology engineer in the development of domain ontologies by applying text mining techniques. In this section we focus in particular on recent developments — as compared to Maedche & Staab (2004). In Section 4 we describe the approaches to text clustering and classification making use of ontologies as background knowledge. In Section 5 we discuss some related work and Section 6 concludes the paper.

2 General Architecture and Ontology Model

Figure 1 illustrates the overall OTTO system architecture. The architecture builds upon the Karlsruhe Ontology and Semantic Web Infrastructure (KAON)[2] that provides the access to implementations of our formal ontology model.

Ontology Model and Infrastructure KAON is a general and multi-functional open source ontology management infrastructure and tool suite developed

2 Forschungszentrum Informatik (FZI, WIM group, Karlsruhe) and Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB, Karlsruhe) (eds.) (2001-2005). KAON Homepage, `http://kaon.semanticweb.org` [accessed May 2005].

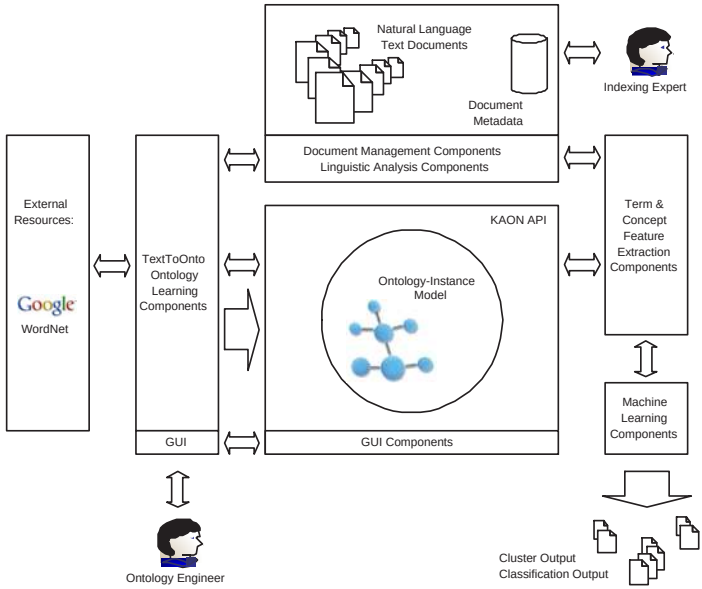

Figure 1: Overall OTTO System Architecture

at Karlsruhe University. KAON is built around the Ontology-Instance-Model (OI-model), a formal ontology model. In what follows we present our definition of an ontology which constitutes the formal model underlying an OI-model and we sketch the basic KAON system infrastructure. However, we only describe those parts of our more extensive ontology definition (E. Bozsak et al. 2002) that are needed for this paper.

Definition 2.1 (Core Ontology) *A* core ontology *is a structure*

$$\mathcal{O} := (C, \leq_C, R, \sigma, \leq_R)$$

consisting of two disjoint sets C and R whose elements are called concept identifiers *and* relation identifiers*, resp., a partial order $\leq_C$ on C, called* concept hierarchy *or* taxonomy*, a function $\sigma\colon R \to C^+$ called* signature*, a partial order $\leq_R$ on R, called* relation hierarchy*, where $r_1 \leq_R r_2$ implies $|\sigma(r_1)| = |\sigma(r_2)|$ and $\pi_i(\sigma(r_1)) \leq_C \pi_i(\sigma(r_2))$, for each $1 \leq i \leq |\sigma(r_1)|$ and C^+ is the set of tuples over C with at least one element and π_i is the i-th component of a given tuple.*

Definition 2.2 (Subconcepts and Superconcepts) *If $c_1 <_C c_2$ for any $c_1, c_2 \in C$, then c_1 is a* subconcept (specialization) *of c_2 and c_2 is a* superconcept (generalization) *of c_1. If $c_1 <_C c_2$ and there exists no $c_3 \in C$ with $c_1 <_C c_3 <_C c_2$, then c_1 is a direct subconcept of c_2, and c_2 is a direct superconcept of c_1, denoted by $c_1 \prec c_2$.*

The partial order $<_C$ relates the concepts in an ontology in form of specialization and generalization relationships, resulting in a hierarchical arrangement of concepts[3]. These relationships correspond to what is generally known as *is-a* or *is-a-special-kind-of* relations[4].

Often we will call concept identifiers and relation identifiers just *concepts* and *relations*, resp., for sake of simplicity. Almost all relations in practical use are binary. For those relations, we define their *domain* and their *range*.

Definition 2.3 (Domain and Range) *For a relation $r \in R$ with $|\sigma(r)| = 2$, we define its* domain *and its range by* $\mathrm{dom}(r) := \pi_1(\sigma(r))$ *and* $\mathrm{range}(r) := \pi_2(\sigma(r))$.

According to the international standard ISO 704, we provide names for the concepts (and relations). Instead of 'name', we here call them 'sign' or 'lexical entries' to better describe the functions for which they are used.

Definition 2.4 (Lexicon for an Ontology) *A* lexicon for an ontology $\mathcal{O}$ *is a tuple $Lex := (S_C, Ref_C)$ consisting of a set S_C, whose elements are called signs for concepts (symbols), and a relation $Ref_C \subseteq S_C \times C$ called lexical reference for concepts, where $(c, c) \in Ref_C$ holds for all $c \in C \cap S_C$. Based on Ref_C, for $s \in S_C$ we define $Ref_C(s) := \{c \in C | (s, c) \in Ref_C\}$. Analogously, for $c \in C$ it is $Ref_C^{-1}(c) := \{s \in S_C | (s, c) \in Ref_C\}$. An ontology with lexicon is a pair $(\mathcal{O}, Lex)$ where $\mathcal{O}$ is an ontology and Lex is a lexicon for $\mathcal{O}$.*

While the above definitions are related to the intensional and lexical aspects of an ontology, the following definition of a knowledge base relates to its extensional aspects:

Definition 2.5 (Knowledge Base) *A* knowledge base *is a structure*

$$KB := (C_{KB}, R_{KB}, I, \iota_C, \iota_R)$$

3 Note that this hierarchical structure is not necessarily a tree structure. It may also be a *directed acyclic graph* possibly linking concepts to multiple superconcepts at the same time.

4 In ontologies that are more loosely defined, the hierarchy may, however, not be as explicit as *is-a* relationships but rather correspond to the notion of *narrower-than* vs. *broader-than*. Note, however, that in many settings this view is considered as a very bad practice as it may lead to inconsistencies when reasoning with ontologies. However, this problem is not preeminent in the context of this work (Wielinga et al. 2001).

consisting of two sets C_{KB} and R_{KB}, a set I whose elements are called instance identifiers *(or* instances *or* objects *for short), a function $\iota_C \colon C_{KB} \to \mathfrak{P}(I)$ called* concept instantiation*, a function $\iota_R \colon R_{KB} \to \mathfrak{P}(I^+)$ with $\iota_R(r) \subseteq \prod_{c \in \sigma(r)} \iota_C(c)$, for all $r \in R$. The function ι_R is called* relation instantiation*,*

where $\mathfrak{P}(M)$ stands for the powerset of a set M and $\prod_i M_i$ for the cross-product of the sets M_i.

KAON features a full-fledged API that allows programmatic access to different implementations of the formal ontology model described. Currently, two different implementations of the KAON API are available: whereas the *KAON Engineering Server* is an ontology server using a scalable database representation of ontologies, *APIonRDF* is a main-memory implementation of the KAON API based on RDFS, a simple modelling language on top of the Resource Description Framework (RDF) formalism, both being developed by the W3C. *KAON OI-modeler* provides a graphical environment for ontology editing.

OTTO Text Mining Extensions OTTO's architecture is organized around KAON's OI-model and features various text mining modules (Figure 1). Separate document corpus management components allow to manage text document corpora and associated metadata information. Another core group of components offers basic linguistic analysis services like stemming, POS pattern analysis, word frequency calculations and the like, which are commonly used by all other components. The TextToOnto ontology learning algorithms, some of which will be described in section 3, can be applied to learn ontological structures from document corpora which are then stored in a corresponding OI-model. Some of the TextToOnto modules also make use of external resources like WordNet or `Google` in order to query the WWW. Comprehensible GUIs provide intuitive access to the learning algorithms as well as to the OI-model for the user. On the other hand, given that a suitable ontology is available, the OTTO concept extraction components allow to analyze text documents and extract a conceptual document representation that complements the classical bag-of-words document representation. We will have a closer look at these modules in section 3. The feature extraction components are carefully designed to allow flexible connections to different software modules that are capable to perform classical machine learning algorithms like classification or clustering. Implemented connectors include connectors to Weka[5], a Java based machine-learning library and application, or MATLAB.

5 Eibe, Frank et al. (eds.) (1999-2005). Weka 3: Data Mining Software in Java, `http://www.cs.waikato.ac.nz/~ml/weka/` [accessed May 2005].

3 The TextToOnto Ontology Learning Framework

TextToOnto[6] is a system conceived to support the ontology engineer in the task of creating and maintaining ontologies. For this purpose, it employs text mining techniques such as term clustering and matching of lexico-syntactic patterns as well as other resources of a general nature such as WordNet (Fellbaum 1998). In what follows, we describe the architecture as well as the algorithms used by the system to facilitate the ontology engineering process.

3.1 The TextToOnto Architecture

The main components of TextToOnto are the following (compare Maedche & Staab (2004) as well as Figure 2):

- The **Ontology Management Component** provides basic ontology management functionality. In particular, it supports editing, browsing and evolution of ontologies. For this purpose it builds upon the Karlsruhe Ontology and Semantic Web Infrastructure (KAON). In fact, KAON's OI-model is the key data structure on which the ontology learning process is centered.

- The **Algorithm Library Component** acts as the algorithmic backbone of the framework. It incorporates a number of text mining methods, e.g. conceptual clustering, terminology extraction, pattern matching as well as machine learning techniques, e.g. association rules and classifiers.

- **Coordination Component:** The ontology engineer uses this component to interact with the different ontology learning algorithms from the algorithm library. Comprehensive user interfaces are provided to select relevant corpora, set different parameters and start the various algorithms.

From a methodological point of view, the data structure around which the whole ontology learning process is centered is the OI-model as described in Section 2. The user can start with an empty OI-model and learn a new ontology from scratch or select an existing one and add new instances or relations. In this paper we do not describe all these components in detail, but refer the reader to Maedche & Staab (2004) instead. The main contribution of the present section is

6 The system is freely available and can be downloaded at Cimiano, Ph. et al. (eds.) (2003-2005). Project TextToOnto Homepage (Sourceforge), `http://sourceforge.net/projects/texttoonto/` [accessed May 2005].

in fact to present new components extending the functionalities of the system as described therein.

3.2 Ontology Learning Algorithms

In earlier work, we presented approaches for learning taxonomic relations via (i) top-down or bottom-up clustering techniques (Maedche et al. 2002; Cimiano et al. 2004b), (ii) matching lexico-semantic patterns or (iii) classification algorithms such as k-Nearest-Neighbours (Maedche & Staab 2002). Further, we also developed algorithms for extracting general binary relations between concepts based on association rules mining (Maedche & Staab 2000). Another possibility we examined is to extract domain ontologies from large, domain independent ontologies by pruning (Volz et al. 2003). In this paper we present three new algorithms actually implemented within TextToOnto with the purpose of:

- constructing taxonomies using a conceptual clustering algorithm, i.e. Formal Concept Analysis (TaxoBuilder component)
- constructing taxonomies by combining information aggregated from WordNet, Hearst (Hearst 1992) patterns matched in a corpus as well as certain heuristics (TaxoBuilder component)
- classifying instances into the ontology by using lexico-syntactic patterns (InstanceExtraction component)
- extracting labelled relations and specifying their domain and range (RelationLearning component)

3.2.1 TaxoBuilder

TaxoBuilder is a component developed for the purpose of learning concept hierarchies from scratch. It can be used in two different modes:

- In **FCA** mode, TaxoBuilder employs the technique described in Cimiano et al. (2004a) to learn a concept hierarchy by means of Formal Concept Analysis (Ganter & Wille 1999).
- In **Combination** mode, TaxoBuilder uses different sources of evidence such as WordNet, Hearst patterns (Hearst 1992) matched in a corpus as well as certain heuristics to find taxonomic relations.

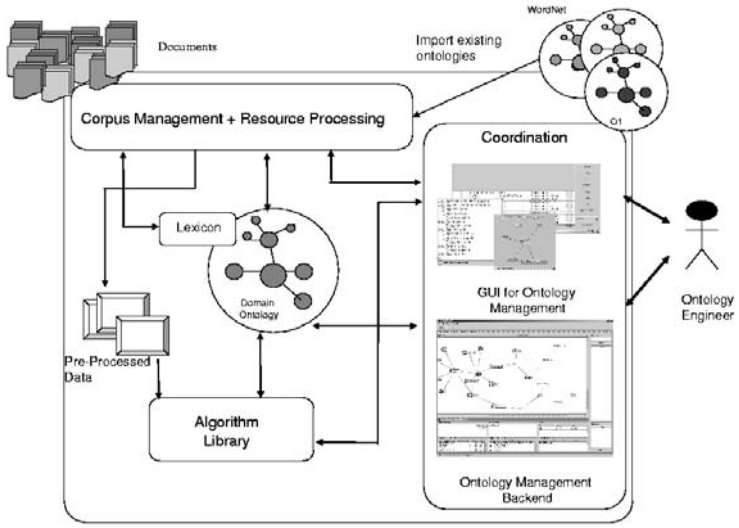

Figure 2: TextToOnto Architecture

In the FCA mode, TaxoBuilder extracts syntactic dependencies from text by applying shallow parsing techniques. In particular it extracts verb-object relations and uses them as context attributes for Formal Concept Analysis as described in Cimiano et al. (2004a) and Cimiano et al. (2004b). The lattice is then built in the background and transformed into an OI-model by removing the bottom *formal concept* and introducing for every formal concept an ontological concept named with its intent. For every element in the extension of this formal concept we introduce an ontological subconcept. Figure 3 shows for example the lattice automatically learned for the following terms: *apartment, hotel, car, bike* and *trip*. The corresponding formal context is depicted in Table 1. As already mentioned, the lattice is calculated in the background and transformed into the OI-model in Figure 4.

This approach has been evaluated in Cimiano et al. (2004a) and Cimiano et al. (2004b) by comparing the automatically generated concept hierarchies with handcrafted hierarchies for a given domain in terms of the similarity measures described in Maedche & Staab (2002).

	runable	**o**fferable	**n**eedable	**s**tartable	**m**eanable	**s**eemable	**a**ttemptable	**c**ruiseable	**f**illable
apartment	x								
hotel	x	x							
car			x						
bike				x					
trip					x	x	x	x	x

Table 1: Formal Context for the terms *apartment, hotel, car, bike* and *trip*

In the *Combination* mode, TaxoBuilder exploits (i) the vertical relations heuristic in Missikoff et al. (2002), (ii) Hearst patterns (Hearst 1992) as well as (iii) the hypernym relations in WordNet (Fellbaum 1998). Now given a pair of terms, say t_1 and t_2, they could be taxonomically related in two ways: is-a(t_1,t_2) or is-a(t_2,t_1). In order to decide in which way they are related we compute the evidence for both of the relations by taking into account the above information sources. In particular, we take into account the above mentioned heuristic, the number of Hearst patterns in the corpus found as well as the number of hypernymic paths in WordNet between two terms. We sum up all these values and choose the relation with maximum evidence. All the taxonomic relations found in this way between a given set of terms in question are then added to the OI-model after removing potential cycles. This method has been proven to be an effective way of quickly learning concept hierarchies. Figure 5 shows a concept hierarchy automatically acquired with the combination method out of 500 texts from the online *Lonely Planet* world guide[7].

