5.3 Volltextsuche und Text Mining

Jochen Dörre, Peter Gerstl und Roland Seiffert

In diesem Kapitel stellen wir grundlegende Konzepte der **Volltextsuche** (engl.: *Information Retrieval*, IR) und des **Text Mining** vor. Beide Themen sind in den letzten Jahren sehr populär geworden, insbesondere durch die zunehmende Bedeutung des Internets (und von Intranets in Organisationen), das einhergeht mit einem dramatischen Anwachsen der für einen Benutzer verfügbaren Datenmenge. So wird zum Beispiel die Größe des sichtbaren Internets im Februar 2001 auf mehr als 1 Mrd. Seiten geschätzt. Zählt man die nur indirekt zugängliche, z.B. nur über Benutzer-Login oder über Anfragen an Datenbanksysteme erreichbare, Information mit, so ist das Datenvolumen bestimmt zehnmal so groß.

Volltextsuche

Volltextsuchverfahren sind die erste Antwort auf das entstandene Problem: Wie finde ich denn in diesen Informationsgebirgen das Dokument, das mir bei meiner augenblicklichen Aufgabe weiterhilft? Das Prinzip ist denkbar einfach: bei einer Analyse aller Texte, die später suchbar sein sollen, werden die vorkommenden relevanten Termini und ihre Positionen ermittelt und in geeigneten Datenstrukturen, dem **Index**, als sog. **Indexterme** abgespeichert. Das ist vergleichbar mit dem Schlagwortverzeichnis am Ende eines Buches. Eine Anfrage an das System entspricht einem Nachschlagen im Index und dem Aufzählen aller Zitatstellen, die diesen Term beinhalten.

Allerdings trägt diese Analogie in Wirklichkeit nicht sehr weit. Selbst das dickste Buch ist nur winzig verglichen mit dem Internet. Auch werden in einem redaktionell erstellten **Schlagwortverzeichnis** nur wenige, ausgewählte Termini aufgeführt. Eine **Suchmaschine** indiziert aber (fast) alle vorkommenden Wörter als Indexterme.[1] Man stelle sich eine große Bibliothek mit einem Schlagwortverzeichnis vor, das alle Wörter aus allen Büchern auflistet.

Andererseits können von einer Suchmaschine sehr leicht komplexe Bedingungen ausgewertet werden, die bei manueller Suche viel zu aufwändig wären. Zum Beispiel kann eine Bedingung „finde alle Zeitungsartikel (Einschränkung der Dokumentart) aus der Zeit vor 1998 (zeitliche Einschränkung über Metadaten) in denen Kohl im gleichen Absatz (text-strukturelle Einschränkung) wie Spende oder DM genannt wird“ sehr effizient ausgewertet werden.

Text Mining zur Strukturierung großer Textkollektionen

Ein Problem, das hier sichtbar wird, ist, dass es bei wachsenden Datenmengen immer präziserer Anfragen bedarf, um noch ausreichend auf die wirklich relevanten Dokumente zu fokussieren. Da aber gleichzeitig der Benutzerkreis der Systeme immer mehr ausgeweitet wird, kann nicht vorausgesetzt werden, dass

[1] Die Tatsache, dass der gesamte Text indiziert wird und nicht wie bei früheren Bibliothekssystemen nur ausgewählte Schlüsselworte, hat den Namen „Volltextsuche“ geprägt.

die Benutzer über das hierzu nötige Wissen verfügen. Daher müssen Benutzerschnittstellen helfen, den Kontext für Anfragen herzustellen, beispielsweise dadurch, dass eine Taxonomie präsentiert wird, in der zunächst in Richtung auf das Interessengebiet navigiert werden kann. Die Taxonomie übernimmt hier die Funktion der inhaltlichen Organisation der Bibliothek in Bereiche. Suchanfragen können dann in einem engeren Kontext ausgewertet werden.

Die manuelle Informationsaufbereitung ist aber in der Regel bei den großen Datenmengen und der hohen Dynamik der Daten nicht machbar. Nehmen wir an, dass ein Intranet so organisiert werden soll, dass jedes Dokument mindestens einer Kategorie in einer Unternehmenstaxonomie zugeordnet ist und mit den relevanten Schlüsselworten verschlagwortet wird und dass jeweils eine kurze Zusammenfassung verfügbar ist. Optimistisch geschätzt braucht ein Mitarbeiter dann mindestens 10 Minuten pro Dokument, um diese Information manuell bereitzustellen. Wenn z.B. ein mittleres Intranet 1 Mio Dokumente umfasst und nur 1000 täglich hinzukommen oder sich ändern, dann bräuchte man zu Beginn ca. 20000 Personenarbeitstage - was ungefähr der Arbeitszeit von 100 Personen über 1 Jahr entspricht - und dann zum laufenden Betrieb täglich ca. 20 Personenarbeitstage, um mit der Veränderung Schritt zu halten.

Es ist also klar, dass solche Aufgaben soweit wie möglich automatisiert werden müssen. Die Text Mining-Verfahren Textklassifikation, Schlüsselwortextraktion und automatische Textzusammenfassung, die hier und in anderen Kapiteln dieses Buches beschrieben werden, helfen in dem obigen Szenario.

Text Mining für die Gewinnung von Erkenntnissen

Data Mining wird oft definiert als der Prozess zur Extraktion von gültiger, bislang unbekannter und verständlicher Information aus großen Datenbanken. Die bekanntesten Verfahren sind *Clustering* zur Segmentierung der Daten in Gruppen mit ähnlichen Eigenschaften ohne Vorgabe der Segmentierungskriterien, *Classification* und *Prediction* zur Analyse und Vorhersage von Werten für Datenelemente basierend auf einem automatisch aus dem Datenbestand erzeugten Modell, oder *Associations* zur Erkennung von Korrelationen zwischen Datenelementen. Die Verwendung von Data Mining-Werkzeugen z.B. im Marketing oder der Kundenbetreuung, ist inzwischen sehr weit verbreitet und erfolgreich.

Im Unterschied zum Data Mining arbeitet **Text Mining** nicht auf den strukturierten Daten einer Datenbank, sondern auf Textdokumenten. Die Analyseaufgaben und -ziele sind ähnlich. Allerdings muss vor einer Analyse von Zusammenhängen zwischen Dokumenten der Inhalt einzelner Dokumente in geeigneter Form erschlossen werden. Dies ist ein zentraler Teilaspekt des Text Mining.

Text Mining ist eine sehr interessante Erweiterung des Data Mining, da weit mehr Information in textueller Form vorliegt als in strukturierten Datenbanken. Die Anwendungsmöglichkeiten sind sehr vielseitig und wir wollen lediglich zwei Beispiele kurz nennen. Erstens, die Analyse von technischen Dokumentationen und Patenten zur Unterstützung des Patentprozesses in Firmen, zur Erkennung von Technologietrends oder zur Competitive Analysis. Zweitens, die Einbeziehung von Kundenkorrespondenz, z.B. in Form von Email von Kunden, im Customer Relation Management (CRM).

5.3.1 Volltextsuche

Um zu verstehen, wie ein System zur Volltextsuche arbeitet und warum es bestimmte Techniken erfolgreich einsetzt, muss man sich zuerst klar werden über die Rahmenbedingungen, unter denen ein solches System vernünftig eingesetzt wird. Die grundsätzliche Aufgabe besteht einfach darin, aus einer großen Sammlung S von elektronisch gespeicherten Texten, diejenigen Dokumente oder Textpassagen wiederzufinden ("retrieve"), die für eine gegebene Anfrage A auf eine gewisse durch das System näher definierte Weise relevant sind. Wesentlich sind hierbei folgende Charakteristika:

- S ist sehr groß (typischerweise von vielen Tausend Seiten Text bis zu Millionen Seiten Text)
- S ist statisch relativ zur Anfragehäufigkeit (typischerweise viele Anfragen zwischen zwei Änderungen von S) und
- eine Anfrage A soll in sehr kurzer Zeit beantwortet sein (typischerweise Sekundenbereich).

Dreh- und Angelpunkt jedes Volltextsuchsystems ist der **Index**. Anhand der eingangs festgestellten Analogie mit einem Schlagwortregister lassen sich einige grundlegende Konzepte der Volltextsuche, die im Index ihren Niederschlag finden, verdeutlichen. Wie bei diesem handelt es sich beim Volltextindex um ein (alphabetisch) sortiertes Verzeichnis bestimmter Wörter oder Begriffe, genannt **Indexterme**, unter denen **Verweise auf Textstellen** aufgelistet sind.

Zweck des Volltextindex ist es natürlich, das rasche Auffinden von Textstellen zu ermöglichen, die für einen bestimmten Indexterm relevant sind. Dabei ist es im Grunde ohne Belang, ob es sich um den Index eines 300 Seiten starken Buches oder den Index von einer Milliarde Internetseiten handelt.

Wie bereits erwähnt, enthält der Index zu jedem Indexterm eine Liste mit geeigneten Verweisen, wie z.B. Seitenzahlen, auf eine bestimmte Art von „Vorkommen" dieses Terms im Originaltext. Die Natur der Verweise ist bei Volltextsuchsystemen der jeweiligen Aufgabe angepasst, die durch das System erledigt werden soll. So kann der elektronische Katalog einer Internetsuchmaschine sich z.B. darauf beschränken, die (codierten) Webadressen aufzulisten, die einen Term enthalten, während in anderen Anwendungen vielleicht granularere Texteinheiten, bis hin zur genauen Buchstabenposition des Terms, adressiert werden müssen. Zur Vereinfachung sprechen wir im Folgenden bei diesen Verweisen von **Textpositionen**, die in abstrakten **Einheiten des Retrieval** (in einem Schlagwortregister also die Seitenzahlen) angegeben sind.

Zusätzlich zu den Verweisen können weitere Informationen gelistet sein. Um noch einmal die Analogie mit einem Schlagwortverzeichnis zu bemühen, denke man beispielsweise an fettgedruckte Verweise, die eine Textstelle gegenüber den anderen als wichtiger markieren. In einem Volltextindex sind solche zusätzlichen Informationen meist statistischer Natur.

Vom Funktionsprinzip leistet die elektronische Volltextsuche, eben übertragen auf das andere Medium, also dasselbe wie das gedruckte Schlagwortverzeichnis. Wie vergleichbar sind aber die Indexterme selber?

Bei den redaktionell erstellten und für die manuelle Suche gedachten Schlagwortregistern wird nur eine kleine Auswahl von Termini in den Index aufgenommen, die repräsentativ für die Textstellen sind, auf die sie verweisen. Oft werden die Termini durch Kontextangaben auf eine eng umgrenzte Bedeutung fokussiert (ein Beispiel für einen entsprechenden Eintrag wäre "Volltextsuche, elektronische"). Bei Volltextsuchsystemen andererseits sind Indexterme typischerweise automatisch als Wortliste aus den Texten extrahiert und nicht auf "repräsentative" Terme beschränkt. Meist wird in irgendeiner Weise eine (sprachabhängige) **Normalisierung** der Worte auf Stammformen vorgenommen. Außerdem werden inhaltsleere Worte wie Artikel, Konjunktionen, Hilfsverben, etc. als sogenannte **Stoppworte** ausgefiltert und nicht indexiert.[2] Man könnte sagen, dass hier durch Masse und Statistik versucht wird, fehlende semantische Fokussierung und Selektivität aufzuwiegen. Es sollte bei diesem Vergleich aber nicht vergessen werden, dass der Einsatz eines Volltextindex sich eben nicht auf das Nachschlagen einzelner Termini beschränkt, sondern dieses Nachschlagen als Grundfunktion komplexere Funktionen ermöglicht, was die Nützlichkeit der einzelnen Einträge in einem anderen Licht erscheinen lässt.

Die Tatsache, dass in Volltextsuchsystemen die Suche nach Einzeltermen auf komplexe Weise miteinander verknüpft werden kann, macht sie gerade erst zu den mächtigen Recherche-Werkzeugen, die sie sind. Dies wird umso offensichtlicher, wenn die Einzelterme sehr große Treffermengen generieren würden, dagegen durch die Verknüpfung ein starker Filtereffekt erzielt wird. Welche Verknüpfungen im Einzelnen von einem System unterstützt werden und welche Bedeutung diese haben, wird bestimmt durch das **Retrievalmodell**.

Komponenten eines Volltextsuchsystems

Abb. 5.2 gibt einen Überblick über die Komponenten und den Datenfluss in einer typischen Textretrieval-Architektur. Da i.A. Textretrieval nur eine von vielen Funktionen eines größeren Gesamtsystems ist, ist dieses Schaubild als Fragment einer Gesamtarchitektur zu begreifen. Insbesondere können die Textbank selbst sowie die Module zu ihrer Verwaltung nicht nur im Textretrieval-Subsystem, sondern auch in anderen Teilen des Gesamtsystems realisiert sein.

Textretrieval ist nach dieser Architektur ein zweistufiger Prozess. Zuerst muss ein Index konstruiert werden, der alle Dokumente umfasst, die Ziel der Suche sein können. Dann können im zweiten Schritt Suchanfragen gegen den Index gestellt werden. Der Suchprozess selbst läuft dabei ohne Zugriff auf die Originaltexte.

Der erste Schritt, die Indexkonstruktion, ist inhärent ein kostspieliger Prozess. Um dies in Zahlen zu verdeutlichen: der Speicherbedarf allein für den Index ist von einer Größenordnung wie der für die Textkollektion selbst, oder knapp darunter, je nachdem wie umfassend das Vokabular und wie granular die Verweise auf Textpositionen sind. In seltenen Fällen kann mithilfe von Kompressionstechniken und weniger granularen Textpositionen eine Größe unter 10% des unkomprimierten Originaltexts erreicht werden (*Witten et al.* 1994, S. 72–115). Bei der

[2] Es gibt aber auch Systeme, bei denen sich der Index tatsächlich über sämtliche vorkommenden Worte erstreckt, so z.B. bei der Internetsuchmaschine `www.fast.com`.