3.2.2 InstanceExtraction

The InstanceExtraction component discovers instances of concepts of a given ontology in a text corpus. So, it needs a text corpus and a non-empty OI-model as input. It can either be used in a semi-automatic or fully automatic way. In the first case, it will present the candidate instances to the user asking for confirmation, while in the second case it will simply add the discovered instances to the corresponding OI-model. In order to discover these instances, InstanceExtraction makes use of a combination of patterns from Hearst (1992) and Hahn & Schnattinger (1998). The user can choose which of the different patterns s/he wants to use. The patterns are described in what follows:

7 Lonely Planet Publications (2005). Lonely Planet Homepage, `http://www.lonelyplanet.com` [accessed May 2005].

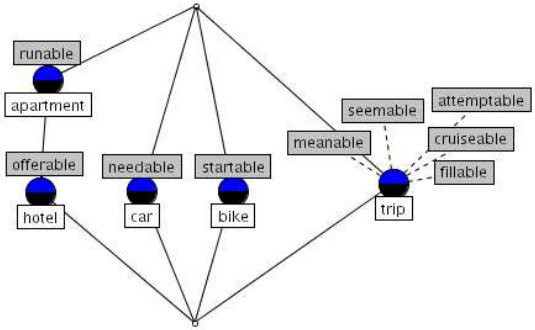

Figure 3: Concept Lattice

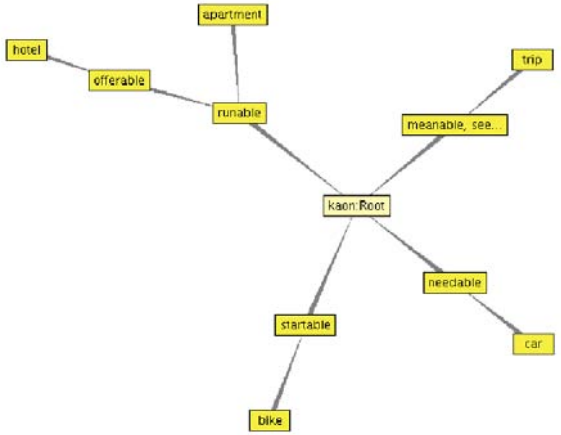

Figure 4: OI-model automatically learned with the FCA approach

Hearst Patterns The first four patterns have been used by Hearst to identify *is-a*-relationships between the concepts referred by two terms in the text. However, they can also be used to categorize a named entity or instance into an ontology. In our approach we have the underlying assumption that common nouns represent concepts and proper nouns represent instances. In order to

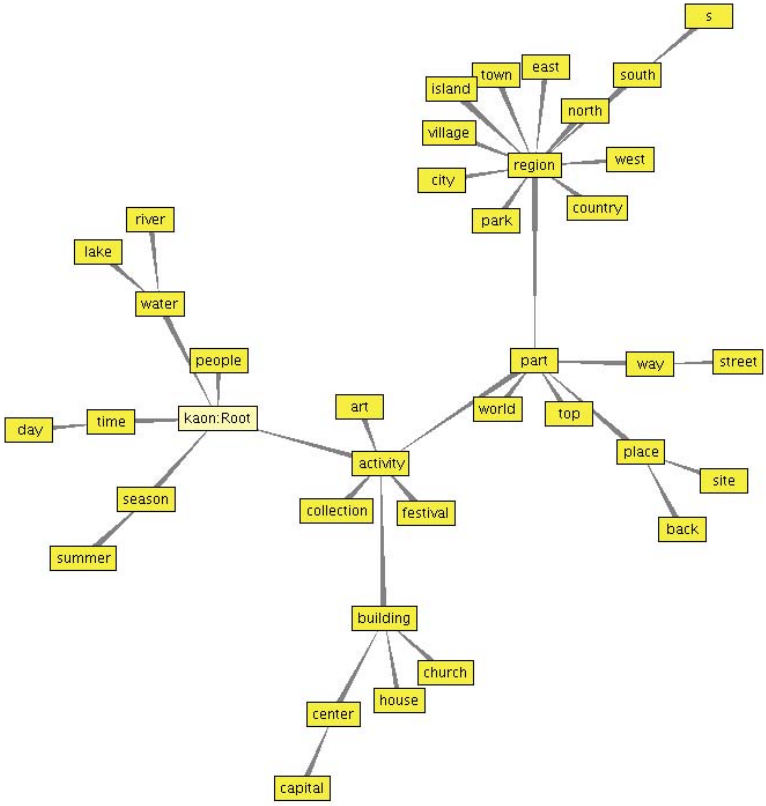

Figure 5: OI-model automatically learned with the combination approach

identify noun phrases representing concepts, henceforth $NP_{CONCEPT}$, and noun phrases representing instances, henceforth $NP_{INSTANCE}$, we use a shallow parsing technique based on matching regular expressions over part-of-speech tags to identify the two types of noun phrases described above. The patterns reused from Hearst are:

HEARST1: $NP_{CONCEPT}$ such as $NP_{INSTANCE}$

HEARST2: such $NP_{CONCEPT}$ as $NP_{INSTANCE}$

HEARST3: $NP_{CONCEPT}$, (especially|including) $NP_{INSTANCE}$

HEARST4: $NP_{INSTANCE}$ (and|or) other $NP_{CONCEPT}$

The above patterns would match the following expressions (in this order): *hotels such as Ritz*; *such hotels as Hilton*; *presidents, especially George Washington*; and *the Eiffel Tower and other sights in Paris*.

Definites The next patterns are about definites, i.e. noun phrases introduced by the definite determiner '*the*'. Frequently, definites actually *refer* to some entity previously mentioned in the text. In this sense, a phrase like '*the hotel*' does not stand for itself, but it points as a so-called anaphora to a unique hotel occurring in the preceding text. Nevertheless, it has also been shown that in common texts more than 50% of all definite expressions are *non-referring*, i.e. they exhibit sufficient descriptive content to enable the reader to uniquely determine the entity referred to from the global context (Poesio & Vieira 1998). For example, the definite description '*the Hilton hotel*' has sufficient descriptive power to uniquely pick-out the corresponding real-world entity for most readers. One may deduce that '*Hilton*' is the name of the real-world entity of type hotel to which the above expression refers.

Consequently, we apply the following two patterns to categorize candidate proper nouns by definite expressions:

DEFINITE1: the $NP_{INSTANCE}$ $NP_{CONCEPT}$

DEFINITE2: the $NP_{CONCEPT}$ $NP_{INSTANCE}$

The first and the second pattern would, e.g., match the expressions '*the Hilton hotel*' and '*the hotel Hilton*', respectively.

Apposition and Copula The following pattern makes use of the fact that certain entities appearing in a text are further described in terms of an apposition as in '*Excelsior, a hotel in the center of Nancy*'. The pattern capturing this intuition looks as follows:

APPOSITION: $NP_{INSTANCE}$, a $NP_{CONCEPT}$

The probably most explicit way of expressing that a certain entity is an instance of a certain concept is by the verb '*to be*', as for example in '*The Excelsior is a hotel in the center of Nancy*'. Here's the general pattern:

COPULA: $NP_{INSTANCE}$ is a $NP_{CONCEPT}$

Pattern	Suggested	Annotator 1	Annotator 2	Annotator 3	Accuracy
HEARST1	2	40.00%	40.00%	60.00%	46.66%
DEFINITE1	19	21.05%	36.84%	36.84%	31.56%
DEFINITE2	74	91.36%	93.83%	96.30%	93.83%
APPOSITION	28	56.00%	62.00%	62.00%	60.00%
COPULA	22	66.67%	66.67%	63.64%	65.66%
ALL	188	69.15%	73.40%	74.47%	72.34%

Table 2: Accuracy of each of the patterns

Evaluation In order to evaluate our pattern-based approach to categorizing instances, we considered the 500 randomly selected web pages from *Lonely Planet* and used a part-of-speech (POS) tagger[8] as well handcrafted rules to match non-recursive NPs representing concepts and instances, respectively, as well as the above patterns.
We then presented the found instance-concept pairs to three different subjects for validation. They had the possibility of validating the relationship, adding the concept name to the instance, rejecting the relationship or expressing their doubt. The possibility of adding the concept name is important when judging a suggestion such as that *Lenin* is an instance of a *museum*. In this case, the users could decide that the suggestion of the system is not totally wrong and correct the suggestion by specifying that *Lenin museum* is the actual instance of a *museum*. In this case we counted the answer of the system as correct. Table 2 gives the accuracy for all the patterns based on the answers of the human subjects to the suggestions of the system. Unfortunately, no HEARST2, HEARST3 or HEARST4 instances were found in the texts, which shows that they are actually the ones which occur most rarely. Interestingly, it can be appreciated that the accuracy varies from pattern to pattern. Overall, the performance of the approach seems very reasonable as more than 72% of the suggested relations are judged as correct by the human subjects.

3.2.3 RelationLearning

The RelationLearning component also discovers candidate relations from text but in contrast to the association rule algorithm described in Maedche & Staab (2004) suggests a name for the relation to the user as well as *domain* and *range* for it. For this purpose, it employs a shallow parsing strategy to extract subcate-

8 We used the QTag POS-Tagger, cf. Mason, O. (1994-2003). QTag Homepage, `http://www.english.bham.ac.uk/staff/omason/software/qtag.html` [accessed May 2005].

gorization frames enriched with selectional restrictions specified with regard to the corresponding OI-model as described in Resnik (1997). In particular, it extracts the following syntactic frames:

- transitive, e.g. love(subj,obj)
- intransitive + PP-complement, e.g. walk(subj,pp(to))
- transitive + PP-complement, e.g. hit(subj,obj,pp(with))

RelationLearning then enriches these subcategorization frames semantically by finding the appropriate concept from a given ontology for each syntactic position. For each occurrence of a given syntactic frame, it extracts the nominal head in each syntactic position and augments the corresponding concept count by one. For each syntactic frame and syntactic position it chooses the most specific concept with maximal count. On the basis of these subcategorization frames, it suggests possible relations to the user for validation. For example, given the following enriched subcategorization frames

love(subj:*person*,obj:*person*)
walk(subj:*person*,to:*place*)
hit(subj:*person*,obj:*thing*,with:*contundent_object*)

the system would suggest the following relations to the user:

love(domain:*person*,range:*person*)
walk_to(domain:*person*,range:*place*)
hit(domain:*person*,range:*thing*)
hit_with(domain:*person*,range:*contundent_object*)

The main problem with this approach to discovering relations is related to data sparseness as for small to medium-sized corpora there are not enough verbs in the text collection connecting all the different concepts of the ontology together. In general with this approach we thus end up with only a small number of relations.

4 Ontology-based Text Clustering and Classification

Due to the ever growing amounts of textual information available electronically, users are facing the challenge of organizing, analyzing and searching large

numbers of documents. Systems that automatically classify text documents into predefined thematic classes or detect clusters of documents with similar content offer a promising approach to tackle this complexity. During the last decades, a large number of machine learning algorithms have been proposed for supervised and unsupervised text categorization. So far, however, existing text categorization systems have typically used the *Bag–of–Words model* known from information retrieval, where single words or word stems are uses as features for representing document content (Salton 1989). In this section we present an approach that exploits existing ontologies by using their lexica and concept hierarchies to improve results in both, supervised and unsupervised settings.

4.1 The Bag-of-Words Model

In the *Bag–of–Words paradigm*, documents are represented as bags of terms. Let D be the set of documents and $T = \{t_1, \ldots, t_m\}$ the set of all different terms occurring in D. The absolute frequency of term $t \in T$ in document $d \in D$ is given by $\mathrm{tf}(d, t)$. Term vectors are denoted $\vec{t_d} = (\mathrm{tf}(d, t_1), \ldots, \mathrm{tf}(d, t_m))$.

Stopwords and Stemming The initial term vectors produced so far in this approach can be further modified as described in what follows. *Stopwords* are words which are considered as non–descriptive within a bag–of–words approach. For example, for english language, it is common practice to use a standard list of 571 stopwords initially designed for the SMART system[9]. Typically, text documents are further processed to reduce the term representation to term stems, e.g. using the Porter stemmer introduced in Porter (1980). Using *stemmed terms*, one can construct a vector representation $\vec{t_d}$ for each text document.

Pruning Pruning rare terms also affects results. Depending on a pre-defined threshold δ, a term t is discarded from the representation (i. e., from the set T), if $\sum_{d \in D} \mathrm{tf}(d, t) \leq \delta$. In our experiments, we have for example used the values 0, 5 and 30 for δ. The rationale behind *pruning* is that infrequent terms do not help for identifying appropriate clusters, but may still add noise to the distance measures degrading overall performance.

Weighting Having extracted the collection of terms that make up the documents in a corpus, the corresponding numeric values of the terms within

9 SMART Project (eds.) Stopword List for English Information Retrieval, `http://www.unine.ch/info/clef/englishST.txt` [accessed May 2005].

the document have to be determined. A special case of term weighting is binary weighting, where the terms are represented as boolean variables. *Tfidf* weighs the frequency of a term in a document with a factor that discounts its importance when it appears in almost all documents. The tfidf (term frequency−inverted document frequency)[10] of term t in document d is defined by: $\text{tfidf}(d,t) := \log(\text{tf}(d,t)+1) * \log\left(\frac{|D|}{\text{df}(t)}\right)$ where $\text{df}(t)$ is the document frequency of term t that counts in how many documents term t appears. If tfidf weighting is applied then we replace the term vectors $\vec{t_d} := (\text{tf}(d,t_1),\ldots,\text{tf}(d,t_m))$ by $\vec{t_d} := (\text{tfidf}(d,t_1),\ldots,\text{tfidf}(d,t_m))$. There are more sophisticated measures than tfidf in the literature (see, e. g., Amati et al. (2001)), but we abstract herefrom, as this is not the main topic of this paper.

Deficiencies By using only single terms to represent document content any chosen machine learning algorithm is restricted to detecting patterns in the used *terminology* only, while *conceptual* patterns remain ignored. Specifically, systems using only words as features exhibit a number of inherent deficiencies:

1. *Multi-Word Expressions* with an own meaning like *"European Union"* are chunked into pieces with possibly very different meanings like *"union"*.

2. *Synonymous Words* like *"tungsten"* and *"wolfram"* are mapped into different features.

3. *Polysemous Words* are treated as one single feature while they may actually have multiple distinct meanings.

4. *Lack of Generalization*: there is no way to generalize similar terms like "beef" and "pork" to their common hypernym "meat".

While items 1 – 3 directly address issues that arise on the lexical level, items 4 rather addresses an issue that occurs at the conceptual level.

In our approach, we use background knowledge in form of simple ontologies (cf. section 2) to improve text classification and clustering results by directly addressing these problems. We propose a hybrid approach for document representation based on the common term stem representation which is enhanced with concepts extracted from the used ontologies.

10 tfidf actually refers to a class of weighting schemata. Above we have given the one we have used.

4.2 Enriching the Document Vectors with Cconcepts

In our approach, we exploit background knowledge about concepts that is explicitly given according to our ontological model (cf. section 2). For this purpose, we extend each term vector $\vec{t_d}$ by new entries for ontological concepts c appearing in the document set. Thus, the vector $\vec{t_d}$ is replaced by the concatenation of $\vec{t_d}$ with the concept vector $\vec{c_d} := (\mathrm{cf}(d, c_1), \ldots, \mathrm{cf}(d, c_l))$ having length $l = |C|$ and where $\mathrm{cf}(d, c)$ denotes the frequency of the appearance of concept $c \in C$ in document d as indicated by applying the reference function Ref_C to all terms in the document d. Hence, a term that also appears in the ontology would be accounted for at least twice in the new vector representation, i. e., once as part of the old $\vec{t_d}$ and at least once as part of $\vec{c_d}$. It could be accounted for also more often, because a term like "bank" has several corresponding concepts in the ontology.

To extract the concepts from texts, we have developed a detailed process, that can be used with any ontology with lexicon. The overall process comprises five processing steps that are described in the following.

1. Candidate Term Detection Due to the existence of multi-word expressions, the mapping of terms to concepts can not be accomplished by querying the lexicon directly for the single words in the document.

We have addressed this issue by developing a candidate term detection algorithm (Bloehdorn & Hotho 2004) that builds on the basic assumption that finding the longest multi-word expressions that appear in the text and the lexicon will lead to a mapping to the most specific concepts. The algorithm works by moving a window over the input text, analyzing the window content and either decreasing the window size if unsuccessful or moving the window further. For English, a window size of 4 is sufficient to detect virtually all multi-word expressions.

2. Syntactical Patterns Querying the lexicon directly for any expression in the window will result in many unnecessary searches and thereby in high computational requirements. Luckily, unnecessary search queries can be identified and avoided through an analysis of the part-of-speech (POS) tags of the words contained in the current window. Concepts are typically symbolized in texts within *noun phrases*. By defining appropriate POS patterns and matching the window content against these, multi-word combinations that will surely not

symbolize concepts can be excluded in the first hand and different syntactic categories can be disambiguated.

3. Morphological Transformations Typically the lexicon will not contain all inflected forms of its entries. If the lexicon interface or separate software modules are capable of performing base form reduction on the submitted query string, queries can be processed directly. For example, this is the case with WordNet. If the lexicon, as in most cases, does not contain such functionalities, a simple fallback strategy can be applied. Here, a separate index of stemmed forms is maintained. If a first query for the inflected forms on the original lexicon turned out unsuccessful, a second query for the stemmed expression is performed.

4. Word Sense Disambiguation Having detected a lexical entry for an expression, this does not necessarily imply a one-to-one mapping to a concept in the ontology. Although multi-word-expression support and POS pattern matching reduce ambiguity, there may arise the need to disambiguate an expression versus multiple possible concepts. The *word sense disambiguation (WSD)* task is a problem in its own right (Ide & Véronis 1998) and was not the focus of our work.

In our experiments, we have used three simple strategies proposed in Hotho et al. (2003c) to process polysemous terms:

- The "all" strategy leaves actual disambiguation aside and uses all possible concepts.
- The "first" strategy exploits WordNet's capability to return synsets ordered with respect to usage frequency. This strategy chooses the most frequent concept in case of ambiguities.
- The "context" strategy performs disambiguation based on the degree of overlap of lexical entries for the semantic vicinity of candidate concepts and the document content as proposed in Hotho et al. (2003c).