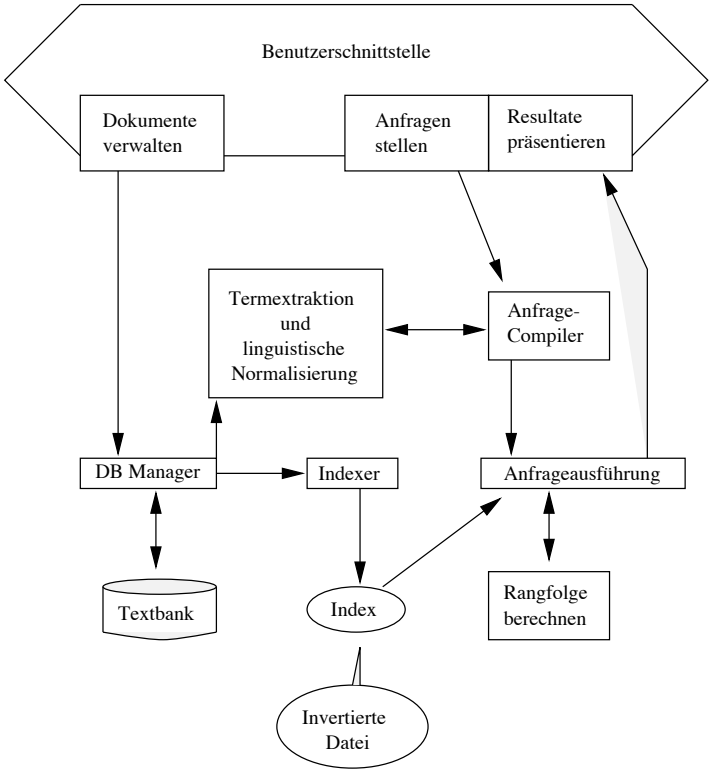

Abbildung 5.2: Die Komponenten eines Volltextsuchsystems

Ressource Rechenzeit schlägt die notwendige Sortierung aller Indexterme und die durch die Menge der Daten erforderliche Speicherung auf Massenspeicher zu Buche. Die Kosten für den Aufbau des Index amortisieren sich jedoch wieder, wenn genügend Anfragen mit dem Index beantwortet werden.

Beim Prozess der Indexkonstruktion spielen Methoden aus zwei sehr unterschiedlichen Gebieten der Informatik eine Rolle. Zum einen wird jeder Einzeltext auf die Liste der in ihm enthaltenen Indexterme reduziert. Dabei kommen computerlinguistische Verfahren zum Einsatz (Stammformbildung etc.). Schon das Auffinden von Wortgrenzen ist bei manchen Sprachen nicht trivial, z.B. bei Thai. Dieser Teil des Prozesses bestimmt letztendlich, welche Terme überhaupt im Index gespeichert werden und für eine Suche zur Verfügung stehen.

Auf der Seite der Sortierung und Abspeicherung der Indexterme und ihrer Positionen kommt es darauf an, mit großen Datenbeständen effizient und sicher umgehen zu können, was mit Verfahren aus dem Arsenal der Datenbanktechnologie bewerkstelligt wird.

Betrachten wir nun die Ausführung einer Suchanfrage. Im Fall einer Ein-Wort-Anfrage muss nur noch die Liste der Textpositionen des gesuchten Worts "nachgeschlagen" werden — fertig. Anfragen können aber auch komplexe Operationen

beinhalten, angefangen bei den Booleschen Operationen `UND, ODER, NICHT`, bis hin zu Einschränkungen der Art, dass zwei Terme in ein und demselben Satz gesucht werden, usf. Bedeutende Modi und Operationen in Suchanfragen sind:

Unscharfe Suche: (engl. fuzzy search) heißt, dass auch Begriffe gesucht werden, die nur ungefähr mit dem Suchbegriff übereinstimmen. So findet man auch Worte mit Rechtschreibvariationen oder Tippfehlern im Originialtext.

Phonetische Suche: sucht nach Worten, die in der Aussprache mit dem Suchbegriff übereinstimmen, wie z.B. Maier mit Meyer.

Phrasensuche: sucht nach Textstellen, an der die Sequenz von Suchbegriffen in derselben Reihenfolge auftritt

Suche in Feldern: beim Indexieren werden Felder wie Autor, Titel, Zusammenfassung usw. festgelegt, auf denen dann eine gezielte Suche ermöglicht wird, wie z.B. "suche Salton in Feld Autor".

Zusätzlich erlauben heutige Retrievalsysteme es, die Resultatsliste nach bestimmten Kriterien zu ordnen. Das bedeutendste Ordnungskriterium ist hier ein Maß der Relevanz eines Resultats für eine Anfrage, mit dem man die Resultate in eine Rangfolge (engl.: ranking) bringt. Bei der Suche in großen Textbanken ist entscheidend für die Retrievalqualität, welche Dokumente durch das Ranking auf die oberen Plätze der Resultatsliste kommen. In Abb.5.2 ist dafür ein eigenes Modul dargestellt, das allerdings in enger Verzahnung mit der Anfrageauswertung arbeitet.

Retrievalmodelle

Mit einem Retrievalmodell definiert man die formale Semantik der Anfrageevaluierung. Durch solch ein Modell wird in abstrakter Weise festgelegt, wie Terme, Operatoren und Modi in einer Anfrage zu interpretieren sind.

Retrievalmodelle gehen der Einfachheit halber von einer logischen Sicht der Dokumente aus. Bei den beiden einfachsten Modellen, die hier vorgestellt werden, genügt es, ein Dokument als Menge (bzw. Multimenge) der es repräsentierenden Indexterme zu betrachten. Wie man genau vom Dokumentinhalt (als Datei bestehend aus Bytes) zu den enthaltenen Indextermen kommt, wird hierbei abstrahiert. Die Einheit des Retrieval ist bei dieser logischen Sicht ein Dokument.

Boolesches Retrievalmodell: Das einfachste Retrievalmodell ist das Boolesche Retrievalmodell (oder auch mengentheoretische Modell). Anfragen bestehen aus Termen, die verknüpft werden können mit Booleschen Operatoren, wie `UND, ODER, NICHT`. Eine Anfrage

```
Schumacher UND Suzuka UND (NICHT Michael)
```

hat dabei die intuitive Bedeutung, dass die gesuchten Dokumente die ersten beiden Terme aber nicht den dritten enthalten sollen. Man kann die Operatoren hier auch mengentheoretisch auffassen, als Operationen auf den durch die Teilanfragen gegebenen Resultatsmengen, wobei `UND` der Schnittmengenoperation, `ODER` der Vereinigung und `NICHT` der Komplementbildung entspricht. Wenn

mit $D[\texttt{Schumacher}]$ die Liste der Dokumente bezeichnet wird, die den Term `'Schumacher'` enthält, und $\mathcal{D}$ die Menge aller indizierten Dokumente bezeichnet, so übersetzt sich die obige Anfrage in den Ausdruck:

$$D[\texttt{Schumacher}] \cap D[\texttt{Suzuka}] \cap \mathcal{D} \setminus D[\texttt{Michael}]$$

So schön die Vorteile seiner einfachen und intuitiv verstandenen Anfragesprache auch sein mögen, stellt sich in der Praxis gerade die Basis seiner Einfachheit, die Zweiwertigkeit, als schwerwiegender Nachteil des Booleschen Modells heraus. Es gibt nur "relevant" oder "irrelevant", eine weitergehende Abstufung wird von diesem Modell nicht getroffen. Das führt in der Praxis oft zu dem unbefriedigenden Verhalten, dass man entweder mit Treffern überhäuft wird oder keine Treffer bekommt. Ein partielles Erfüllen einer Anfrage ist nicht vorgesehen. Wenn z.B. bei einem Dokument nur zwei der drei Teilbedingungen aus der obigen Anfrage wahr sind, ist es genauso "irrelevant" für die Anfrage, wie ein Dokument, das keine der Bedingungen erfüllt.