5. Generalization The last step in the process is about going from the specific concepts found in the text to more general concept representations. However, we do not only add the concepts directly representing the terms but also the corresponding superconcept along the path to the root of the concept hierarchy. An important issue here is to restrict the number of levels up in the hierarchy considered for adding superconcepts. The following procedure realizes this idea

by adding to the concept frequency of higher level concepts in a document d the frequencies of their subconcepts (of at most r levels down in the hierarchy). I. e., the vectors we consider are first of the form $\vec{t_d} := (\mathrm{tf}(d,t_1),\ldots,\mathrm{tf}(d,t_m), \mathrm{cf}(d,c_1),\ldots,\mathrm{cf}(d,c_n))$ (the concatenation of an initial term representation with a concept vector). Then the frequencies of the concept vector part are updated, for a user-defined $r \in \mathbb{N}_0$, in the following way: For all $c \in C$, replace $\mathrm{cf}(d,c)$ by $\mathrm{cf}'(d,c) := \sum_{b \in H(c,r)} \mathrm{cf}(d,b)$, where $H(c,r) := \{c' | \exists c_1,\ldots,c_i \in C\colon c' \prec c_1 \prec \ldots \prec c_i = c,\ 0 \leq i \leq r\}$ gives for a given concept c the r next subconcepts in the taxonomy. In particular $H(c,\infty)$ returns all subconcepts of c. This implies: The strategy $r = 0$ does not change the given concept frequencies, $r = n$ adds to each concept the frequency counts of all subconcepts in the n levels below it in the ontology and $r = \infty$ adds to each concept the frequency counts of all its subconcepts.

4.3 Machine Learning Components and Results

As documents have been processed with the term and concept extraction components, they can be processed using standard machine learning algorithms. Currently, we use an interface that allows easy integration of the resulting hybrid document feature representations into WEKA[11], a Java-based multi-purpose machine learning environment.

Unsupervised Text Categorization (Clustering) deals with grouping documents together that are homogenous in some way. In contrast to supervised text categorization, where the classes in question are assigned outside the learning environment, it is the very task of the clustering algorithm to find good groups (clusters) in the first hand when no classes are given a priori.

For clustering (Steinbach et al. 2000), it has been shown that Bi-Section-KMeans – a variant of KMeans – frequently outperforms standard KMeans as well as agglomerative clustering techniques. Thus, we make use of Bi-Section-KMeans as clustering method. The similarity between two text documents $d_1, d_2 \in D$ is measured by the cosine of the angle between the vectors $\vec{t}_1, \vec{t}_2$ representing them:

$$\cos(\sphericalangle(\vec{t}_1,\vec{t}_2)) = \frac{\vec{t}_1 \cdot \vec{t}_2}{\| \vec{t}_1 \| \cdot \| \vec{t}_2 \|}$$

In experiments reported in a previous paper (Hotho et al. 2003c), we showed that conceptual representations can significantly improve text cluster purity by

11 See footnote 5 above.

reducing the variance among the representations within the given classes of related documents. In the experiments on the well-known Reuters-21578 corpus using WordNet as ontology, we were able to show a significant improvement of up to 8% using a simple word sense disambiguation strategy combined with generalization based on term and concept vectors. We observed a performance drop without using any word sense disambiguation. An investigation of the different clusters revealed that some given classes of the Reuters corpus could be found with a high purity by the clustering algorithm while for other classes purity decreases.

Supervised Text Categorization Not surprisingly, supervised text categorization and clustering are closely related as both are concerned with "groupings" of objects. However, in the supervised setting, these groupings are given by the common membership to a thematic class that is assigned to sample documents before the training process starts. The training process then induces hypotheses of how the document space is shaped according to which new documents are assigned target categorizations.

Many different supervised categorization algorithms have been designed and virtually all of them have been used for text categorization tasks, including probabilistic classifiers like Naïve Bayes, Linear Discriminant Functions like Perceptrons or more recently Support Vector Machines, Decision Trees and Decision Rule Classifiers, Nonparametric Classifiers like k-Nearest-Neighbours and Ensemble Classifiers, most namely Bagging and Boosting. Comparisons like in Sebastiani (2002) suggest that *Boosting* (Schapire & Singer 2000) and *Support Vector Machines* (Joachims 1998) are the most promising approaches for handling text classification tasks.

In a recent experimental evaluation on two well-known text corpora (Bloehdorn & Hotho 2004), the Reuters-21578 corpus and the medical document corpus OHSUMED, were able to show the positive effects of our approach. Using Boosting as actual learning algorithm and both, term stems and concepts as features, we were able to achieve consistent improvements of the categorization results. In terms of the well-known F_1 measure, that combines precision and recall results this improvement was in the 1% – 3% range for the Reuters-21578 corpus and in the 2.5% – 7% range for the OHSUMED corpus[12]. The difference between both evaluations is probably explained best by the fact that the medical documents in the OHSUMED corpus make heavy use of multi-word-

12 These figures are based on *macro-averaged* F_1 results with *micro-averaged* results being slightly worse on the Reuters-21578 corpus while being fairly similar on the OHSUMED corpus.

expressions, synonyms and very specific terms which obfuscates a pure term based representation very much, while conceptual features tend to reduce noise in these situations.

5 Related Work

In this section we discuss work related to text mining techniques for ontology learning as well as text clustering and classification techniques relying on background knowledge.

Ontology Learning There is quite a long tradition in learning concept hierarchies by clustering approaches such as the ones presented in Hindle (1990); Pereira et al. (1993); Faure & Nedellec (1998); Caraballo (1999); Bisson et al. (2000) as well as by matching lexico-syntactic patterns as described in Hearst (1992, 1998); Charniak & Berland (1999); Poesio et al. (2002); Ahmid et al. (2003); Jouis (1993); Seguela (2001); Cimiano et al. (2004). In this section we focus on the discussion of frameworks and systems designed for supporting the ontology engineering process. In the ASIUM system (Faure & Nedellec 1998) nouns appearing in similar contexts are iteratively clustered in a bottom-up fashion. In particular, at each iteration, the system clusters the two most similar extents of some argument position of two verbs and asks the user for validation. Bisson et al. (2000) present an interesting framework and a corresponding workbench - Mo'K - allowing users to design conceptual clustering methods to assist them in an ontology building task. The framework is general enough to integrate different clustering methods. Velardi et al. (2001) present the OntoLearn system which discovers i) the domain concepts relevant for a certain domain, i.e. the relevant terminology, ii) named entities, iii) 'vertical' (is-a or taxonomic) relations as well as iv) certain relations between concepts based on specific syntactic relations. In their approach a 'vertical' relation is established between a term t_1 and a term t_2, i.e. is-a(t_1,t_2), if the head of t_2 matches the head of t_1 and additionally the former is additionally modified in t_1. Thus, a 'vertical' relation is for example established between the term 'international credit card' and the term 'credit card', i.e. is-a(international credit card,credit card).

Background Knowledge for Text Categorization Tasks To date, the work on integrating semantic background knowledge into text categorization is quite scattered. Much of the early work with semantic background knowledge in infor-

mation retrieval was done in the context of *query expansion* techniques (Bodner & Song 1996). Others like Green (1999) or Kushal Dave (2003) were more or less successful in using WordNet synsets to improve the text clustering task. Further they only investigate the use of WordNet and not ontologies in general by only applying a small number of strategies of the kind that we have investigated.

Recent experiments with conceptual feature representations for supervised text categorization are presented in Wang et al. (2003). These and other similar published results are, however, still too few to allow insights on whether positive results can be achieved in general. In some cases, even negative results were reported. For example, a comprehensive comparison of approaches with different document representations based on word senses and different learning algorithms ends with the conclusion of the authors that *"the use of word senses does not result in any significant categorization improvement"* (Kehagias et al. 2000). While we have been able to confirm the results they achieved for their method inventory, we have also shown that an enriched set of methods improves results by a large margin. In particular, we have found that ontology-based approaches benefit from feature weighting and word sense disambiguation.

Alternative approaches for conceptual representations of text documents that are not based on background knowledge compute kind of "statistical" concepts. Very good results with a probabilistic variant of LSA known as Probabilistic Latent Semantic Analysis (pLSA) were recently reported in Cai & Hofmann (2003). The experiments reported therein are of particular interest as the classification was also based on AdaBoost and was also using a combined term-concept representation, the latter being however automatically extracted from the document corpus using pLSA. We have investigated some of these approaches. We have been able to show that indeed LSA improves text clustering. In addition, we could show that ontology based approaches further improve the results achieved by LSA. Further comparisons with pLSA remain to be done in future research.

6 Conclusion and Further Work

Exploiting knowledge present in textual documents is an important issue in building systems for knowledge management and related tasks. In this paper we have presented OTTO (OnTology-based Text mining framewOrk), a framework centered around the KAON OI-model for the interaction between ontologies, i.e. explicit formalizations of a shared conceptualization and natural language texts in two directions.

First, natural language processing techniques combined with machine learning algorithms allow to build or extend ontologies in a semi-automatic manner. This field, known as ontology learning, is critical for building domain specific ontologies with fewer manual effort. We have presented recent innovations in this field that have been implemented in the TextToOnto modules of our OTTO framework.

Second, background knowledge in form of ontologies enhances the performance of classical text mining tasks such as text classification and text clustering. Semantic features extracted from ontologies with help of the OTTO text mining components leverage the classical bag–of–words representation to a higher semantic level and thereby improve classification accuracy and cluster purity.

Future work in this area will focus on a more thorough analysis how domain ontologies learned by means of ontology learning techniques can improve text classification and clustering tasks on documents from the same corpus, compared to using general purpose ontologies or linguistic resources like WordNet. Preliminary results show that this is a promising approach and will heavily influence the design of future OTTO module extensions.

References

Ahmid, K., Tariq, M., Vrusias, B., & Handy, C. (2003). Corpus-based thesaurus construction for image retrieval in specialist domains. In *Proceedings of the 25th European Conference on Advances in Information Retrieval (ECIR)*.

Amati, G., Carpineto, C., & Romano, G. (2001). Fub at trec-10 web track: A probabilistic framework for topic relevance term weighting. In *TREC 2001*. online publication.

Bisson, G., Nedellec, C., & Canamero, L. (2000). Designing clustering methods for ontology building - The Mo'K workbench. In *Proceedings of the ECAI Ontology Learning Workshop*.

Bloehdorn, S. & Hotho, A. (2004). Boosting for Text Classification with Semantic Features. In *Proceedings of the MSW 2004 workshop at the 10th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* .

Bodner, R. C. & Song, F. (1996). Knowledge-Based Approaches to Query Expansion in Information Retrieval. In *Advances in Artificial Intelligence*. New York, NY, USA: Springer.

Cai, L. & Hofmann, T. (2003). Text Categorization by Boosting Automatically Extracted Concepts. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, Toronto, Canada. ACM Press.

Caraballo, S. (1999). Automatic construction of a hypernym-labeled noun hierarchy from text. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, (pp. 120–126).

Charniak, E. & Berland, M. (1999). Finding parts in very large corpora. In *Proceedings of the 37th Annual Meeting of the ACL*, (pp. 57–64).

Cimiano, P., Hotho, A., & Staab, S. (2004a). Clustering ontologies from text. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*.

Cimiano, P., Hotho, A., & Staab, S. (2004b). Comparing conceptual, divisive and agglomerative clustering for learning taxonomies from text. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*.

Cimiano, P., Pivk, A., Schmidt-Thieme, L., & Staab, S. (2004). Learning taxonomic relations from heterogeneous evidence. In *Proceedings of the ECAI'04 Workshop on Ontology Learning and Population*.

E. Bozsak et al. (2002). KAON - Towards a large scale Semantic Web. In *Proceedings of the Third International Conference on E-Commerce and Web Technologies (EC-Web)*. Springer Lecture Notes in Computer Science.

Faure, D. & Nedellec, C. (1998). A corpus-based conceptual clustering method for verb frames and ontology. In Velardi, P. (Ed.), *Proceedings of the LREC Workshop on Adapting lexical and corpus resources to sublanguages and applications*.

Fellbaum, C. (1998). *WordNet, an electronic lexical database*. MIT Press.

Foskett, D. J. (1997). Thesaurus. In P. Willett & K. Sparck-Jones (Eds.), *Reproduced in Readings in Information Retrieval* (pp. 111–134). Morgan Kaufmann.

Ganter, B. & Wille, R. (1999). *Formal Concept Analysis – Mathematical Foundations*. Springer Verlag.

Green, S. J. (1999). Building hypertext links by computing semantic similarity. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, *11*(5), 713–730.

Hahn, U. & Schnattinger, K. (1998). Towards text knowledge engineering. In *AAAI'98/IAAI'98 Proceedings of the 15th National Conference on Artificial Intelligence and the 10th Conference on Innovative Applications of Artificial Intelligence*.

Hearst, M. (1992). Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics*.

Hearst, M. (1998). Automated discovery of wordnet relations. In C. Fellbaum (Ed.), *WordNet: An Electronic Lexical Database*. MIT Press.

Hindle, D. (1990). Noun classification from predicate-argument structures. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, (pp. 268–275).

Hotho, A., Staab, S., & Stumme, G. (2003a). Explaining text clustering results using semantic structures. In *Principles of Data Mining and Knowledge Discovery, 7th European Conference, PKDD 2003*.

Hotho, A., Staab, S., & Stumme, G. (2003b). Ontologies improve text document clustering. In *Proc. of the ICDM 03, The 2003 IEEE International Conference on Data Mining*, (pp. 541–544).

Hotho, A., Staab, S., & Stumme, G. (2003c). Wordnet improves Text Document Clustering. In *Proceedings of the Semantic Web Workshop of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, Toronto, Canada. ACM Press.

Ide, N. & Véronis, J. (1998). Introduction to the special issue on word sense disambiguation: The state of the art. *Computational Linguistics*, *24*(1), 1–40.

Joachims, T. (1998). Text Categorization with Support Vector Machines: Learning With Many Relevant Features. In *Proceedings of ECML-98*.

Jouis, C. (1993). *Contribution a la conceptualisation et a la Modelisation des connaissances a partir d'un analyse linguistique de textes. Realisation d'un prototype: le systeme SEEK*. PhD thesis, Universite Paris III - Sorbonne Nouvelle.

Kehagias, A., Petridis, V., Kaburlasos, V. G., & Fragkou, P. (2000). A Comparison of Word- and Sense-Based Text Categorization Using Several Classification Algorithms. *Journal of Intelligent Information Systems*, *21*(3), 227–247.

Kushal Dave, Steve Lawrence, D. M. P. (2003). Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In *Proceedings of the Twelfth International World Wide Web Conference, WWW2003*, (pp. 519–528). ACM.

Maedche, A., Pekar, V., & Staab, S. (2002). Ontology learning part one - on discovering taxonomic relations from the web. In *Web Intelligence*. Springer.

Maedche, A. & Staab, S. (2000). Discovering conceptual relations from text. In Horn, W. (Ed.), *Proceedings of the 14th European Conference on Artificial Intellignece (ECAI'2000)*.

Maedche, A. & Staab, S. (2002). Measuring similarity between ontologies. In *Proceedings of the European Conference on Knowledge Acquisition and Management (EKAW)*. Springer.

Maedche, A. & Staab, S. (2004). Ontology learning. In S. Staab & R. Studer (Eds.), *Handbook on Ontologies* (pp. 173–189). Springer.

Missikoff, M., Navigli, R., & Velardi, P. (2002). The usable ontology: An environment for building and assessing a domain ontology. In *Proceedings of the International Semantic Web Conference (ISWC)*.

Pereira, F., Tishby, N., & Lee, L. (1993). Distributional clustering of english words. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, (pp. 183–190).

Poesio, M., Ishikawa, T., im Walde, S. S., & Viera, R. (2002). Acquiring lexical knowledge for anaphora resolution. In *Proceedings of the 3rd Conference on Language Resources and Evaluation*.

Poesio, M. & Vieira, R. (1998). A corpus-based investigation of definite description use. *Computational Linguistics*, *24*(2), 183–216.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, *14*(3), 130–137.

Resnik, P. (1997). Selectional preference and sense disambiguation. In *Proceedings of the ACL SIGLEX Workshop on Tagging Text with Lexical Semantics: Why, What, and How?*

Salton, G. (1989). *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer*. Addison-Wesley.

Schapire, R. E. & Singer, Y. (2000). BoosTexter: A Boosting-based System for Text Categorization. *Machine Learning*, *39*(2/3), 135–168.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, *34*(1), 1–47.

Seguela, P. (2001). *Construction de modeles de connaissances par analyse linguistique de relations lexicales dans les documents techniques*. PhD thesis, Universite de Toulousse.

Staab, S. & Studer, R. (Eds.). (2004). *Handbook on Ontologies*. Springer.

Steinbach, M., Karypis, G., & Kumar, V. (2000). A comparison of document clustering techniques. In *KDD Workshop on Text Mining*.

Velardi, P., Fabriani, P., & Missikoff, M. (2001). Using text processing techniques to automatically enrich a domain ontology. In *Proceedings of the ACM International Conference on Formal Ontology in Information Systems*.

Volz, R., Studer, R., Maedche, A., & Lauser, B. (2003). Pruning-based identification of domain ontologies. *Journal of Universal Computer Science*, *9*(6), 520–529.

Wang, B. B., Mckay, R. I., Abbass, H. A., & Barlow, M. (2003). A comparative study for domain ontology guided feature extraction. In *Proceedings of the 26th Australian Computer Science Conference (ACSC-2003)*, (pp. 69–78)., Adelaide, Australia. Australian Computer Society, Inc.