Um diese Blindheit für partielle Treffer zu überwinden, werden sogenannte erweiterte Boolesche Modelle eingesetzt. Dabei nimmt man an, dass jede Anfrage jedem Dokument einen "Relevanzwert" zuordnet. Meist wird verlangt, dass dies eine Zahl zwischen 0 und 1 ist, die man als Relevanzwahrscheinlichkeit auffasst. Zu jedem Booleschen Verknüpfungsoperator legt man noch eine Relevanzfunktion fest, die den Relevanzwert für den Gesamtausdruck aus den Werten für die Teilausdrücke berechnet. Die Relevanzfunktion für `UND` sollte z.B. nur dann einen hohen Wert (Wahrscheinlichkeit) liefern, wenn beide Eingabewerte hoch waren. Die Minimumfunktion ist ein möglicher Kandidat. Der mit der Minimumfunktion ausgestattete `UND`-Operator ist zwar geeignet, wahrscheinlichkeitsbehaftete Aussagen über Anfragerelevanz eines Dokuments sinnvoll zu verknüpfen, es leistet aber nicht, was oben im Hinblick auf partielle Erfüllung von `UND`-Aussagen gefordert wurde. Dazu muss eine Funktion gewählt werden, die auch dann noch ein von 0 verschiedenes Resultat zulässt, wenn einzelne Argumente 0 sind. Durch Wahl von verschiedenen Funktionen lassen sich somit verschiedene Modi für einen Operator realisieren. Im ersten Fall spricht man vo einem strikten `UND`-Operator, im zweiten Fall von einem unscharfen (oder fuzzy) `UND`. Wenn hier der Anschein entsteht, dass die Interpretation eines fuzzy `UND` eine arbiträre Sache ist, so ist das nicht unbegründet. Eine theoretische Fundierung gibt es hier bisher nicht, und jedes Suchsystem wählt sich seine Interpretation nach eigenem Gutdünken (bzw. Optimierung auf bestimmte benchmark queries) aus.

Vektormodell: Einen völlig anderen Ansatz als das Boolesche Modell wählt das Vektormodell (oder auch Vektorraummodell). Statt in mikroskopischer Sicht auf einzelne (Index-)Terme zu fokussieren, geht man hier eher makroskopisch vor und nimmt an, dass Dokumente charakterisiert werden durch die Statistik ihrer Terme. Wenn eine Anfrage gestellt wird, versucht man diejenigen Dokumente herauszufiltern, deren Termstatistik am besten zur Anfrage "passt". Man benutzt dazu Formeln, mit denen man basierend auf den Termstatistiken ein skalares Maß für die Ähnlichkeit zwischen zwei Dokumenten berechnen kann.

Retrievalsysteme nach dem Vektormodell erzielen immer wieder die besten Ergebnisse. Dies mag erstaunlich klingen, wenn man bedenkt, dass diese Modelle die Bedeutung von Dokumenten allein durch relativ simple Termstatistiken erfassen. Das im Vektormodell zugrunde gelegte Ähnlichkeitsmaß zwischen Dokumenten approximiert eben oft sehr gut ein Maß für thematische Nähe, die in der Intention einer Suchanfrage eine zentrale Rolle spielt.

Im Vektormodell wird eine logische Sicht der Dokumente angenommen, bei der ein Dokument eine Multimenge ist. Meist geht man noch einen Schritt weiter und verallgemeinert ein Dokument zu einer Abbildung von Indextermen auf beliebige positive reelle Zahlen, genannt Gewichte. D.h. ein Dokument wird ausgedrückt durch einen Vektor von Gewichten $(w_1, w_2, \ldots, w_t)$, wobei t die Anzahl aller Indexterme der Kollektion ist und $w_j \geq 0$ das Gewicht zu Indexterm k_j.

Betrachten wir z.B. den letzten Satz als ein Dokument und nehmen als Gewichte jeweils die Häufigkeiten der Worte. Stoppworte werden nicht berücksichtigt. Es müssen also Vektorelemente für die vorkommenden Indexterme `0`, `Anzahl`, `ausgedrückt`, `Dokument`, `Gewicht`, `...` auf die entsprechenden Häufigkeiten `1`, `1`, `1`, `1`, `2`, `...` gesetzt werden. Alle anderen Vektorelemente sind 0.

So betrachtet sind Dokumente Vektoren im t-dimensionalen Vektorraum, der durch die Indexterme (die Dimensionen) aufgespannt wird. Beim Vergleich dieser Vektoren abstrahiert man von ihrer absoluten Länge, da nur die relative Gewichtung der einzelnen Indexterme von Belang ist. Dementsprechend verwendet man als gängiges Ähnlichkeitsmaß für Vektoren den Winkel zwischen ihnen, bzw. noch einfacher zu berechnen, den Kosinus des Winkels. Im Vektormodell wird dieses Maß genutzt, um einen Grad der Ähnlichkeit zwischen der Anfrage und jedem Dokument der Kollektion zu berechnen.

Wir definieren für das Vektormodell $\mathcal{V}$ folgendes. Eine Anfrage ist (wie ein Dokument) ein t-dimensionaler Vektor von Gewichten $\vec{q} = (w_{1,q}, w_{2,q}, \ldots, w_{t,q})$. Dokument d_i ist repräsentiert durch den Vektor $\vec{d_i} = (w_{1,i}, w_{2,i}, \ldots, w_{t,i})$.

Die Bewertungsfunktion $rank : \mathbf{D} \times \mathbf{Q}_{\mathbf{K}}^{\mathcal{V}} \mapsto [\mathbf{0}, \mathbf{1}]$ für beliebige Anfragen definiert sich dann wie folgt:

$$rank(d_j, q) \quad = \quad \frac{\vec{d_j} \cdot \vec{q}}{|\vec{d_j}| \times |\vec{q}|} \quad = \quad \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,q}^2}}$$

Im Gegensatz zum Booleschen Modell gibt es im Vektormodell keine unterschiedlichen Operatoren, aus denen eine Anfrage geformt werden kann. Es gibt nur eine Liste von Indextermen mit Gewichten, die wir in erster Näherung als Häufigkeiten auffassen können. Die Suchanfrage liefert diejenigen Dokumente, deren Gewichtsvektoren dem der Anfrage am ähnlichsten sind, im Sinne von geringstem Winkel. Man abstrahiert also von den Längen der Vektoren — in der Formel die zwei Wurzelausdrücke im Nenner. Anschaulich gesprochen vergleicht man nicht absolute, sondern relative Häufigkeiten der Terme.

In der Praxis werden oft Varianten des reinen Vektormodells eingesetzt, die eine effizientere Berechnung ermöglichen, z.B. wenn Teile der Summe durch Schätzungen ersetzt werden.

TF/IDF: Wesentlich für eine gute Retrievalqualität in diesem Modell ist, dass die Relevanz von Termen für ein Dokument in den Termgewichten ihren Niederschlag findet. Wenn ein Term eine größere Bedeutung für ein Dokument hat, soll er auch mit einem höheren Gewicht im Vektor vertreten sein. Eine ganze Klasse von Ranking-Funktionen, die auf einfache Weise die Relevanz der Terme bestimmt, wurde unter dem Namen *TF/IDF* bekannt. Sie basieren auf zwei Grundannahmen. Erstens muss bei einem Dokument, das einen bestimmten Term häufiger enthält als ein anderes, dieser Term eine größere Bedeutung haben, und zweitens sollte ein Term, der in der Gesamtkollektion seltener auftritt als andere, höher bewertet werden — **Termhäufigkeit** (TF für *term frequency*) mal **inverse Dokumenthäufigkeit** (IDF für *inverse document frequency*).

Eine beliebte TF/IDF-Formel wird in *Salton* 1989 beschrieben. Bezeichnen wir mit $w_{i,j}$ das Gewicht von Term j in Dokument i, mit $t_{i,j}$ die Anzahl der Vorkommnisse von Term j in Dokument i, mit f_j die Anzahl der Dokumente, die j enthalten und mit N die Anzahl der Dokumente in der Kollektion, dann berechnet sich das Gewicht als $w_{i,j} = t_{i,j} \cdot log\frac{N}{f_j}$.

Bewertung der Retrievalqualität

Wenn Suchergebnisse als Listen von Rangfolgen vorliegen, dann ist nicht von vornherein offensichtlich, nach welchen Kriterien die Güte des jeweiligen Resultats berechnet werden kann. Ist etwa eine Antwort besser, die zwar nur einen kleinen Teil der relevanten Dokumente enthält, aber diese in richtiger Reihenfolge, gegenüber einer Antwort mit allen relevanten, aber auch vielen nicht-relevanten Dokumenten, oder gegenüber einer mit mehr relevanten in falscher Reihenfolge?

In der Praxis hat es sich bewährt, die Güte einer Suchmaschine entlang der zwei Dimensionen **Präzision** und **Vollständigkeit** (engl. *precision* and *recall*) zu messen. Je nach Anwendungsfall kann dann auf das eine oder das andere mehr Wert gelegt werden.

Mit Präzision eines einzelnen Suchergebnisses bezeichnet man den Anteil der Dokumente am Suchergebnis, die tatsächlich relevant sind in Bezug auf die Gesamtgröße des Ergebnisses. Die Vollständigkeit des Ergebnisses ist der Anteil der relevanten Dokumente im Suchergebnis in Bezug auf die Menge aller für diese Suche relevanten Dokumente. Angenommen für eine gegebene Suche liefert die Suchmaschine 10 Dokumente, wovon 8 tatsächlich relevant sind, aber die Dokumentkollektion enthält weitere 72 relevante Dokumente so ist die Präzision dieses Ergebnisses $P = 8/10 = 80\%$ und die Vollständigkeit $V = 8/80 = 10\%$.

Um der Tatsache Rechnung zu tragen, dass eine Suchmaschine Resultate in sortierter Rangfolge liefert, werden Präzision und Vollständigkeit nicht nur der Gesamtliste, sondern für jedes Anfangsstück des Resultats berechnet. Man erzeugt daraus eine Kurve, die die Präzision bei verschiedenen Stufen der Vollständigkeit angibt. Um über viele Anfragen zu mitteln, wird meist die Methode der "mikro-gemittelten interpolierten 11-Punkt-Präzision" verwendet. Dabei bildet man an den 11 Vollständigkeitsstufen 0%, 10%, ..., 100% jeweils den Mittelwert der an diesen Punkten interpolierten Präzisionen.

Um Präzision und Vollständigkeit in einem einzigen Maß zu kombinieren, wird oft das sogenannte F-Maß, das geometrische Mittel von Präzision $P(j)$ und Vollständigkeit $V(j)$ der ersten j Resultate, verwendet (mit β wird die Gewichtung von P gegenüber V bestimmt; in der Regel setzt man $\beta = 1$):

$$F(j) = \frac{1 + \beta^2}{\frac{\beta^2}{V(j)} + \frac{1}{P(j)}}.$$

5.3.2 Text Mining

Der Schwerpunkt bei der Volltextsuche liegt auf der Eingrenzung eines Informationsbedürfnisses. Verfahren aus dem Bereich des Text Mining können hierbei hilfreich sein, indem sie bei der Analyse der Suchresultate helfen und Hinweise zur Verbesserung der Suchanfrage geben. Darüber hinaus lassen sich Text Mining Verfahren einsetzen, um, vergleichbar mit der Analyse umfangreicher Datenbestände im Data Mining, neue Zusammenhänge zu entdecken und für planerische Tätigkeiten aufzubereiten.

Im Folgenden werden wir einige Teilaspekte des Text Mining erläutern, die als Grundlage für eine Fülle von Text Mining-Verfahren dienen, von denen wir hier nur einen kleinen Ausschnitt exemplarisch beleuchten können. Diese Aspekte sind: (a) Analyse von Einzeltexten, (b) Merkmalsextraktion, (c) Analyse von Textkollektionen, (d) Maß für die inhaltliche Distanz zwischen Texten.

Analyse von Einzeltexten

Eine wichtiger Teilaspekt der Einzeltextanalyse besteht darin, die textuellen Einheiten zu extrahieren und zu normalisieren, die für weitergehende Arbeitsschritte erforderlich sind. Analog zum Vorbereitungsschritt bei der Analyse strukturierter Daten im Data Mining, bei dem Formate auf eine einheitliche Form gebracht werden, Inkonsistenzen und Inkorrektheiten – soweit möglich – behoben oder entsprechend markiert werden, kann man die Analyse von Einzeltexten als Vorverarbeitungsschritt betrachten, der die relevanten Daten in einheitlicher und kompakter Form für verschiedene Arten der weitergehenden Analyse bereitstellt. Teilaufgaben dieses Schrittes sind die Erkennung des Datenformates, der verwendeten Zeichencodierung (Codepage) und etwaiger Hinweise auf die Dokumentstruktur (Titel, Paragraphen, ...). Generell können diese Informationen aus sehr unterschiedliche Quellen stammen wie dem Übertragungsprotokoll, bestimmten Kennungen (Dateiname, Datenbankeintrag, ...), oder aus einer Analyse expliziter Markierungen im Dokumenttext (Markup). Im Falle von fehlender Information werden bestimmte Annahmen getroffen. Prioritäten werden benötigt, um im Falle von widersprüchlicher Information die zuverlässigste Quelle zu identifizieren. Die Erkennung der Zeichencodierung hat einen besonderen Stellenwert, weil sie eine Voraussetzung für alle weitergehenden, inhaltsbasierten Analysen ist. In Fällen, in denen keine explizite Quelle für die Zeichencodierung bekannt ist, können einfache Klassifikationsverfahren helfen, diesen Wert zu ermitteln. Das geht dann Hand in Hand mit der Ermittlung der Sprache, in der ein Text verfasst ist.

Üblicherweise verwendet man eine Mischung aus einem Profil typischer Zeichenfolgen einer Sprache ($n-$Gramme) und häufiger kurzer Wörter. Daraus können heutige Verfahren bereits bei Textausschnitten von wenigen hundert Zeilen mit einer Genauigkeit jenseits der 90% eine korrekte Aussage über die vermutliche Zeichencodierung/Sprache eines Textes treffen. Verfahren dieses Typs können auch dazu verwendet werden, Texte von binären Inhalten zu unterscheiden um textuelle Inhalte aus multimedialen Dokumenten herauszufiltern.