Wielinga, B. J., Schreiber, A. T., Wielemaker, J., & Sandberg, J. A. C. (2001). From Thesaurus to Ontology. In *Proceedings of the ACM SIGART International Conference on Knowledge Capture*. ACM Press.

Matthias Dehmer

Data Mining-Konzepte und graphentheoretische Methoden zur Analyse hypertextueller Daten

1 Einleitung

Der vorliegende Artikel hat das Hauptziel, eine verständliche Übersicht bezüglich der Einsetzbarkeit von *Data Mining*-Konzepten auf hypertextuellen Daten zu geben, wobei insbesondere graphentheoretische Methoden fokussiert werden. Die Anwendung von klassischen Data Mining-Konzepten, wie z.B. die Cluster- und die Klassifikationsanalyse, auf webbasierte Daten wird als *Web Mining* bezeichnet. Ein Teilbereich des Web Mining, der in dieser Arbeit besonders im Vordergrund steht, ist das *Web Structure Mining*, welches die Aufdeckung und die Erforschung von strukturellen Aspekten webbasierter Hypertextstrukturen zum Ziel hat. Die strukturelle Untersuchung von Hypertexten und speziell deren *graphentheoretische Analyse* hat sich besonders durch die Entwicklung des *World Wide Web* (WWW) zu einem eigenständigen Forschungsbereich im Hypertextumfeld entwickelt. Vergleicht man den aktuellen Forschungsstand dieses Bereiches jedoch aus der Sicht der Informationssysteme im Hypertextumfeld – den Hypertextsystemen – so fällt auf, dass die Entwicklung und Erforschung der Hypertextsysteme deutlich stärker und schneller fortgeschritten ist als die der strukturellen Analyse. Mit der Bedeutung der multimedialen Kommunikation stellen aber gerade graphentheoretische Methoden ein hohes Analysepotenzial zur Verfügung. Es besteht jedoch noch eine Herausforderung in der Entwicklung aussagekräftigerer, graphbasierter Modelle und graphentheoretischer Analysealgorithmen, die webbasierte Dokumentstrukturen ohne großen Strukturverlust verarbeiten können.

Dieser Artikel ist wie folgt strukturiert: In Kapitel (2) wird zunächst eine kurze Zusammenfassung der Grundlagen bezüglich Hypertext und Hypermedia gegeben. Während in Kapitel (3) Data Mining-Konzepte und die Teilgebiete des Web Mining vorgestellt werden, gibt Kapitel (4) einen Überblick über bestehende Arbeiten der graphentheoretischen Analyse von Hypertexten. Kapitel (5) stellt *Struktur entdeckende* Verfahren, die Clusteringverfahren, vor, die hier insbesondere als Motivation zur Anwendung auf Ergebnisse zu sehen sind, welche mit graphbasierten Methoden des Web Structure Mining erzielt werden.

2 Grundlagen von Hypertext und Hypermedia

Ausgehend vom klassischen Buchmedium ist die Struktur, und in der Regel auch die Lesereihenfolge, eines Buches sequentiell. Dagegen ist die Kerneigenschaft von *Hypertext*, dass die textuellen Informationseinheiten, die so genannten *Knoten*, auf der Basis von *Verweisen*, oder auch *Links* genannt, in Form eines gerichteten Graphen, also *nicht linear*, miteinander verknüpft sind (Kuhlen 1991). Die einfachste graphentheoretische Modellierung einer Hypertextstruktur ist die Darstellung als unmarkierter, gerichteter Graph $\mathcal{H} := (V, E)$, $E \subseteq V \times V$. Dabei heißen die Elemente $v \in V$ Knoten von $\mathcal{H}$ und $e \in E$ wird als gerichtete Kante bezeichnet.

Der Hypertext-Begriff lässt eine unterschiedliche Interpretationsform zwischen den Geisteswissenschaften und der modernen Informatik erahnen. So kann man abhängig von der Fachdisziplin und vom Autor durchaus auf unterschiedliche Definitionen des Hypertextbegriffs stoßen, und Hypertext wird somit oft als Technologie, Methode oder Metapher bezeichnet. Tatsächlich wurden in der Literatur unzählige Definitionen und Ausprägungen von Hypertext gegeben, siehe z.B. Charney (1987); Halasz (1987); Oren (1987). Aus diesen Definitionen – wobei die Autoren unterschiedliche Aspekte betonen – stellt Hofmann (1991) dennoch vier wichtige Kernpunkte heraus, die er für eine vollständige Charakterisierung von Hypertext in der Informatik als notwendig ansieht:

- Hypertexte haben die Gestalt von gerichteten Graphen (Netzwerke). Die Knoten enthalten bzw. repräsentieren die Informationen, die durch Verweise, die Links, miteinander verknüpft sind.
- Das Lesen als auch das Schreiben von Hypertext sind nichtlineare Tätigkeiten. Eine Datenstruktur, die diese Vernetzung unterstützt, ist dabei die Voraussetzung.
- Hypertexte sind nur in einem *medialen* Kontext, also maschinenunterstützt denkbar. Direkte Anwendungen davon sind klassische Hypertext- und Onlinesysteme.
- Hypertexte besitzen einen *visuellen* Aspekt. Das bedeutet, dass Hypertext nicht nur ein Konzept der Informationsstrukturierung, sondern auch eine Darstellungs- und Zugriffsform von textuellen Informationen ist.

Auch in der Sprachwissenschaft und in der Linguistik wurde Hypertext als eine neue Form der schriftlichen Sprachverwendung studiert, z.B. Lobin (1999); Stor-

rer (2004). Dabei wurden insbesondere linguistische Aspekte, wie *Kohärenz*- und *Kohäsionsbeziehungen*, in Hypertext untersucht. Eine bekannte Studie in diesem Problemkreis wurde von Storrer (1999) durchgeführt. In dieser Arbeit geht es im Wesentlichen um die Fragestellung, ob die Ergebnisse über Untersuchungen von Kohärenzbildungsprozessen in linear organisierten Texten für den Entwurf von Hypertexten übertragbar sind. Weiterhin wurde die interessante Problemstellung der *automatischen Generierung* von Hypertext aus natürlich sprachigem Text untersucht, insbesondere, wie und unter welchen Kriterien Hypertext automatisiert konstruierbar ist. Ein linguistisches Kriterium, welches als Grundlage zur Generierung von Hypertext aus Texten dient, ist von Mehler (2001) angegeben worden, wobei hier weitere bekannte Arbeiten zur automatischen Generierung von Hypertexten, aus informationswissenschaftlicher Sicht, beschrieben worden sind.

3 Problemstellungen des Web Mining

Durch die Entstehung des World Wide Web ist die Popularität von Hypertext in den neunziger Jahren deutlich gestiegen. Ein sehr bekanntes, modernes Forschungsfeld, das hypertextuelle Einheiten nach vielen Gesichtspunkten untersucht, ist das *Web Mining* (Chakrabarti 2002). Unter dem Begriff Web Mining versteht man genauer die Anwendung von *Data Mining*-Verfahren (Han & Kamber 2001) auf webbasierte, hypertextuelle Daten mit dem Ziel der automatischen *Informationsextraktion* und der Datenanalyse. Daher werden im Folgenden die Bereiche des Data Mining und deren Kernaufgaben vorgestellt. Data Mining Verfahren wurden entwickelt, um die gigantischen Datenmengen in vielen industriellen und wissenschaftlichen Bereichen zu analysieren und damit neues *Wissen* zu gewinnen. Beispielsweise liegen in vielen Unternehmen riesige Mengen von Kundendaten vor, jedoch ist das Wissen über die Anforderungen und über das Verhalten der Kunden oft nicht besonders ausgeprägt. Solche Datenbestände werden dann in *Data Warehouse* Systemen gespeichert und mit Methoden des Data Mining untersucht. Das Ziel einer solchen Untersuchung ist die Entdeckung von statistischen Besonderheiten und Regeln innerhalb der Daten, die beispielsweise für Studien des Kunden- oder Kaufverhaltens eingesetzt werden können. Die Schwerpunkte der Teilbereiche des Data Minings, lassen sich mit der Hilfe der folgenden Übersicht erläutern:

- Die Suche nach *Assoziationsregeln* (Hastie et al. 2001): Ein bekanntes Beispiel ist die so genannte *Warenkorbanalyse*, die zum Ziel hat, aus dem

aktuellen Kaufverhalten Assoziationsregeln für zukünftiges Kaufverhalten abzuleiten.

- Die *Clusteranalyse* (Everitt et al. 2001): Der entscheidende Unterschied zwischen der Clusteranalyse und der *Kategorisierung* ist, dass bei der Clusteranalyse das Klassensystem von vorneherein unbekannt ist. Das Ziel ist die Gruppierung der Datenobjekte in Gruppen (Cluster), so dass sich die Objekte innerhalb eines Clusters möglichst ähnlich und zwischen den Clustern möglichst unähnlich sind. Dabei basiert die Ähnlichkeit zwischen den Objekten auf einem jeweils problemspezifischen Ähnlichkeitsmaß.

- Die *Kategorisierung* (Duda et al. 2000): Sie stellt Verfahren für die Einordnung von Objekten in Kategoriensysteme bereit. Die Kategorisierung stellt mit Hilfe von Zusammenhängen zwischen gemeinsamen Mustern und Merkmalen ein Kategoriensystem für die vorhandenen Objekte her, um dann auf der Basis eines statistischen Kategorisierungsmodells unbekannte Objekte in das Kategoriensystem einzuordnen. Bekannte Kategorisierungsverfahren stammen aus dem Bereich des *Machine Learning* oder basieren z.B. auf *Entscheidungsbäumen*.

- Die *Regressionsanalyse* (Hastie et al. 2001): Die Regressionsanalyse ist ein Verfahren aus der mathematischen Statistik, welches auf Grund von gegebenen Daten einen mathematischen Zusammenhang in Gestalt einer Funktion zwischen zwei oder mehreren Merkmalen herstellt.

Durch die äußerst starke Entwicklung des World Wide Web gewinnt die Anwendung von Data Mining Verfahren auf webbasierte Daten immer mehr an Bedeutung. Während das Allgemeinziel des Web Mining die Informationsgewinnung und die Analyse der Webdaten ist, werden drei bekannte Teilbereiche detailliert unterschieden (Kosala & Blockeel 2000):

- *Web Content Mining*: Das World Wide Web enthält mittlerweile viele Milliarden von Webseiten. Täglich kommen hunderttausende dazu. Das Web Content Mining stellt Methoden und Verfahren bereit, mit deren Hilfe Informationen, und damit neues Wissen, aus dieser Datenflut automatisch extrahiert werden können. Diese Verfahren finden beispielsweise bei der Informationssuche mit *Suchmaschinen* im World Wide Web eine Anwendung. Während bekannte Suchmaschinen wie z.B. `Yahoo` auf einer einfachen textuellen Schlagwortsuche basieren, stellt die Konzeption neuer, besserer Verfahren für die Informationssuche im Bereich des Web

Content Mining immer noch eine große Herausforderung dar. Die aktuellen Suchmaschinen sind nicht in der Lage, *semantische Zusammenhänge* zwischen webbasierten Dokumenten zu detektieren bzw. die Dokumente nach semantischen Gesichtspunkten zu kategorisieren.

- *Web Structure Mining*: Die Aufgabe des Web Structure Mining ist es, strukturelle Informationen von Websites zu erforschen und zu nutzen, um inhaltliche Informationen zu gewinnen, wobei die *interne und externe Linkstruktur* eine wichtige Rolle spielt. Interne Linkstrukturen können mit Auszeichnungssprachen wie `HTML` oder `XML` abgebildet werden und beschreiben innerhalb eines Knotens eingebettete Graphstrukturen. Die externe Linkstruktur beschreibt die Verlinkung der Webseiten untereinander und lässt sich in Form eines hierarchisierten und gerichteten Graphen darstellen. Die Graphstruktur des World Wide Web ist in den letzten Jahren in vielen Arbeiten intensiv untersucht worden (Deo & Gupta 2001), wobei diese Studien zur Entwicklung und Verbesserung von Suchalgorithmen im World Wide Web geführt haben (Kleinberg 1998). Weiterhin sind *Ausgangsgrad-* und *Eingangsgradverteilungen* von Knoten, *Zusammenhangskomponenten* und der *Durchmesser* des WWW-Graphen untersucht worden. Detaillierte Ergebnisse solcher Untersuchungen sind z.B. in Broder et al. (2000); Deo & Gupta (2001) zu finden. Eine sehr bekannte Arbeit, die im Bereich des Web Structure Mining eine wichtige Anwendung innerhalb der bekannten Suchmaschine `Google` gefunden hat, stammt von Kleinberg. In Kleinberg (1998) führte er die Begriffe *Hubs* und *Authorities* ein. Kleinberg bezeichnet Authorities als Webseiten, die aktuelle und „inhaltlich brauchbare" Informationen enthalten, wobei sich diese graphentheoretisch durch hohe Knoten-Eingangsgrade auszeichnen. Dagegen werden Hubs als solche Webseiten bezeichnet, die viele „gute Links" zu gewissen Themengebieten offerieren. Ein guter graphentheoretischer Indikator für potentielle Hubs ist nach Kleinberg ein hoher Knoten-Ausgangsgrad der betrachteten Webseite.

- *Web Usage Mining*: Unter dem Web Usage Mining (Rahm 2000) versteht man die Suche und Analyse von Mustern, die auf das Nutzungsverhalten eines Users schließen lässt. Üblich ist dabei, die Anwendung von Data Mining Verfahren mit dem Ziel, das Zugriffsverhalten mit Hilfe von *Web-Logs* zu protokollieren. Die Ergebnisse solcher Analysen sind für Unternehmen, besonders aber für Online-Versandhäuser aller Art interessant, weil aus ihnen Aussagen zur Effektivität, zur Qualität und zum Optimierungsbedarf

der Websites abgeleitet werden können. Da bei vielbesuchten Websites täglich riesige Datenmengen allein von Web-Logs anfallen, kann der Einsatz von Data Warehouse Systemen notwendig werden, um diese großen Datenmengen zielgerecht und effizient zu verarbeiten.

Die Bedeutung und die Vertiefung des Web Structure Mining, soll in diesem Artikel anhand von zwei weiteren Problemstellungen hervorgehoben werden und zwar 1.) zum Einen im Hinblick auf geplante Arbeiten im Bereich der strukturellen Analyse von webbasierten Hypertexten und 2.) zum Anderen als Motivation für das Kapitel (5):

1. Das Allgemeinziel des Web Structure Mining ist die Erforschung der strukturellen Eigenschaften von webbasierten Dokumentstrukturen und den daraus resultierenden Informationen. An diesem Ziel orientierend, soll an dieser Stelle auf ein Problem aufmerksam gemacht werden, welches bei der inhaltsorientierten Kategorisierung von webbasierten Hypertexten auftritt. Mehler et al. (2004) stellen die Hypothese auf, dass die beiden Phänomene *funktionale Äquivalenz* und *Polymorphie* charakteristisch für webbasierte Hypertextstrukturen sind. Dabei bezieht sich der Begriff der funktionalen Äquivalenz auf das Phänomen, dass dieselbe Funktions- oder Inhaltskategorie durch völlig verschiedene Bausteine webbasierter Dokumente manifestiert werden kann. Der Begriff der Polymorphie bezieht sich auf das Phänomen, dass dasselbe Dokument zugleich mehrere Funktions- oder Inhaltskategorien manifestieren kann. Nach Definition ist die Hypertextkategorisierung (Fürnkranz 2001) aber funktional, das heißt, jede webbasierte Einheit, z.B. eine Webseite, wird höchstens einer Kategorie zugeordnet. Die Ergebnisse der praktischen Kategorisierungsstudie (Dehmer et al. 2004; Mehler et al. 2004) untermauern jedoch die aufgestellte Hypothese, da es zu einer fehlerhaften Kategorisierung im Sinne von extremen Mehrfachkategorisierungen der Webseiten kam. Letztendlich folgt aber aus der auf der Basis des bekannten *Vektorraummodells* (Ferber 2003) durchgeführten Studie, dass diese Modellierung unzureichend ist. Das Ziel bleibt daher eine verstärkte strukturelle Analyse und eine adäquate Modellierung webbasierter Dokumente.

2. Im Hinblick auf die Bestimmung der Ähnlichkeit webbasierter Dokumente fassen *Document Retrieval* Anwendungen die Dokumente als die Mengen ihrer Wörter auf und berechnen auf der Basis des Vektorraummodells deren Ähnlichkeit. Als Motivation für eine graphorientierte Problemstellung innerhalb des Web Structure Mining und für Kapitel (5), wird eine

Methode von (Dehmer et al. 2004; Emmert-Streib et al. 2005) zur Bestimmung der strukturellen Ähnlichkeit skizziert, die nicht auf der vektorraumbasierten Repräsentation beruht, sondern auf der Graphdarstellung von webbasierten Hypertexten. Ausgehend von der automatisierten Extraktion der Hypertexte und einer GXL-Modellierung (Winter 2002) der Graphen, werden hierarchisierte und gerichtete Graphen erzeugt, die komplexe Linkstrukturen berücksichtigen (Mehler et al. 2004). Diese Graphen werden in eindimensionale Knotensequenzen abgebildet. Das Problem der strukturellen Ähnlichkeit von zwei Graphen ist dann gleichbedeutend mit der Suche eines optimalen Alignments dieser Sequenzen (bezüglich einer Kostenfunktion α). Da es sich um hierarchisierte Graphstrukturen handelt, erfolgt die Bewertung der Alignments ebenenorientiert durch die induzierten Ausgangsgrad- und Eingangsgradsequenzen auf einem Level $i, 0 \leq i \leq h$, wobei h die Höhe der Hypertextstruktur bezeichnet. Die Berechnung der Ähnlichkeit erfolgt schließlich über ein Maß, in das die Werte der Ähnlichkeiten von Ausgangsgrad- und Eingangsgradalignments eingehen. Da diese Methode in seiner algorithmischen Umsetzung effizient ist, verspricht sie im Hinblick auf die Verarbeitung von Massendaten ein für das Web Structure Mining hohes Anwendungspotenzial, z.B.:

- Die Bestimmung der strukturellen Ähnlichkeit von webbasierten Dokumentstrukturen, wie z.B. graphbasierte Websitestrukturen in Form von hierarchisierten und gerichteten Graphen oder DOM-Trees (Chakrabarti 2001).
- Suche und struktureller Vergleich von Graphpatterns in webbasierten Hypertextstrukturen bei Fragen der Interpretation von Hypertext-Navigationsmustern.
- Besseres Verständnis der graphentheoretischen Struktur webbasierter Hypertexte.

4 Graphentheoretische Analyse von Hypertextstrukturen

Wie in Kapitel (2) bereits dargestellt, lässt sich die auszeichnende strukturelle Eigenschaft von Hypertext, die Nichtlinearität, in Form eines Netzwerks mit Hilfe einer graphentheoretischen Modellierung beschreiben. Damit liegt die Frage nach der Einsetzbarkeit von graphentheoretischen Analysemethoden auf der Hand. Das vorliegende Kapitel soll einen Eindruck über die Realisierbarkeit graphbasierter Modellierungen und über die Tragfähigkeit der Aussagen geben,

die man mit einfachen graphentheoretischen Modellen, angewendet auf die Hypertextstruktur, erzielen kann. Als erste Motivation für graphorientierte Methoden sei die Analyse des oft zitierten „*lost in hyperspace*"-Problems (Unz 2000) genannt. Aus der Natur der graphbasierten Modellierung, einer hohen Komplexität der vorliegenden Hypertextstruktur, einem fehlenden kontextuellen Zusammenhang der Links und der Tatsache, dass der Navigierende nur einen eingeschränkten Bereich im Hypertextgraph rezipiert, folgt, dass der *Hypertextuser* die Orientierung verlieren kann. Graphentheoretische Analysemethoden, die als Abstraktionswerkzeug zu verstehen sind, werden oft eingesetzt, um das „lost in hyperspace"-Problem besser unter Kontrolle zu halten. Dazu werden graphentheoretische Kenngrößen definiert, die beispielsweise Aussagen über die Erreichbarkeit von Knoten und deren Einfluss im Hypertextgraph treffen (Botafogo & Shneiderman 1991; Botafogo et al. 1992; Ehud et al. 1994). Die Definition von *Indizes* zur Beschreibung typischer Ausprägungen von Hypertextgraphen kann als weitere Motivation für den Einsatz graphentheoretischer Methoden angesehen werden. Beispielsweise können solche Maße von *Hypertextautoren* eingesetzt werden, um den *Vernetztheitsgrad* und die *Linearität* einer Hypertextstruktur zu bestimmen (Botafogo et al. 1992). Eine weitaus tiefer gehende Fragestellung wäre an dieser Stelle, ob man auf der Basis von graphentheoretischen Indizes eine Gruppierung von ähnlichen Strukturen vornehmen könnte, um dann auf ähnliche Funktionen und Qualitätsmerkmale zu schließen. In jedem Fall müssen aber Fragen nach der *Einsetzbarkeit* und der *Interpretierbarkeit* solcher Maßzahlen gestellt werden, die in Kapitel (4.1) kurz diskutiert werden.

Dieses Kapitel gibt im Wesentlichen einen Überblick über die bekannten Arbeiten der graphentheoretischen Analyse von Hypertextstrukturen, wobei es nicht den Anspruch auf Vollständigkeit erhebt. Einerseits werden damit Möglichkeiten vorgestellt wie man mit einfachen graphentheoretischen Mitteln Hypertexte auf Grund charakteristischer Eigenschaften beschreiben und solche Maße auf Probleme der Hypertextnavigation anwenden kann. Andererseits zeigen einige der nachfolgenden Arbeiten die Grenzen von graphentheoretischen Maßzahlen auf, die sich z.B. in der Allgemeingültigkeit ihrer Aussagekraft und in der Interpretierbarkeit ihrer Wertebereiche äußern.

Die in der Fachliteratur existierenden Ansätze und Arbeiten, die sich mit der graphentheoretischen Analyse und Beschreibung von Hypertextstrukturen beschäftigen, verfolgen im Wesentlichen die folgenden Ziele:

- Die strukturelle Beschreibung und Charakterisierung von Hypertexten durch *globale* graphentheoretische Maße. Sie heißen global, weil sie auf

der gesamten Hypertextstruktur definiert sind. Sehr bekannte Beispiele sind die Hypertextmetriken *Compactness* und *Stratum* von Botafogo et al. (1992).

- Die Suche, die Bestimmung und die graphentheoretische Interpretation von Graphmustern in Hypertexten. Solche spezifischen Graphmuster werden oft bei der Beschreibung von Hypertext-Navigationsproblemen (McEneaney 2000; Unz 2000) und im Zusammenhang von Lernproblemen (Noller et al. 2002; Winne et al. 1994) mit Hypertext analysiert und interpretiert.

Die ersten einschneidenden Arbeiten im Bereich der strukturellen Analyse von Hypertexten stammen von Botafogo & Shneiderman (1991); Botafogo et al. (1992); Ehud et al. (1994). In Botafogo et al. (1992) wurden die bekannten Hypertextmetriken Compactness und Stratum definiert, wobei in dieser Untersuchung Hypertextgraphen als unmarkierte, gerichtete Graphen $\mathcal{H} = (V, E)$, $E \subseteq V \times V$, aufgefasst werden. Mit Hilfe der *konvertierten Distanzmatrix*

$$(\mathcal{KDM}_{ij})_{ij} := \begin{cases} w_{ij} & : \text{falls } w_{ij} \text{ existiert} \\ \mathcal{K} & : \text{sonst,} \end{cases} \tag{1}$$

wobei w_{ij} den kürzesten Weg von v_i nach v_j und $\mathcal{K}$ die Konvertierungskonstante[1] bezeichnet, wird Compactness definiert als

$$\mathcal{C} := \frac{(|V|^2 - |V|) \cdot \mathcal{K} - \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} \mathcal{KDM}_{ij}}{(|V|^2 - |V|) \cdot \mathcal{K} - (|V|^2 - |V|)}. \tag{2}$$

$|V|$ bezeichnet die Ordnung (Anzahl der Knoten) des Hypertextgraphs und nach Definition gilt $\mathcal{C} \in [0, 1]$. Es ist $\mathcal{C} = 0 \iff \mathcal{H} = (V, \{\})$. Weiterhin gilt $\mathcal{C} = 1 \iff |E| = |V \times V| - |V|$. $(|V|^2 - |V|) \cdot \mathcal{K}$ ist der Maximalwert der Matrixelemente aus der konvertierten Distanzmatrix. Er wird erreicht, falls $E = \{\}$. $(|V|^2 - |V|)$ ist der minimale Wert der Summe der Matrixelemente und wird erreicht, wenn $\mathcal{H}$ der *vollständige Graph* ist.

Informell ausgedrückt, gibt der Wert für das Gütemaß Compactness bezüglich einer bestimmten Hypertextstruktur Aufschluss darüber, wie „dicht" die Hypertextstruktur vernetzt ist. Ein hoher Compactness-Wert im Sinne von Botafogo sagt aus, dass von jedem Knoten aus jeder andere Knoten leicht erreicht werden kann.

1 Botafogo et al. (1992) setzen in ihren Untersuchungen $\mathcal{K} = |V|$.

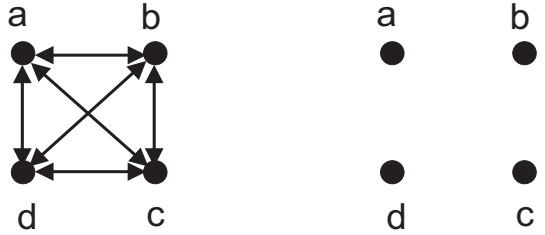

Abbildung 1: Der vollständige gerichtete Graph K_4 und der entsprechende Graph mit der leeren Kantenmenge

Als Beispiel betrachte man die Graphen aus Abbildung (1). Der erste Graph ist der vollständige[2] gerichtete Graph K_4 und nach Gleichung (2) folgt $\mathcal{C} = 1$. Der zweite Graph besitzt die leere Kantenmenge, deshalb $\mathcal{C} = 0$. In Botafogo et al. (1992) wurde von einigen Hypertexten der Compactness-Wert bestimmt und näher untersucht. So besaß beispielsweise die hypertextuelle Beschreibung des Fachbereichs Informatik der Universität Maryland CMSC (Computer Science Department at the University Maryland) einen Compactness-Wert von $\mathcal{C}$=0.53. Für das Buch in Hypertextform HHO (Hypertext Hands On!, Shneiderman & Kearsley (1989)) wurde der Wert $\mathcal{C}$=0.55 ermittelt. Da es sich bei diesen Hypertexten um hierarchische, baumähnliche Graphen handelte lag die Vermutung nahe, dass ein Compactness-Wert von ca. 0.5 typisch für solch strukturierte Hypertexte ist. Die Bildung eines Intervalls, in das man die Compactness-Werte von Hypertexten einordnen kann, um dann aus dem Wert innerhalb dieses Intervalls auf Gütemerkmale wie z.B. „gutes Navigationsverhalten“ zu schließen, ist jedoch aus Gründen der nicht eindeutigen Interpretierbarkeit dieser Hypertextmetrik nicht möglich.

Für die Definition von Stratum betrachte man die Distanzmatrix von $\mathcal{H}$

$$(\mathcal{D}_{ij})_{ij} := \begin{cases} w_{ij} & : \text{falls } w_{ij} \text{ existiert} \\ \infty & : \text{sonst.} \end{cases}$$

$(\hat{\mathcal{D}}_{ij})_{ij}$ sei die Matrix, die man durch Ersetzung der Matrixelemente ∞ durch 0 in $(\mathcal{D}_{ij})_{ij}$ erhält. Botafogo zeigt in Botafogo et al. (1992), dass damit für Stratum die Gleichungen

2 Allgemein wird der vollständige Graph mit n Knoten in der Graphentheorie mit K_n bezeichnet.

$$\mathcal{S} = \begin{cases} \frac{4\sum_{i=1}^{|V|}\left(\left|\sum_{j=1}^{|V|}\hat{\mathcal{D}}_{ji}-\sum_{j=1}^{|V|}\hat{\mathcal{D}}_{ij}\right|\right)}{|V|^3} & : \text{falls } |V| \text{ gerade} \\ \frac{4\sum_{i=1}^{|V|}\left(\left|\sum_{j=1}^{|V|}\hat{\mathcal{D}}_{ji}-\sum_{j=1}^{|V|}\hat{\mathcal{D}}_{ij}\right|\right)}{|V|^3-|V|} & : \text{falls } |V| \text{ ungerade,} \end{cases}$$

bestehen. Nach Definition von $\mathcal{S}$ gilt $\mathcal{S} \in [0,1]$. $\mathcal{S} = 0$ bedeutet, dass die Hypertextstruktur in sich geschlossen und beispielsweise kreisförmig angeordnet ist. $\mathcal{S} = 1$ beschreibt $\mathcal{H}$ in Form einer vollständig linearen Graphstruktur. Wenn man zur gegebenen Hypertextstruktur die zugehörige Hierarchisierung betrachtet, drückt Stratum aus wie tief und linear die hierarchische Struktur ist. Beide Maße, Compactness und Stratum, sind auf unmarkierten gerichteten Graphen definiert und beinhalten keinerlei semantische Relationen des vorgelegten Hypertextes. Botafogo et al. führten diese Untersuchungen durch, in dem sie von allen semantischen, pragmatischen und syntaktischen Typmerkmalen der hypertextuellen Träger abstrahierten. Ein bekanntes Phänomen von *quantitativen* Maßen zur strukturellen Charakterisierung von Hypertexten und zur Beschreibung von Hypertextnavigationsproblemen ist, dass die Ergebnisse solcher Maße oft vom konkret betrachteten Hypertext abhängen und somit mit anderen Messungen schlecht vergleichbar sind. Um diesem Problem entgegen zu wirken, stellte Horney (1993) eine weitere Untersuchung zur Messung von Hypertextlinearität, in Bezug auf die Hypertextnavigation, an. Dabei untersuchte Horney Pfadmuster, die durch bestimmte Aktionen der *User* im Hypertext erzeugt wurden, indem er Pfadlängen ausgehend von Knoten und *Vaterknoten* bestimmte und mittelte. Dieses Prinzip wandte er auf das gesamte Hypertextdokument an und erhielt somit lineare Funktionen für diese Sachverhalte, die er als ein Maß für die Linearität eines Hypertextes definierte.

Abgesehen von Botafogo et al. untersuchten und evaluierten Bra. & Houben (1997) ebenfalls Compactness und Stratum. Da in Botafogo et al. (1992) Compactness und Stratum unter der Annahme definiert worden sind, dass im Hypertextgraph lediglich Vorwärtsbewegungen[3] ausgeführt werden, definierten sie Compactness und Stratum neu, und zwar unter dem Aspekt, Backtracking-Bewegungen[4] im Hypertextgraph durchzuführen. Somit werden durch die modifizierten Maße *navigational Compactness* und *navigational Stratum* von De Bra et al. die Navigationsstrategien von Usern in Hypertextstrukturen besser abgebildet.

3 Im Sinne von Botafogo et al. (1992) heißt das: Falls der Weg von v_i zu v_j nicht existiert, wird er mit der Konvertierungskonstante $\mathcal{K}$ bewertet.

4 Das heißt, man folgt der gerichteten Kante (v_j, v_i), falls man vorher die Bewegung (v_i, v_j) ausgeführt hat.

Ebenfalls wurden die Auswirkungen von Compactness und Stratum auf die Hypertext-Navigation in McEneaney (2000) untersucht, indem aus den schon bekannten Maßen Pfadmetriken definiert und diese empirisch evaluiert wurden. Anstatt der in Botafogo et al. (1992) definierten Matrizen, verwendete McEneaney Pfadmatrizen für die analoge Anwendung dieser Hypertextmetriken. In einer Pfadmatrix repräsentiert ein Matrixelement die Häufigkeit von Knotenübergängen von einem Knoten zu jedem anderen Knoten im Navigationspfad. Diese Pfadmetriken ermöglichen aus Navigationsmustern, dargestellt durch Navigationspfade, die Navigationsstrategien von Hypertextusern zu erkennen.

Außer Compactness, Stratum und den bisher vorgestellten Maßen gibt es noch weitere graphentheoretische Maße im Hypertextumfeld, die jetzt vorgestellt werden. Unz (2000) beschreibt außer Compactness und Stratum die zwei weiteren Maße *Density* und *Kohäsion*. Hauptsächlich gibt Unz (2000) aber einen umfassenden Überblick über das Thema „Lernen mit Hypertext", insbesondere bezogen auf Navigationsprobleme und die Informationssuche in Hypertexten. Density und Kohäsion wurden ursprünglich von Winne et al. (1994) eingeführt, um das Verhalten von Hypertextusern im Zusammenwirken mit bestimmten Lernaktionen, wie z.B. „Einen Text markieren", „Einen Text unterstreichen" und „Eine Notiz machen" im Hypertextsystem `STUDY` graphentheoretisch zu analysieren. Um die spezifischen Graphmuster der Hypertextuser zu gewinnen, bilden Winne et al. formale Sequenzen von ausgeführten Lernaktionen in *Adjazenzmatrizen* ab und erhalten so Graphmuster, die das Benutzerverhalten wiedergeben. Dabei hat eine gewöhnliche Adjazenzmatrix die Gestalt

$$\mathcal{A} := \begin{cases} 1 & : (v_i, v_j) \in E \\ 0 & : \text{sonst}\,. \end{cases}$$

Um dann messen zu können, welche Aktionen bei den Hypertextusern welche Auswirkungen hatten, definierten Winne et al. die graphentheoretischen Maßzahlen

$$\mathcal{D} := \frac{\sum_{i=1}^{|V|} \sum_{j=1}^{|V|} a_{ij}}{|V|^2}, \qquad \text{(Density)} \tag{3}$$

und

$$\mathcal{COH} := \frac{\sum_{i=1}^{|V|} \sum_{j=1}^{|V|} a_{ij} \cdot a_{ji}}{\frac{|V|^2 - |V|}{2}}. \qquad \text{(Kohäsion)} \tag{4}$$

In den Gleichungen (3), (4) bezeichnet a_{ij} den Eintrag in der Adjazenzmatrix in der i-ten Zeile und der j-ten Spalte. $\mathcal{D}$ gibt das Verhältnis von der Anzahl

der tatsächlich vorkommenden Kanten, zur Anzahl aller möglichen Kanten inklusive *Schlingen* an und nach Definition gilt $\mathcal{D} \in [0,1]$. $\mathcal{COH}$ misst den Anteil von zweifach-gerichteten Kanten – das sind Kanten der Form $(v_i, v_j), (v_j, v_i)$ für zwei Knoten $v_i, v_j \in V$ – ohne Schlingen. Der Ausdruck $\frac{|V|^2-|V|}{2}$ gibt die Anzahl aller möglichen Knotenpaare an und es gilt ebenfalls $\mathcal{COH} \in [0,1]$. Aus der Definition der Kohäsion schlossen Winne et al. (1994) nun: Je höher der Wert für die Kohäsion eines betrachteten Graphmusters ist, desto weniger schränkten die Lernaktionen den Hypertextuser ein. Allgemeiner betrachtet kann man diese Maße als benutzerspezifische Präferenzen innerhalb des Graphmusters interpretieren. Weitergehender untersuchten Noller et al. (2002) diese Problematik und entwickelten eine automatisierte Lösung zur Analyse von Navigationsverläufen. Die Navigationsmuster analysierten sie mit graphentheoretischen Mitteln und interpretierten sie ebenfalls als psychologische Merkmale wie z.B. gewisse Verarbeitungsstrategien, konditionales Vorwissen und benutzerspezifische Präferenzen.