Sind Zeichencodierung, Textstruktur und Sprache bekannt, so kann eine weitergehende Textanalyse erfolgen, deren Ergebnis eine **Merkmalsmatrix** ist. Eine Merkmalsmatrix repräsentiert den Textinhalt in einer Form, die sich als Eingabe für eine Reihe statistischer Analyseverfahren eignet.

Die Zeilen der Matrix repräsentieren Merkmale. Sie enthalten einen normalisierten Merkmalsterm, also einen Ausdruck aus der Sprache, in welcher der Text verfasst ist, und eine Reihe von sprachlichen oder statistischen Eigenschaften, die diesem Ausdruck zugeschrieben werden. Das Verfahren, das diese Matrix für einen gegebenen Text ermittelt, wird als *Merkmalsextraktion* bezeichnet.

Merkmalsextraktion: Die einfachste Form der Merkmalsextraktion ist das Zerlegen eines Textes in Worte, auch **Tokenisierung** genannt (vgl. Unterkapitel 4.1). Als Nebenprodukte kann dieser Prozess Satzgrenzen erkennen und Normalisierungen durchführen, wie die Vereinheitlichung von Schreibvarianten auf Zeichenebene.

In den meisten Fällen ist diese Art der Analyse zu schwach, um eine hohe Qualität der nachfolgenden Prozesse sicherzustellen. Das erfordert weitere, tiefergehende Analysen, die mehr sprachspezifisches Wissen, beispielsweise in Form von maschinenauswertbaren Wörterbüchern, einbeziehen. Auf der Grundlage eines solchen Wörterbuches können verschiedene Wortformen auf eine grundlegende Stammform zurückgeführt werden. Eine Spalte in der Merkmalsmatrix ist dann üblicherweise für die Wortklasse vorgesehen, die ja in diesem Falle eine wichtige Termeigenschaft darstellt.

Ein Beispiel soll das illustrieren: ein Text enthält die Worte „mouse“ und „mice“. Die Analyse ergibt für beide Worte dieselbe Stammform „mouse“, im ersten Fall möglicherweise ein Verb oder ein Nomen, im zweiten Fall definitiv ein Nomen. Die Ergebnismatrix enthält in diesem Falle nur eine Zeile für den normalisierten Term „mouse“. In der Spalte über die Vorkommenshäufigkeit steht eine 2 und in der Spalte für die Wortklasse steht, dass es sich bei dem Term wohl um ein Nomen handelt. Die Tatsache, dass es sich bei der ersten Form auch um ein Verb handeln kann ist für die weitere Verarbeitung in der Regel ohne Belang, da es nichts an der Zugehörigkeit zum semantischen Konzept ‘Maus’ ändert.

Zahlenwerte (22, „twenty-two“, ...) und Datumsangaben (13.10, Oct 13, ...) können durch entsprechende Algorithmen ebenfalls auf eine normalisierte Form zurückgeführt werden. Entsprechendes ist für Abkürzungen und Akronyme bei Einbeziehung geeigneter Wissensquellen möglich. Bei Eigennamen kann es komplizierter werden, wenn aus dem Text nicht mehr klar hervorgeht, ob mit „Clinton“ der ehemalige Präsident der Vereinigten Staaten oder ein gleichnamiger Ort in New Jersey gemeint ist. Sind entsprechende sprachliche Indikatoren vorhanden

(„Mr. Clinton“, „President Clinton“, „Bill Clinton“, ...), so lässt sich die Normalisierung mit hoher Wahrscheinlichkeit korrekt durchführen, aber was passiert, wenn der Text nur den Namen „Clinton“ enthält?

An diesem Beispiel wird offensichtlich, dass die Merkmalsextraktion ein grundlegendes Dilemma hat, nämlich, dass ein Merkmal eigentlich ein semantisches Konzept repräsentiert, die in Texten enthaltenen sprachlichen Ausdrücke jedoch oft keine eindeutigen Rückschlüsse auf die betreffenden Konzepte ermöglichen. Verschiedene Ausdrücke können sich auf dasselbe Konzept beziehen und identische Ausdrücke können verschiedene Konzepte bezeichnen. Generell sind sprachliche Ausdrücke in einem gewissen Maße unterspezifiziert, so dass automatische Verfahren immer nur eine Annäherung erreichen können.

Weitere Spalten der Merkmalsmatrix enthalten in der Regel numerische Information über die Häufigkeit, mit welcher die Merkmale innerhalb einer bestimmten Einheit, üblicherweise dem ganzen Text, enthalten sind. Dieser Häufigkeitswert muss zusätzlich mit dem Textumfang normalisiert sein, so dass Zeilen verschiedener Matrizen, die denselben Term enthalten, miteinander vergleichbar sind.

Die weiter unten beschriebenen Verfahren zur Analyse von Textkollektionen führen in der Regel keine eigene Textanalyse durch sondern operieren direkt mit den Merkmalsmatrizen, die die Dokumente der Kollektion repräsentieren.

Anwendungen der Merkmalsextraktion: Direkte Anwendungen von Merkmalsextraktion sind die Hervorhebung (z.b. Unterstreichung) wichtiger Ausdrücke eines Textes sowie die Extraktion von repräsentativen Wörtern (Schlüsselwortextraktion) und Sätzen (automatische Textzusammenfassung – siehe Unterkapitel 5.6).

Indirekte Anwendungen sind solche, die Merkmalsextraktion zum Beispiel im Zusammenspiel mit der Analyse von Textkollektionen als Vorverarbeitung für weitergehende Mining Verfahren nutzen, wie sie im Folgenden beschrieben werden.

Analyse von Textkollektionen

Das Ziel der Analyse von Textkollektionen ist es, signifikante Zusammenhänge oder Unterschiede zwischen Texten explizit zu machen und für die weitergehende Verarbeitung aufzubereiten. Je nach Einsatzgebiet geht es darum, eine große Ansammlung von Dokumenten durch entsprechende Gruppierung mit einer inhaltlich motivierten Struktur zu versehen oder Dokumente in ein vorgegebenenes Schema einzuordnen, wie man es z.B. von Internetportalen kennt.

Bei dem erstgenannten Verfahren, das seinen Ursprung im Data Mining hat, spricht man von **Clustering**. Dieses Verfahren werden wir im Folgenden näher erläutern. Die Zuordnung zu vorgegebenen, in einem separaten Schritt trainierten Kategorien, ist Gegenstand von Unterkapitel 5.4 dieses Buches. Die Abbildungen 5.3 und 5.4 veranschaulichen den wesentlichen Unterschied zwischen den beiden Verfahren. Aus Sicht der verwendeten Algorithmen können sie auch als

Spezialfälle von **überwachtem Lernen** (engl. *supervised learning*) im Falle von Textklassifikation und nicht-überwachtem Lernen (engl. *unsupervised Learning*) im Falle von Clustering charakterisiert werden.

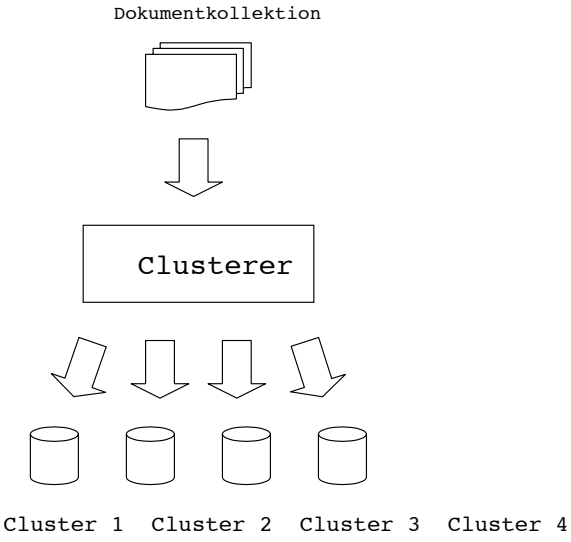

Abbildung 5.3: Clustering bedeutet, dass die Kollektion von Dokumenten in **Gruppen** (Cluster) aufgeteilt wird, die der Algorithmus **dynamisch generiert.**

Bevor wir uns der Beschreibung des Clusteringverfahren zuwenden können, müssen wir noch darauf eingehen, wie Distanzmaße zwischen Texten definiert werden. Ein solches Maß dient dazu, Entscheidungen bezüglich der Zugehörigkeit zu einer bestimmten Klasse von Texten zu treffen.

Distanzmaße zwischen Texten: Die Grundlage vieler in der Praxis verwendeter Distanzmaße ist ein hochdimensionaler **Merkmalsvektor**, der wie folgt ermittelt wird: Alle Texte einer Kollektion werden einer systematischen Merkmalsextraktion unterzogen. Die Summe aller ermittelten Merkmale bildet den **Merkmalsraum.** Durch geeignete Selektion wird dann üblicherweise die Gesamtzahl der zu betrachtenden Merkmale weiter reduziert. Dafür eignen sich sprachspezifische Verfahren, die sogenannte **Stoppwörter** wie Artikel oder Präpositionen eliminieren, oder rein statistische Verfahren, die auf der Herausfilterung relativ nieder- und hochfrequenter Terme beruhen. Nach diesem auch als **Merkmalsreduktion** bezeichneten Schritt hat der Merkmalsraum die Dimension n. Jeder einzelne Text der Kollektion kann nun durch einen Vektor $\vec{v}$ beschrieben werden. Der Wert eines Elements v_j entlang jeder der Dimensionen entstammt der Spalte der das Dokument repräsentierenden Merkmalsmatrix,

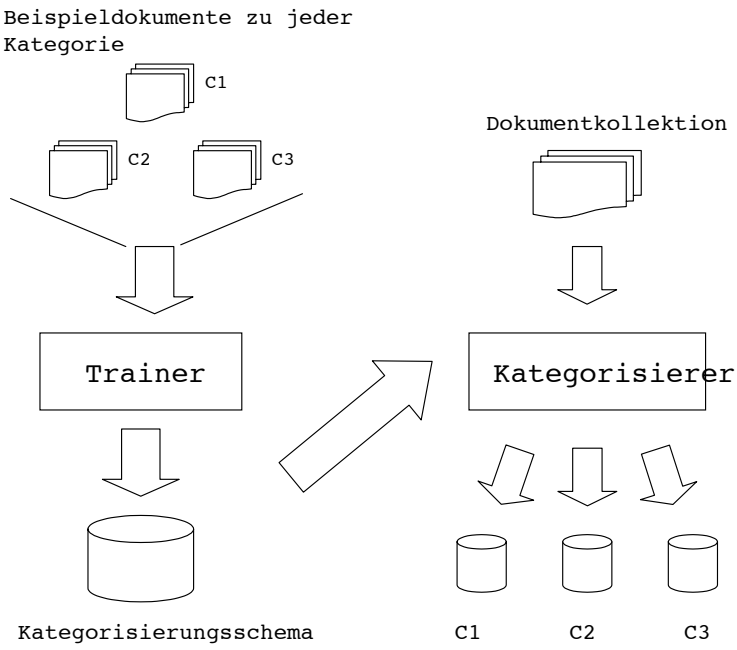

Abbildung 5.4: Kategorisierung bedeutet, dass Dokumente **Kategorien** zugeordnet werden, die in einer benutzerdefinierten Taxonomie **vorgegeben** sind.

welche die Vorkommenshäufigkeit des Merkmals enthält. Der konkrete Wert von v_j hängt oft vom jeweiligen anzuwendenden Verfahren ab. In manchen Fällen reicht es aus, durch 0 oder 1 zu repräsentieren, ob das Merkmal im Text auftritt, in anderen Fällen verwendet man eine absolute oder auch normierte Häufigkeit. Die Normierung dient dazu, die unterschiedliche Länge von Texten zu kompensieren. Die Distanz zweier Texte wird nun anhand der diese Texte repräsentierenden Vektoren berechnet. Einfache Maße sind der Abstand der durch die Vektoren angegebenen Punkte im n–dimensionalen Raum, oder der Winkel zwischen den Vektoren. In der Regel wird der Wert auf einen Bereich zwischen 0 und 1 normiert, beispielsweise durch Verwendung von $cos(\alpha)$ anstelle des Winkels α. Oft wird auch ein inverses Maß benutzt, das dann eher "Nähe" ausdrückt, so dass beispielsweise ein Wert von 1 vollständige Übereinstimmung der Vektoren bedeutet.

Manche Verfahren benutzen einen einzelnen Vektor als Repräsentanten für eine ganze Menge von Vektoren. Meist wird hierfür der sogenannte **Zentroid-Vektor** gewählt, der als der Vektor eines – in der Regel nicht tatsächlich vorhandenen Textes – entspricht, der genau im Zentrum des durch alle Vektoren

aufgespannten Raumes liegt. Anschaulich kann man sich hier einen Mittelwert oder Schwerpunkt vorstellen. Für die weitere Darstellung werden wir, wo dies benötigt wird, einfach die Existenz eines geeigneten Distanzmaßes annehmen.