Bisher sind hauptsächlich graphentheoretische Maße vorgestellt worden, die zur strukturellen Charakterisierung von Hypertext und zur Interpretation von Graphmustern dienen. Bekannt sind aber auch solche graphentheoretischen Maße, die zur Charakterisierung von Graphelementen konstruiert wurden, vor allem für Knoten in einem Graph. Solche Maße sind in der Fachliteratur allgemeiner als *Zentralitätsmaße* bekannt und finden starke Anwendung in der *Theorie der sozialen Netzwerke*. Sehr bekannte und grundlegende Arbeiten in diesem Bereich findet man bei Harary (1965). *Knotenzentralitätsmaße*, die etwas über die „Wichtigkeit" und „Bedeutsamkeit" von Knoten im Graph aussagen, wurden auch von Botafogo et al. (1992) definiert, bzw. bekannte Maße in einem neuen Kontext angewendet. So definierten sie die Maße

$$\mathrm{ROC}_v := \frac{\sum_{i=1}^{|V|}\sum_{j=1}^{|V|}\mathcal{KDM}_{ij}}{\sum_{j=1}^{|V|}\mathcal{KDM}_{vj}}, \qquad \text{(Relative Out Centrality)}$$

$$\mathrm{RIC}_v := \frac{\sum_{i=1}^{|V|}\sum_{j=1}^{|V|}\mathcal{KDM}_{ij}}{\sum_{j=1}^{|V|}\mathcal{KDM}_{jv}}. \qquad \text{(Relative In Centrality)}$$

Dabei bedeuten $\mathcal{KDM}_{ij}$ wieder die Einträge in der konvertierten Distanzmatrix, die durch die Definitionsgleichung (1) bereits angegeben wurde. Botafogo et

al. wandten das ROC-Maß an, um beispielsweise so genannte *Landmarks* zu kennzeichnen. So werden identifizierbare Orientierungspunkte im Hypertext bezeichnet, weil Landmarks die Eigenschaft besitzen, mit mehr Knoten verbunden zu sein als andere Knoten im Hypertext. Botafogo et al. kennzeichneten damit Knoten mit einem hohen ROC-Wert als Kandidaten für Landmarks. Dagegen sind Knoten mit niedrigem RIC-Wert im Hypertextgraph schwer zu erreichen. Letztlich dienen aber diese beiden Maße zur Analyse von Navigationsproblemen und damit wieder zum besseren Umgang mit dem „lost in hyperspace"-Problem. Als Abschluss dieser Übersicht wird noch eine Arbeit genannt, die ein graphentheoretisches Maß für den Vergleich von Hypertextgraphen liefert. Dafür definierten Winne et al. (1994) das Maß *Multiplicity* für zwei gerichtete Graphen $\mathcal{H}_1$ und $\mathcal{H}_2$ als,

$$\mathcal{M} := \frac{\sum_{i=1}^{|V|} \sum_{j=1}^{|V|} a_{ij} \cdot b_{ij}}{|V|^2} \qquad i \neq j\,. \tag{5}$$

Nach Definition gilt $\mathcal{M} \in [0,1]$ und a_{ij} bzw. b_{ij} bezeichnen in Gleichung (5) die Einträge in der Adjazenzmatrix von $\mathcal{H}_1$ bzw. $\mathcal{H}_2$. Dabei wird hier die Knotenmenge V als gemeinsame Knotenmenge der beiden Graphen angesehen und Multiplicity misst damit die Anzahl der gemeinsamen Kanten beider Graphen, relativ zur Anzahl aller möglichen Kanten. Die Motivation zur Definition von Multiplicity war, individuelle Taktiken und Strategien, die sich in zwei Graphen niederschlagen, vergleichbarer zu machen.

4.1 Kritik und Ausblick

Die Darstellungen in Kapitel (4) zeigen, dass die Wirkung und die Aussagekraft von globalen Maßen zur strukturellen Charakterisierung von Hypertexten und zur Beschreibung von Graphmustern, z.B. Navigationsverläufe, beschränkt ist. Das liegt zum einen daran, dass einige der vorgestellten Maße für speziellere Problemstellungen entwickelt worden sind oder in einer speziellen Studie entstanden sind, z.B. bei Winne et al. (1994). Auf der anderen Seite erlauben quantitativ definierte Maße, z.B. Compactness (Botafogo et al. 1992), keine allgemeingültigen Aussagen über eine verlässliche strukturelle Klassifikation von Hypertextgraphen bzw. über die Güte und Verwendbarkeit solcher Strukturen. Eine aussagekräftige Evaluierung der Maße und die Interpretation einer solchen Auswertung ist in vielen Fällen nicht erfolgt. Ein positiver Aspekt ist die durchgängig klare, einfache mathematische Modellierung und die leichte Implementierbarkeit, indem von komplexeren Typmerkmalen der Knoten und

Links abstrahiert wird. Der negative Aspekt, der daraus unmittelbar resultiert, ist die fehlende semantische Information über solche Typmerkmale, die sich in insbesondere in der mangelnden Interpretierbarkeit von Werteintervallen innerhalb des ausgeschöpften Wertebereichs äußert.

Für den Vergleich von Hypertextgraphen, im Hinblick auf lernpsychologische Implikationen, ist das Maß Multiplicity von Winne et al. (1994), welches über der Kantenschnittmenge definiert ist, vorgestellt worden. Beispielsweise ist mit Multiplicity kein ganzheitlich struktureller Vergleich komplexer Hypertextgraphen möglich, da dieses Maß zu wenig der „gemeinsamen Graphstruktur" erfasst. Wünschenswert wäre für den strukturellen Vergleich solcher Hypertextgraphen ein Modell, welches (i) möglichst viel der gemeinsamen Graphstruktur erfasst und (ii) parametrisierbar ist, d.h. die Gewichtung spezifischer Grapheigenschaften. An dieser Stelle sei nun als Ausblick und Motivation für weitere Arbeiten die automatisierte Aufdeckung und die verstärkte Erforschung der graphentheoretischen Struktur, gerade für webbasierte Hypertexte, genannt, weil (i) bisher wenig über deren charakteristische graphentheoretische Struktur und deren Verteilungen bekannt ist (Schlobinski & Tewes 1999) und (ii) im Hinblick auf anwendungsorientierte Problemstellungen die Graphstruktur ganz besonders als Quelle zur Informationsgewinnung dienen kann. Das bedeutet mit stetig steigender Anzahl der hypertextuellen Dokumente im WWW werden Aufgaben wie die gezielte Informationsextraktion, das automatisierte *webbasierte Graphmatching* und die Gruppierung ähnlicher Graphstrukturen (s. Kapitel (5)) für ein effizientes *Web Information Retrieval* immer wichtiger. In Bezug auf das webbasierte Graphmatching wurde bereits das am Ende des Kapitel (3) skizzierte Verfahren von Dehmer et al. (2004); Emmert-Streib et al. (2005) erwähnt.

5 Verfahren zur Clusterung von Daten

In Kapitel (4) sind bekannte Arbeiten zur graphentheoretischen Analyse von Hypertextstrukturen vorgestellt worden. Dabei kamen auch Maße zur Beschreibung typischer Ausprägungen von Hypertextstrukturen und deren Anwendungen zur Besprechung. Im Hinblick auf die Entwicklung weiterführender graphentheoretischer Methoden im Bereich des Web Structure Mining werden in diesem Kapitel eine Gruppe von *multivariaten Analysemethoden*, die *Clusteringverfahren*, vorgestellt. Bei den im Kapitel (4) dargestellten Verfahren, stand die Charakterisierung typischer Ausprägungen graphbasierter Hypertexte auf der Basis numerischer Maßzahlen im Vordergrund. Im Gegensatz dazu gehören die Clusteringverfahren zur Gruppe der Struktur entdeckenden Verfahren, weil

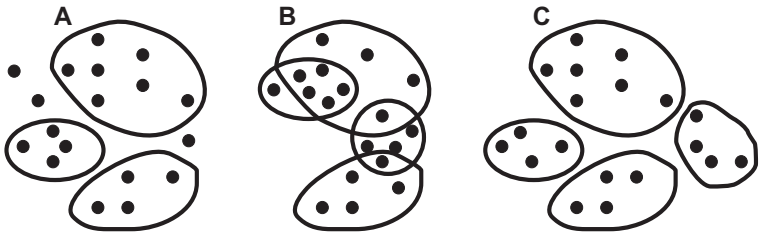

Abbildung 2: A: Disjunkte, aber nicht partitionierende Clusterung mit *nicht gruppierbaren* Objekten. B: Überlappende Clusterung. C: Partitionierende Clusterung

deren Ziel die Aufdeckung von strukturellen Zusammenhängen zwischen den betrachteten Objekten ist. Dabei ist die Einbeziehung mehrerer vorliegender Objektausprägungen die stark auszeichnende Eigenschaft von Clusteringverfahren (Backhaus et al. 2003). Als Motivation zum vorliegenden Kapitel können Clusteringverfahren, als Bindeglied des webbasierten Graphmatching, beispielsweise (i) zur Aufdeckung von *Typklassen*[5] webbasierter Hypertexte eingesetzt werden oder (ii) zur Trennung von strukturell signifikant unterschiedlichen Webseiten.

Clusteringverfahren (Everitt et al. 2001) werden zur Gruppierung (Clusterung) von Objekten angewendet, um möglichst *homogene* Cluster zu erzeugen. In der Regel ist bei Beginn der Clusterung die Anzahl der Cluster und die Clusterverteilung unbekannt, somit auch die Zuordnung der Objekte innerhalb der einzelnen Cluster. Clusteringverfahren sind deshalb im Bereich des *Unsupervised Learning* (Hastie et al. 2001) angesiedelt, weil sie „unüberwacht", also ohne Lernregeln, eine möglichst optimale Clusterung erzeugen sollen. Die Clusterung soll die Kerneigenschaft besitzen, dass ähnliche Objekte in Clustern zusammengeschlossen werden, so dass die Objekte der gefundenen Cluster eine ganz bestimmte *Charakteristik* aufweisen, bzw. jedes Cluster einen eigenen *Typ* repräsentiert. Die Abbildung (2) zeigt verschiedene Varianten von Clusterungen, die entweder je nach Anwendungsfall gewünscht sind oder deren Effekte, z.B. die Überlappung der Cluster, verfahrensbedingt auftreten.

Formeller ausgedrückt lässt sich diese Aufgabe für das Web Mining folgendermaßen beschreiben: Es sei $D := \{d_1, d_2, \ldots, d_n\}$, $\mathbb{N} \ni n > 1$ die Menge der zu clusternden Dokumente. Will man die Clusteraufgabe in voller Allgemeinheit beschreiben, so fasst man die Dokumentenmenge als eine Menge $O := \{O_1, O_2, \ldots, O_n\}$ von unspezifizierten Objekten $O_i, 1 \leq i \leq n$ auf. Eine *Clusterung* C_{fin} ist nun eine k-elementige *disjunkte Zerlegung* von D, al-

5 Z.B. die Klasse der Mitarbeiterseiten innerhalb eines akademischen Webauftritts

so $C_{fin} := \{C_i \subseteq D | 1 \leq i \leq k\}$. Die Cluster C_i sollen dabei die Eigenschaft besitzen, dass basierend auf einem problemspezifischen Ähnlichkeitsmaß $s : D \times D \longrightarrow [0,1]$ (oder Abstandsmaß $d : D \times D \longrightarrow [0,1]$), die Elemente $d \in C_i$ eine hohe Ähnlichkeit zueinander besitzen, wohingegen die Elemente $d, \tilde{d}$ mit $d \in C_i \wedge \tilde{d} \in C_j, i \neq j$ eine geringe Ähnlichkeit zueinander besitzen sollen. Falls die Ähnlichkeits- oder Abstandsmaße bei webbasierten Dokumentstrukturen auf *inneren* (strukturellen) Eigenschaften des Dokuments basieren, ist z.B. die Darstellung gemäß Vektorraummodell oder eine graphentheoretisch basierte Modellierung gemeint.

In der Praxis des Web Mining finden oft *partitionierende-* und *hierarchische* Clusteringverfahren Anwendung, wobei es noch eine Vielzahl anderer Verfahren gibt, z.B. *graphentheoretische*, *probabilistische* und *Fuzzy* Clusteringverfahren (Everitt et al. 2001). Bevor ein Clusteringverfahren angewendet wird, ist es wichtig, die Ausprägungen der Beschreibungsmerkmale zu analysieren, um dann entscheiden zu können, ob zur Beschreibung der Unterschiede zwischen den Dokumenten ein Ähnlichkeits- oder ein Abstandsmaß gewählt wird. Die Frage nach der Lösung einer Clusteraufgabe stellt in der Regel ein Problem dar, da sie von der jeweiligen Anwendung und vom Verwendungszweck der Clusterung abhängt. Oft wählt man eine überschaubare Anzahl der gewonnenen Cluster aus, um sie entweder (i) aus der jeweiligen Anwendungsperspektive zu interpretieren oder (ii) sie mit statistischen Mitteln auf ihre Aussagekraft hin zu überprüfen. Generell sind die Anforderungen an moderne Clusteringverfahren hoch, da sie auf der Basis ihrer Konzeption möglichst viele Eigenschaften besitzen sollen, z.B.:

- geringe Parameteranzahl
- einfache Interpretierbarkeit der Cluster
- gute Eigenschaften bei *hochdimensionalen* und *verrauschten* Daten
- die Verarbeitung von möglichst vielen Datentypen.

Jedoch ist nicht jedes Verfahren, das diese Eigenschaften besitzt, für eine Clusteraufgabe geeignet, weil die Verfahren gewisse Vor- und Nachteile besitzen, die in der Regel von den Daten, dem zugrundeliegenden Ähnlichkeits- oder Abstandsmaß und der Konstruktion des Verfahrens abhängen. Dennoch sind die meisten bekannten Clusteringverfahren theoretisch und praktisch intensiv untersucht worden, so dass sie gut voneinander abgrenzbar sind und somit die Auswahl eines Verfahrens für eine Clusteraufgabe leichter fällt.

5.1 Interpretation von Clusterlösungen

Um die Wirkungsweise von Clusteringverfahren besser zu verstehen, wird zunächst allgemein die Forderung der *Homogenität*, die bereits in Kapitel (5) kurz erwähnt wurde, erläutert. Eine anschauliche Interpretation dieses Maßes, bezüglich eines Clusters C, liefert Bock (1974), indem er die Homogenität als numerische Größe $h(C) \geq 0$ beschreibt, die angibt, wie ähnlich sich die Objekte in C sind, oder anders formuliert, wie gut sich diese Objekte durch ihre charakteristischen Eigenschaften beschreiben lassen. Ausgehend von einer Objektmenge $O = \{O_1, O_2, \ldots, O_n\}$, einem Cluster $C \subseteq O$ und einer Ähnlichkeitsmatrix $(s_{ij})_{ij}$, $1 \leq i \leq n$, $1 \leq j \leq n$, $s_{ij} \in [0,1]$ gibt Bock (1974) ein Maß für die Homogenität von C durch

$$h(C) := \frac{1}{|C| \cdot (|C| - 1)} \sum_{\mu \in I_C} \sum_{\nu \in I_C} s_{\mu\nu} \in [0,1] \tag{6}$$

an, wobei I_C die entsprechende Indexmenge von C bezeichnet. Je größer $h(C)$ ist, desto homogener ist C und umgekehrt. Ist anstatt der Ähnlichkeitsmatrix eine Distanzmatrix $(d_{ij})_{ij}$, $1 \leq i \leq n$, $1 \leq j \leq n$ gegeben, so sind

$$\begin{aligned} h_1^\star(C) &:= \frac{1}{|C| \cdot (|C| - 1)} \sum_{\mu \in I_C} \sum_{\nu \in I_C} d_{\mu\nu}\,, \\ h_2^\star(C) &:= \frac{1}{2|C|} \sum_{\mu \in I_C} \sum_{\nu \in I_C} d_{\mu\nu} \end{aligned}$$

Maße für die *Inhomogenität* und es gilt hier: je kleiner die Werte von $h_i^\star(C), i \in \{1,2\}$ sind, desto homogener ist C und umgekehrt.

Insgesamt gesehen kann oftmals das Ergebnis einer Clusterung als der erste Schritt betrachtet werden, um detailliertes Wissen über die betrachteten Objekte zu erlangen und um darüber hinaus eventuell neue Eigenschaften der Objekttypen zu erkennen. Weiterhin ist es notwendig die Interpretation einer Clusterlösung vor einem speziellen Anwendungshintergrund zu sehen oder das Ergebnis der Clusterung stellt die Grundlage für eine weitergehende praktische Anwendung dar, da eine Clusterlösung für sich isoliert betrachtet, keine weitreichende Aussagekraft besitzt.

5.2 Hierarchische Clusteringverfahren

Um nun die grundlegende Funktionsweise von hierarchischen Clusteringverfahren für das Web Mining zu beschreiben, sei wieder die Dokumentenmen-

ge $D := \{d_1, d_2, \ldots, d_n\}$ mit einem problemspezifischen Ähnlichkeitsmaß $s : D \times D \longrightarrow [0,1]$ (oder Abstandsmaß) betrachtet. Bock motiviert in Bock (1974) hierarchische Clusteringverfahren mit Eigenschaften der Homogenität in Bezug auf partitionierende Clusteringverfahren, bei denen $C_{fin} := (C_1, C_2, \ldots, C_k)$ die Eigenschaften einer *Partition* (siehe Kapitel (5.3)) von D erfüllt. Dabei ist es offensichtlich, dass bei partitionierenden Verfahren (i) größere Homogenitätswerte der Cluster C_i durch eine größere Kardinalität der Menge C_{fin} erreicht werden können, und umgekehrt (ii) sich hohe Homogenitätswerte nur bei hinreichend großer Kardinalität von C_{fin} erreichen lassen. Prinzipiell kann man zwei Arten von partitionierenden Verfahren unterscheiden: (i) die Kardinalität der Menge C_{fin} ist vorgegeben oder (ii) die Homogenitätswerte der Cluster C_i werden von Anfang an durch Schranken gefordert. Dann ergibt sich im ersten Fall die Homogenität der Cluster durch das Verfahren selbst und im zweiten Fall ist k von der geforderten Ähnlichkeit innerhalb der Cluster abhängig. Da aber bei Clusteraufgaben die Zahl k und die Werte der Homogenitätsschranken in der Regel nicht bekannt sind, gelten beide der eben vorgestellten Möglichkeiten als nicht optimal. Hierarchische Clusteringverfahren versuchen dieses Problem dadurch zu lösen, dass sie eine Sequenz von Clusterungen erzeugen mit dem Ziel, dass die Homogenitätswerte der Cluster mit wachsendem k steigt. Weiterhin gilt nach Konstruktion dieser Verfahren, dass immer homogenere Cluster dadurch gebildet werden, dass größere Cluster in kleinere unterteilt werden und dass dieses Prinzip beliebig nach unten fortgesetzt wird. Generell werden bei hierarchischen Clusteringverfahren *divisive* (top-down) oder *agglomerative* (bottom-up) Clusteringverfahren unterschieden, wobei sich in der Praxis die agglomerativen Verfahren durchgesetzt haben. Chakrabarti (2002) gibt eine Vorschrift in *Pseudocode* an, aus der die wesentlichen Konstruktionsschritte von agglomerativen Verfahren leicht zu erkennen sind:

1. Die initiale und damit die feinste Partition von D ist $C_{fin} = \{C_1, C_2, \ldots, C_n\}$, wobei $C_i = \{d_i\}$.
2. `while` $|C_{fin}| > 1$ `do`
3. Wähle $C_i, C_j \in C_{fin}$ und berechne den Abstand $\alpha(C_i, C_j)$
4. Streiche C_i und C_j aus C_{fin}
5. Setze $\gamma = C_i \cup C_j$
6. Füge γ in C_{fin} ein
7. `od`

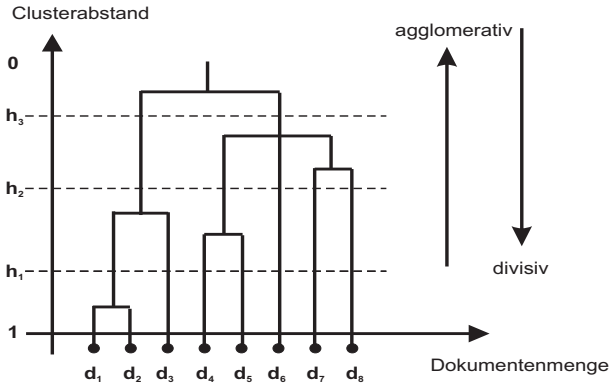

Abbildung 3: Dendrogramm für eine Clusteraufgabe mit acht Dokumenten. Die gestrichelten Linien deuten die gewählten Homogenitätsstufen an.

Das Ergebnis einer Clusterung mit hierarchischen Verfahren lässt sich als *Dendrogramm* visualisieren. Ein Dendrogramm einer fiktiven Clusterung zeigt die Abbildung (3). Dabei lassen sich nun auf jeder gewünschten Homogenitätsstufe h_i die Cluster ablesen und strukturell miteinander vergleichen. Man erkennt in Abbildung (3) deutlich ein auszeichnendes Merkmal eines agglomerativen Clusteringverfahrens: Auf der untersten Ebene stellen die Dokumente einelementige Cluster $\{d_1\}, \{d_2\}, \ldots, \{d_8\}$ dar; mit fallender Homogenität werden die Cluster auf den Ebenen immer gröber, bis sie zu einem einzigen verschmolzen werden, welches alle Dokumente enthält. Ein weiteres wichtiges Merkmal eines hierarchischen Clusteringverfahrens liegt darin, dass Dokumente, die auf der Basis eines Ähnlichkeitsmaßes als sehr ähnlich gelten, sehr früh zu einem Cluster verschmolzen werden. Das ist aber gleichbedeutend damit, dass der dazugehörige Homogenitätswert h_i im Dendrogramm nahe bei eins liegt. Weiterhin sind die Cluster auf den jeweiligen Homogenitätsstufen im Dendrogramm bezüglich ihrer inneren Struktur interpretierbar, da ein Cluster, das im Dendrogramm über mehrere Homogenitätsstufen in sich geschlossen bleibt, als sehr homogen angesehen werden kann. Wird dagegen ein Dokument erst im letzten oder vorletzten Schritt mit einem Cluster verschmolzen, so muss es auf Grund seiner Merkmale weniger ähnlich sein, als die Dokumente in einem sehr homogenen Cluster. Für das Ergebnis einer Clusteraufgabe, die mit einem hierarchischen Verfahren gelöst werden soll, ist aber auch die Güte der Daten, die Aussagekraft des zugrundeliegenden Ähnlichkeits- oder Abstandsmaßes

und vor allen Dingen die Wahl des Maßes α entscheidend, um die Abstände $\alpha(C_i, C_j)$ zweier Cluster zu berechnen. Ausgehend von einem Ähnlichkeitsmaß $s : D \times D \longrightarrow [0,1]$ und den Clustern C_i und C_j, sind

$$\begin{aligned}
\alpha_{SL}\left(C_i, C_j\right) &:= \min_{d,\tilde{d}} \left\{ s(d,\tilde{d}) \,|\, d \in C_i,\, \tilde{d} \in C_j \right\} \text{ (\textit{Single Linkage})}, \\
\alpha_{AL}\left(C_i, C_j\right) &:= \frac{1}{|C_i||C_j|} \sum_{\tilde{d} \in C_i} \sum_{d \in C_j} s(d,\tilde{d}) \text{ (\textit{Average Linkage})}, \\
\alpha_{CL}\left(C_i, C_j\right) &:= \max_{d,\tilde{d}} \left\{ s(d,\tilde{d}) \,|\, d \in C_i,\, \tilde{d} \in C_j \right\} \text{ (\textit{Complete Linkage})},
\end{aligned}$$

gängige Clusterabstände.

Zusammenfassend formuliert ist die übersichtliche und anschauliche Darstellbarkeit des Ergebnisses in Form eines Dendrogramms als positive Eigenschaft von hierarchischen Clusteringverfahren zu sehen. Das Dendrogramm, welches auch als *Baumstruktur* visualisiert werden kann, verlangt dabei nicht eine Clusteranzahl als Vorgabe, sondern auf jeder Ebene entsteht eine Anzahl von Clustern in natürlicher Weise. Weiterhin sind die einfache Implementation und die gute Interpretierbarkeit der entstehenden Cluster als Vorteile von hierarchischen Verfahren zu werten. Für Daten, bei denen eine hierarchische Struktur zu erwarten ist, sind hierarchische Clusteringverfahren besonders sinnvoll. Da in der Regel diese Kenntnis nicht vorhanden ist, muss das Dendrogramm für den jeweiligen Anwendungsfall interpretiert werden, da die hierarchische Struktur durch den Algorithmus erzwungen wird. Als Nachteil ist die Komplexität von hierarchischen Clusteringverfahren zu sehen, weil die Erzeugung der Ähnlichkeitsmatrix bereits quadratische Laufzeit besitzt und somit für Massendaten problematisch wird. Die Verwendung von verschiedenen Clusterabständen ist ebenfalls ein kritischer Aspekt, da Clusterabstände wie Single Linkage bzw. Complete Linkage oft die Tendenz zur Entartung haben, z.B. die Bildung von besonders großen bzw. kleinen Clustern.

5.3 Partitionierende Clusteringverfahren

In diesem Kapitel werden die Ziele und die grundlegende Wirkungsweise von partitionierenden Clusteringverfahren erläutert. Wieder ausgehend von der Dokumentenmenge D und einem Ähnlichkeitsmaß $s : D \times D \longrightarrow [0,1]$, bildet die Menge $C_{fin} := (C_1, C_2, \ldots C_k)$ eine partitionierende Clusterung von D, falls die Eigenschaften $C_i \cap C_j, i \neq j$ (Disjunktheit) und $\bigcup_{1 \leq i \leq k} C_i = D$ (volle

Überdeckung der Menge D) erfüllt sind. Basierend auf der vorgegebenen Menge D, formulierte Bock Bock (1974) die Hauptaufgabe der partitionierenden Clusteringverfahren als die Suche nach einer disjunkten, also nicht überlappenden, Clusterung, die die obigen Eigenschaften einer Partition besitzt und die auszeichnenden Merkmale der Dokumente optimal wiederspiegelt. Weiterhin schlägt Bock (1974) Ansätze zur Lösung dieses Problems vor, z.B.:

- Bereitstellung von statistischen oder entscheidungstheoretischen Modellen, mit denen die noch unbekannten Cluster und deren Objekteigenschaften als Parameter behandelt und abgeschätzt werden können
- Einführung eines *Optimalitätskriteriums*, auf dem die *lokal optimale* Clusterung maßgeblich basiert
- Initiale Festlegung von Startclustern und anschließende Konstruktion der gesuchten Cluster
- Zuhilfenahme von daten- und anwendungsspezifischen Heuristiken

Bei partitionierenden Verfahren ist die finale Clusteranzahl k bei Beginn der Clusterung nicht bekannt und die Dokumente $d \in D$ werden ausgehend von gewählten Startclustern solange ausgetauscht, bis sich auf Grund eines Abbruchkriteriums eine möglichst lokal optimale Clusterung ergibt. Dagegen liegt bei der hierarchischen Clusterung auf jeder Hierarchiestufe eine eindeutige Menge von Clustern verfahrensbedingt vor, wobei diese Cluster nicht mehr aufgebrochen werden. Das in Theorie und Praxis bekannteste partitionierende Clusteringverfahren ist das $k - means$ Verfahren (Hastie et al. 2001), wobei es in verschiedenen Ausprägungen existiert, die sich meistens in der Art und Formulierung des Optimalitätskriteriums unterscheiden. Da k-means nur für quantitative Eingabedaten konzipiert ist, deren Abstände oft über die quadrierte *Euklidische Distanz* berechnet werden, eignet sich für das *Dokumentenclustering* eine Abwandlung von k-means, das $k - medoids$ Verfahren (PAM=Partitioning Around Medoids, cf. Han & Kamber (2001)). Anstatt von numerischen Startobjekten, die bei Beginn die Clusterzentren repräsentieren, wählt man in k-medoids Objekte (*Medoide*) aus D als Clusterzentren. Im weiteren Verlauf des Verfahrens werden lediglich die Ähnlichkeiten bzw. die Distanzen benötigt, um das Optimalitätskriterium, in Form einer Zielfunktion, und die neuen Medoids zu berechnen. Die wesentlichen Schritte von k-medoids, lassen wie sich nachfolgend formulieren, wobei davon ausgegangen wird, dass die Dokumente $d \in D$ in einer für das Clustering geeigneten Repräsentation vorliegen (Han & Kamber 2001):

1. Wähle zufällig k Dokumente als initiale Medoide und definiere damit die Menge M ($|M| = k$)
2. `while` (no change) `do`
3. Ordne jedes verbleibende Dokument dem nächsten Medoid zu (minimalem Abstand)
4. Wähle zufällig ein Dokument $d_r \in D$, das kein Medoid ist
5. Berechne auf der Basis eines Kostenkriteriums c die Gesamtkosten S des Austauschs von d_r mit dem aktuellen Medoid d_{act}
6. `if` c `then` tausche d_{act} mit d_r um eine neue Menge M von Medoiden zu bilden
7. `od`

Vorteile von partitionierenden Clusteringverfahren wie k-means und k-medoids sind ihr intuitiver Aufbau und die einfache Implementierbarkeit. Als Lösungen liefern solche Verfahren aber nur *lokale Optima*, da mit einer anderen Startkombination eventuell eine bessere Clusterlösung berechnet werden könnte. Um diesem Problem entgegenzuwirken, bietet sich entweder eine Kombination mit anderen Clusteringverfahren oder eine iterierte Anwendung an. Ein Nachteil von beiden Verfahren, k-means und k-medoids, ist offensichtlich die Vorgabe der initialen Clusterzahl k, da diese in der Regel unbekannt ist. Eine weitere Schwäche von k-means ist die mangelnde *Robustheit* des Verfahrens, das heißt das Verhalten bezüglich „Ausreißern“, da bei der Berechnung der quadrierten Euklidischen Distanzen offensichtlich hohe Distanzwerte ermittelt werden und diese die Clusterbildung stark beeinflussen. Dagegen besitzt k-medoids eine schlechtere Komplexität in Bezug auf Massendaten, aber eine bessere Robustheit (Hastie et al. 2001).