Clustering: Die Aufgabe des Clustering besteht darin, eine Menge von Texten so zu strukturieren, dass zueinander inhaltlich ähnliche Texte nahe zusammen auftreten, wohingegen die Distanz zwischen inhaltlich verschiedenen Texten groß sein soll. Dazu wird die Textkollektion vom Clusteringverfahren in, üblicherweise disjunkte, Teilmengen partitioniert. Diese Teilmengen nennt man dann Cluster. In der Regel wird noch eine Struktur zwischen den Clustern ermittelt, die ein Distanzmaß zwischen den Clustern ausdrückt. Das Ziel eines jeden Clusteringverfahrens ist es, das Ergebnis so zu optimieren, dass der "Abstand" von Texten innerhalb eines Clusters möglichst minimal und die Distanz zwischen verschiedenen Clustern möglichst maximal ist. Das erste Verfahren, das wir kurz beschreiben, ist ein hierarchisches Clustering (*Maarek und Wecker* 1994). Dessen Ergebnis ist eine Menge von Clustern und ein Baum, wobei die Cluster den Blättern des Baumes zugeordnet sind. Die "Zweige" des Baumes repräsentieren das Distanzmaß zwischen den Clustern. Je kürzer der minimale Pfad im Baum von einem Cluster zum anderen ist, desto stärker ähneln sich die Texte dieser Cluster. Eine äquivalente Interpretation der Struktur ist es, die Cluster selbst als hierarchisch aufzufassen, so dass jeder Knoten im Baum einem Cluster entspricht, das alle Texte, die zu den Clustern der darunter liegenden Knoten gehören, enthält. Das hier beispielhaft betrachtete Verfahren arbeitet von den Blättern zur Wurzel (Bottom-Up). Zunächst wird jeder Text einem einelementigen Blattcluster zugeordnet. Dann wird die Distanz zwischen den Clustern paarweise bestimmt und die beiden jeweils nächsten Cluster werden zu einem neuen Cluster zusammengefasst. Dieser Prozess wird iteriert, bis schließlich nur noch ein Cluster vorhanden ist, das dann alle Texte umfasst. Das Ergebnis ist ein binärer Baum, in dem die Distanz zwischen zwei Texten durch die Länge des kürzesten Pfades zwischen den entsprechenden Blättern repräsentiert wird. Die in einem binären Baum enthaltene sehr reichhaltige Information kann auf einen übersichtlicheren, $n-$ären Baum abgebildet werden, indem nur ein bestimmter Wertebereich der Textähnlichkeit berücksichtigt wird. Der Vorteil des bottom-up Verfahrens liegt in der erreichten optimalen Homogenität der Cluster und der Reihenfolgeunabhängigkeit des Verfahrens. Der größte Nachteil besteht in der ungefähr quadratischen Komplexität des Algorithmus. Damit sind der Skalierbarkeit deutliche, prinzipielle Grenzen gesetzt.

Ein anderer Ansatz ist ein lineares Clusteringverfahren. Im Prinzip wird hier ein Durchlauf über die Texte durchgeführt und bei jedem Dokument entschieden, ob es einem bereits bestehenden Cluster zuzurechnen ist oder ob ein neuer Cluster dafür angelegt werden soll. Das Verfahren ist somit reihenfolgenabhängig. Daher wird der Prozess in der Regel mit einer anderen Reihenfolge der Texte wiederholt und auf Konvergenz geprüft. Zusätzlich wird ein Distanzmaß zwischen den einzelnen resultierenden Clustern berechnet, das dann eine graphartige Struktur über den Clustern ergibt. Der Vorteil dieses Verfahrens ist seine bessere Skalierbarkeit, da die Komplexität nur linear von der – eventuell durch

einen Parameter begrenzten – maximalen Zahl der Cluster und zulässigen Iterationen sowie der Anzahl der Texte abhängt.

Anwendungen des Clustering: Unter einer **Taxonomie** versteht man ein hierarchisches Klassifikationsschema, das dazu geeignet ist, eine Wissensdomäne inhaltlich zu strukturieren. Ein Beispiel für eine sehr bekannte Taxonomie für die Information im Internet ist das Schema, das Yahoo (siehe `www.yahoo.com`) zur Katalogisierung von Internetseiten verwendet. Solch komplexe Taxonomien werden manuell erstellt und gewartet, die Zuordnung von Information zu einzelnen Kategorien kann manuell oder mittels der in Kapitel 5.4 beschriebenen Verfahren auch automatisch oder halbautomatisch durchgeführt werden. Es gibt aber auch Ansätze, die versuchen, mit Text Mining-Methoden inhaltsorientierte Taxonomien automatisch aus einer gegebenen Textkollektion zu erstellen und zu warten (siehe *Dörre et al.* 1999; *Chakrabarti et al.* 1998). Vollautomatische Verfahren benutzen in der Regel die oben beschriebenen Clustering-Verfahren als Grundbaustein. Aufbauend auf dem hierarchischen Clustering, kann man eine Textkollektion zunächst inhaltsorientiert in eine Baumstruktur organisieren. Die Cluster nahe der Blätter in der Baumstruktur zeichnen sich durch hohe Homogenität aus und sind oftmals gute Kandidaten für inhaltsnahe Kategorien einer Taxonomie. Die oberen Ebenen des Baumes sind schwieriger automatisch zu bestimmen. Das Problem hierbei besteht darin, dass nahe der Wurzel einer Taxonomie echte Abstraktionen erwartet werden, die sich nicht mehr notwendigerweise direkt anhand des Vokabulars eines Textes ergeben. Eine echte semantische Analyse der Texte wäre nötig, um automatisch entsprechende Abstraktionen in guter Qualität durchführen zu können. Ein Ansatz wäre, durch Zuhilfenahme geeigneter externer Wissensquellen, beispielsweise in Form von Thesaurus-Information, die Wörter auf echte Oberbegriffe abzubilden und dadurch eine Reanalyse des oberen Teils der Struktur mit relativ einfachen Mitteln durchzuführen. Uns sind allerdings keine vollautomatischen Verfahren bekannt, die hier wirklich überzeugende Ergebnisse erreichen. Die besten Ergebnisse werden im Moment mit halbautomatischen Verfahren erzielt. Ausgehend von einer Textkollektion und möglicherweise einer vorgegebenen Grobstruktur mit beispielhaften Texten wird eine erste Version einer Taxonomie erzeugt. Diese Taxonomie wird überprüft und verändert (beispielsweise können Kategorien aufgespalten oder zusammengefasst werden, Texte einer anderen Kategorie zugeordnet werden und so weiter). Dies wird dann als Basis für eine erneute automatische Analyse verwendet und so die Taxonomie verbessert. Das Ergebnis ist eine qualitativ hochwertige Taxonomie und ein Modell, das dazu benutzt werden kann, neue Dokumente entsprechend zu klassifizieren.

5.3.3 Literaturhinweise

Eine sehr gute Einführung in Information Retrieval mit einem breitangelegten Überblick über die dort verwendeten Konzepte und Algorithmen bietet *Baeza-Yates und Ribeiro-Neto* 1999. Wegen des sehr umfangreichen Glossars und der umfassenden Literaturliste bietet sich das Buch als Nachschlagewerk für ein

tiefergehendes Studium an. Viele der dargestellten Konzepte sind auch wichtige Grundlagen für das Text Mining. Wer sich für mehr Details der Implementierung von Volltextsuche interessiert, dem sei *Witten et al.* 1994 empfohlen.

Für den Bereich Text Mining gibt es unseres Wissens noch kein vergleichbar umfassendes Einführungsbuch. Außer diesem Artikel behandeln noch weitere Beiträge in diesem Buch wichtige Teilbereiche des Text Minings, z.B. Textzusammenfassung (siehe Unterkap. 5.6) oder Textklassifikation (siehe Unterkap. 5.4). Die entsprechenden Literaturlisten geben genügend Hinweise für die Vertiefung des Themas.

Auch finden sich immer häufiger Beiträge zu Text Mining auf wichtigen Konferenzen verwandter Gebiete. Insbesondere zu nennen sind:

- SIGIR (ACM SIGIR International Conference on Research and Development in Information Retrieval): http://www.acm.org/sigir/
- KDD (ACM SIGKDD International Conference on Knowledge Discovery in Databases and Data Mining): http://www.acm.org/sigkdd/
- VLDB (International Conference on Very Large Databases): http://www.vldb.org/