5.4 Sonstige Clusteringverfahren

Bisher wurden die hierarchischen und partitionierenden Clusteringverfahren detaillierter vorgestellt, da diese Verfahren aus praktischen Gründen und auf Grund ihrer recht guten Interpretationsmöglichkeiten im Umfeld des Web Mining oft eingesetzt werden. In der Fachliteratur werden jedoch noch viele andere Clusteringverfahren behandelt, siehe z.B. Everitt et al. (2001); Fasulo (1999). Zwei werden im folgenden noch skizziert:

- *Graphentheoretische* Clusteringverfahren: Ausgehend von der Dokumentenmenge D und einem problemspezifischen Abstandsmaß (ein Ähnlichkeitsmaß kann leicht in ein Abstandsmaß umgewandelt werden) $d : D \times D \longrightarrow [0,1]$, wird eine Abstandsmatrix $(d_{ij})_{ij}$, $1 \leq i \leq n$, $1 \leq j \leq n$ induziert, wobei $d_{ij} \in [0,1]$. Diese Struktur kann, graphentheoretisch interpretiert, als ein kanten-markierter, vollständiger und ungerichteter Graph $G_D = (V_D, E_D, f_{E_D}, A_{E_D}), f_{E_D} : E_D \longrightarrow A_{E_D} := \{(d_{ij})_{ij}\}_{1 \leq i \leq n, 1 \leq j \leq n}$ betrachtet werden. Nun interessiert man sich für *Umgebungen* in denen, auf Grund der Abstandswerte d_{ij}, ähnliche Dokumente gruppiert werden und die Menge D somit auf diese Weise geclustert werden kann. Bock (1974) ´ charakterisiert dieses Problem mit dem Begriff der d-Umgebung. Er versteht unter der d-Umgebung des Dokuments $d_k \in D$ die Menge der Dokumente $d_i \in D$, deren Abstandswerte die Ungleichung $d_{ik} \leq d$, $d > 0$ erfüllen. Genauer formuliert, definierte Bock ein Cluster $C \subseteq D$ als d-Cluster falls (i) $C \neq \{\}$, (ii) $\forall\ d_k \in C$ gehört auch die d-Umgebung von d_k zum d-Cluster dazu und (iii) kein Cluster $\tilde{C}$ mit $\tilde{C} \subseteq C$ darf die Eigenschaften (i) und (ii) erfüllen. Man betrachte nun denjenigen Teilgraph $G_D^d = (V_D, E_D^d)$, $E_D^d = E_D \setminus \{e = \{d_i, d_j\} \mid f_{E_D}(e) > d, \forall\, d_i, d_j \in V_D\}$ von G_D, für dessen Kantenmarkierungen die Ungleichungen $f_{E_D}(e) \leq d, \forall e \in E_D^d$ gelten. Bock bewies, dass die d-Cluster gerade die *Zusammenhangskomponenten* (Harary 1974) des Teilgraphen G_D^d von G_D sind. Die Abbildung (4) zeigt beispielhaft für eine Menge $D = \{d_1, d_2, \ldots, d_5\}$ mit gegebener Distanzmatrix den vollständigen Graph G_D und den Teilgraph $G_D^{0.5}$. Ein wichtiges und einfaches graphentheoretisches Konstruktionsmittel für die d-Cluster ergibt sich sofort aus dem *minimalen Spannbaum* von G_D. Dabei ist der minimale Spannbaum gerade der Teilgraph B_D mit den Eigenschaften: (i) B_D ist ein Baum (Harary 1974), (ii) B_D enthält alle Knoten aus G_D und (iii) die Summe seiner Kantenmarkierungen fällt minimal aus. Die Konstruktionsmethode des minimalen Spannbaums und die anschließende Gewinnung der d-Cluster wird ausführlich in Bock (1974) beschrieben. Weitere graphentheoretische Clusteringverfahren werden in Fasulo (1999) vorgestellt. Je nach Anwendungsfall werden auch *Dichte-basierte* Clusteringverfahren verwendet, die auf Grund ihrer Konstruktionsweise sehr verwandt zu graphentheoretischen Verfahren sind. Sie werden in Fasulo (1999); Han & Kamber (2001) näher beschrieben. Mehler (2002) stellt einen Algorithmus zur *perspektivischen* Clusterung ausgehend von so genannten Kohäsionsbäumen vor, die insbesondere der automatischen Textverlinkung dienen.

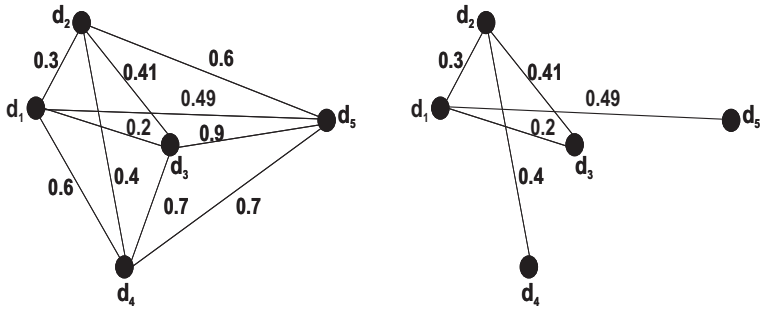

Abbildung 4: $|D| = 5$. Der vollständige Graph G_D und sein Teilgraph $G_D^{0.5}$

- *Probabilistische* Clusteringverfahren: Chakrabarti (2002) beschreibt Probleme des Clustering für webbasierte Dokumente in Bezug auf das Vektorraummodell. Algorithmen im *Web Information Retrieval* setzen voraus, dass die Elemente im Dokumentraum zufälligen Prozessen unterliegen, wobei die Verteilungen innerhalb der Dokumente zunächst nicht bekannt sind. Probabilistische Clusteringverfahren ordnen die Objekte mit einer bestimmten Wahrscheinlichkeit einem Cluster zu, dabei ist aber in der Regel die Verteilung der Objekte und die Anzahl der Cluster unbekannt. Ein sehr bekannter Algorithmus im Bereich der probabilistischen Clusteringverfahren ist der EM-Algorithmus (*Expectation Maximization*), der im Wesentlichen auf zwei Schritten beruht: (i) die Bestimmung der Clusterwahrscheinlichkeiten (Expectation) und (ii) die Parameterabschätzung der Verteilung mit dem Ziel, die Wahrscheinlichkeiten zu maximieren (Maximization). Der EM-Algorithmus wird, bezogen auf das Web Information Retrieval, ausführlich in Chakrabarti (2002) erklärt, wobei man weitere Überblicke in Everitt et al. (2001); Fasulo (1999) findet.

6 Ausblick

In diesem Artikel wurden Data Mining-Konzepte besprochen mit dem Ziel, sie auf bestehende und zukünftige Problemstellungen des Web Mining anzuwenden. Hierbei lag die besondere Betonung auf dem Web Structure Mining. Weiterhin wurden bestehende Arbeiten in der graphentheoretischen Analyse von Hypertextstrukturen besprochen.

Im Zuge der webbasierten Kommunikation wäre es für die zukünftige Entwicklung des Web Structure Mining sehr interessant, neuere Ergebnisse in den Bereichen

- Aufdeckung und bessere Beschreibung bestehender webbasierter Graphstrukturen,
- Fortschritte in der adäquaten und aussagekräftigen Modellierung Webbasierter Hypertexte, besonders in Hinsicht auf eine bessere Möglichkeit der inhaltsbasierten Kategorisierung sowie
- neuere und damit leistungsfähigere graphentheoretische Analysealgorithmen für hypertextuelle Graphstrukturen

zu gewinnen. Gerade in dem Umfeld des Web Structure Mining, wo mit graphentheoretischen Methoden und Data Mining-Verfahren Eigenschaften, Ausprägungen und sogar strukturelle Vergleiche hypertextueller Graphstrukturen bestimmt werden, besteht besonderer Bedarf. Insbesondere sind damit graphentheoretische Methoden angesprochen, mit denen eine aussagekräftige Ähnlichkeitsgruppierung, z.B. auf der Basis spezifischer Eigenschaften oder auf der Graphstruktur selbst, möglich ist. Darauf basierend könnten einige anwendungsorientierte Problemstellungen, z.B. die strukturorientierte Filterung und Fragen bezüglich zeitlich bedingter struktureller Veränderungen webbasierter Hypertextstrukturen, besser gelöst werden. Dabei werden einige der Clusteringverfahren, die im Kapitel (5) vorgestellt wurden, zur Lösung solcher Aufgaben beitragen. Betrachtet man aber die Anzahl der heute vorliegenden Clusteringverfahren so erscheint die Auswahl eines geeigneten Verfahrens für den gewünschten Anwendungsfall jedoch nicht leicht. Die Auswahl sollte sich auf jeden Fall an den vorliegenden Daten, am zugrundeliegenden Ähnlichkeitsmaß und an der geplanten Weiterverwendung einer Clusterlösung orientieren. Zur Interpretation einer Clusterlösung sind in Kapitel (5.1) mathematische Verfahren vorgestellt worden. In Hinsicht auf die Clusterung strukturell ähnlicher webbasierter Hypertextstrukturen ist es denkbar, z.B. auch visuelle oder anwendungsbezogene Kriterien als zusätzliche Gütekennzeichen einer Clusterlösung zu definieren. Somit stellt eine Clusterlösung dann kein isoliert betrachtetes Ergebnis dar, sondern dient als Grundlage für die oben skizzierten Anwendungen im Web Structure Mining.

Literatur

Backhaus, K., Erichson, B., Plinke, W., & Weiber, R. (2003). *Multivariate Analysemethoden*. Springer.

Bock, H. H. (1974). *Automatische Klassifikation. Theoretische und praktische Methoden zur Gruppierung und Strukturierung von Daten*. Studia Mathematica - Mathematische Lehrbücher, Vandenhoeck & Ruprecht Verlag.

Botafogo, R., Rivlin, E., & Shneiderman, B. (1992). Structural analysis of hypertexts: Identifying hierarchies and useful metrics. *ACM Transactions on Information Systems*, *10*(2), 142–180.

Botafogo, R. A. & Shneiderman, B. (1991). Identifying aggregates in hypertext structures. In *Proc. of the 3th annual ACM conference on Hypertext*, (pp. 63–74).

Bra., P. D. & Houben, G. J. (1997). Hypertext metrics revisited: Navigational metrics for static and adaptive link structures. `http://citeseer.ist.psu.edu/139855.html` (seen 05/2005).

Broder, A., Kumar, R., Maghoul, F., Raghavan, P., Rajagopalan, S., Stata, R., Tomkins, A., & Wiener, J. (2000). Graph structure in the web: Experiments and models. In *Proc. of the 9th World Wide Web Conference*.

Chakrabarti, S. (2001). Integrating the document object model with hyperlinks for enhanced topic distillation and information extraction. In *Proc. of the 10th International World Wide Web Conference*, (pp. 211–220).

Chakrabarti, S. (2002). *Mining the Web: Discovering Knowledge from Hypertext Data*. San Francisco: Morgan Kaufmann.

Charney, D. (1987). Comprehending non-linear text: The role of discourse cues and reading strategies. In *Proc. of the ACM conference on Hypertext, Hypertext'87*, (pp. 109–120).

Dehmer, M., Gleim, R., & Mehler, A. (2004). A new method of measuring similarity for a special class of directed graphs. Tatra Mountains Mathematical Publications, Submitted for publication.

Dehmer, M., Mehler, A., & Gleim, R. (2004). Aspekte der Kategorisierung von Webseiten. In *GI-Edition - Lecture Notes in Informatics (LNI) - Proceedings, Jahrestagung der Gesellschaft für Informatik*, (pp. 39–43).

Deo, N. & Gupta, P. (2001). World Wide Web: A graph-theoretic perspective. Technical report, Computer Science Technical report, University of Central Florida.

Duda, R. O., Hart, P. E., & Stork, D. G. (2000). *Pattern Classification*. Wiley - Interscience.

Ehud, R., Botafogo, R. A., & Shneiderman, B. (1994). Navigating in hyperspace: Designing a structure-based toolbox. *Commun. ACM*, *37*(2), 87–96.

Emmert-Streib, F., Dehmer, M., & Kilian, J. (2005). Classification of large graphs by a local tree decomposition. In *to appear in in: Proceedings of DMIN'05, International*

Conference on Data Mining, In conjunction with: World Congress in Applied Computing 2005, Las Vegas/USA.

Everitt, B. S., Landau, S., & Leese, M. (2001). *Cluster Analysis*. Arnold Publishers.

Fasulo, D. (1999). An analysis of recent work on clustering algorithms. Technical report, Technical Report 01-03-02, University of Washington, Seatle/USA.

Ferber, R. (2003). *Information Retrieval*. dpunkt.Verlag.

Fürnkranz, J. (2001). Hyperlink ensembles: A case study in hypertext classification. Technical report, University Vienna, Technical Report No. OEFAI-TR-2001-30.

Halasz, F. G. (1987). Reflections on notecards: Seven issues for the next generation of hypermedia systems. In *Proc. of the ACM conference on Hypertext, Hypertext'87*, (pp. 345–366).

Han, J. & Kamber, M. (2001). *Data Mining: Concepts and Techniques*. Morgan and Kaufmann Publishers.

Harary, F. (1965). *Structural models. An introduction to the theory of directed graphs*. Wiley, New York.

Harary, F. (1974). *Graphentheorie*. Oldenbourg Verlag.

Hastie, R., Tibshirani, R., & Friedman, J. H. (2001). *The Elements of Statistical Learning*. Springer.

Hofmann, M. (1991). *Benutzerunterstützung in Hypertextsystemen durch private Kontexte*. PhD thesis, Springer.

Horney, M. (1993). A measure of hypertext linearity. *Journal of Educational Multimedia and Hypermedia*, *2*(1), 67–82.

Kleinberg, J. M. (1998). Authoritative sources in a hyperlinked environment. In *Proceedings of the 9th annual ACM-SIAM Symposium on Discrete Algorithms*, (pp. 668–677).

Kosala, R. & Blockeel, H. (2000). Web Mining Research: A survey. *SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery & Data Mining*, *2*.

Kuhlen, R. (1991). *Hypertext - Ein nicht-lineares Medium zwischen Buch und Wissensbank*. Springer.

Lobin, H. (1999). *Text im digitalen Medium. Linguistische Aspekte von Textdesign, Texttechnologie und Hypertext Engineering*. Westdeutscher Verlag.

McEneaney, J. E. (2000). Navigational correlates of comprehension in hypertext. In *Proc. of the ACM conference on Hypertext*, (pp. 251–255).

Mehler, A. (2001). *Textbedeutung. Zur prozeduralen Analyse und Repräsentation struktureller Ähnlichkeiten von Texten*. Peter Lang, Europäischer Verlag der Wissenschaften.

Mehler, A. (2002). Hierarchical orderings of textual units. In *Proc. of COLING'02, Taipeh/Taiwan*.

Mehler, A., Dehmer, M., & Gleim, R. (2004). Towards logical hypertext structure. a graph-theoretic perspective. In *Proc. of I2CS'04, Guadalajara/Mexico*.

Noller, S., Naumann, J., & Richter, T. (2002). Logpat - Ein webbasiertes Tool zur Analyse von Navigationsverläufen in Hypertexten. `http://www.psych.uni-goettingen.de/congress/gor-2001` (seen 05/2005).

Oren, T. (1987). The architecture of static hypertext. In *Proc. of the ACM conference on Hypertext, Hypertext'87*, (pp. 291–306).

Rahm, E. (2000). Web Usage Mining. *Datenbank-Spektrum*, *2*(2), 75–76.

Schlobinski, P. & Tewes, M. (1999). Graphentheoretische Analyse von Hypertexten. `http://www.websprache.uni-hannover.de/networx/docs/networx-8.pdf` (seen 05/2005).

Shneiderman, B. & Kearsley, G. (1989). *Hypertext Hands On!: An introduction to a new way of organizing and accessing information*. Addison Wesley.

Storrer, A. (1999). Kohärenz in Text und Hypertext. In L. H. (Ed.), *Text im digitalen Medium. Linguistische Aspekte von Textdesign, Texttechnologie und Hypertext Engineering* (pp. 33–65). Wiesbaden/Germany: Westdeutscher Verlag.

Storrer, A. (2004). Text und Hypertext. In L. H. (Ed.), *Texttechnologie. Perspektiven und Anwendungen*. Wiesbaden/Germany: Stauffenburg Verlag.

Unz, D. (2000). *Lernen mit Hypertext. Informationsuche und Navigation*. Waxmann Verlag.

Winne, P. H., Gupta, L., & Nesbit, L. (1994). Exploring individual differences in studying strategies using graph theoretic statistics. *The Alberta Journal of Educational Research*, *40*, 177–193.

Winter, A. (2002). Exchanging Graphs with GXL. `http://www.gupro.de/GXL` (seen 05/2005).

Autorenverzeichnis

Stephan Bloehdorn
Forschungsgruppe Wissensmanagement
Institut für Angewandte Informatik und
Formale Beschreibungsverfahren - AIFB
Universität Karlsruhe (TH)
76128 Karlsruhe
sbl@aifb.uni-karlsruhe.de

Philipp Cimiano
Forschungsgruppe Wissensmanagement
Institut für Angewandte Informatik und
Formale Beschreibungsverfahren - AIFB
Universität Karlsruhe (TH)
76128 Karlsruhe
cimiano@aifb.uni-karlsruhe.de

Matthias Dehmer
Fachgebiet Telekooperation
Fachbereich Informatik
Technische Universität Darmstadt
Hochschulstr. 10
64289 Darmstadt
dehmer@tk.informatik.tu-darmstadt.de

Andreas Hotho
Fachgebiet Wissensverarbeitung
FB 17 Mathematik / Informatik
Universität Kassel
Wilhelmshöher Allee 73
34121 Kassel
hotho@cs.uni-kassel.de

Edda Leopold
Knowledge Discovery Group
Fraunhofer-Institut für
Autonome Intelligente Systeme (AiS)
Schloß Birlinghoven
53754 Sankt Augustin
edda.leopold@ais.fraunhofer.de

Alexander Mehler
Computerlinguistik und Texttechnologie
Fakultät für Linguistik
und Literaturwissenschaft
Universität Bielefeld
Postfach 10 01 31
D-33501 Bielefeld
alexander.mehler@uni-bielefeld.de

Andreas Nürnberger
Arbeitsgruppe Information Retrieval
Institut für Wissens- und Sprachverarbeitung
Otto-von-Guericke-Universität Magdeburg
Universitätsplatz 2
39106 Magdeburg
nuernb@iws.cs.uni-magdeburg.de

Gerhard Paaß
Knowledge Discovery Group
Fraunhofer-Institut für
Autonome Intelligente Systeme (AiS)
Schloß Birlinghoven
53754 Sankt Augustin
gerhard.paass@ais.fraunhofer.de

Steffen Staab
ISWeb - Information Systems and
Semantic Web
Institut für Informatik
Universität Koblenz-Landau
Postfach 201 602
56016 Koblenz
staab@uni-koblenz.de

Christian Wolff
Medieninformatik
Institut für Medien-, Informations-
und Kulturwissenschaft
Universität Regensburg
93040 Regensburg
christian.wolff@sprachlit.uni-regensburg.de