korpus.html

Zur Sammlung, Datenbankbasierten Erfassung, Annotation und Auswertung von HTML-Dokumenten*

Georg Rehm

Zusammenfassung

Ein derzeit in der Konzeptionierungsphase befindliches Forschungsvorhaben untersucht Eigenschaften von Hypertextdokumenten des World Wide Web. Dieser Beitrag beschreibt das zu diesem Zweck aufgebaute Korpus von HTML-Dokumenten, wobei sowohl die Phasen der Datensammlung, -erfassung und -annotation als auch weitergehende Untersuchungen erläutert werden.

9.1. Konzeption des Korpus

Es erscheint zunächst unsinnig, derart dynamische Informationen wie über das World Wide Web (Berners-Lee et al., 1992) erhältliche HTML-Dokumente (Raggett et al., 1999), die auf Knopfdruck aus aller Welt abrufbar sind, in Form eines Korpus zu archivieren. Verschiedene Gründe sprechen jedoch für diese Vorgehensweise: Die informatische, linguistische und computerlinguistische Forschung benötigt zu verschiedenen Zwecken Referenzkorpora (Hawking et al., 1999, Walker, 1999): Im Falle des angesprochenen Forschungsvorhabens sollen rekurrente Strukturen in Webseiten untersucht und klassifiziert werden. Linguisten, die sich mit computervermittelter Kommunikation beschäftigen, benötigen Korpora, die aus Chat-Protokollen, Sammlungen elektronischer Briefe (Runkehl et al., 1998) und auch Webseiten bestehen, um die enthaltenen sprachlichen Strukturen untersuchen zu können (siehe etwa Storrer, 1999). Informatiker können aufgrund statistischer Informationen über die Größe von in verschiedenen Subnetzen angebotenen Dokumenten und Dateitypen bessere Caching-Verfahren für Proxy-Server (Turau, 1998) und Load-Balancing Algorithmen entwickeln.

Seit Januar 2001 entsteht ein Korpus, das die deutschsprachigen Dokumente möglichst vieler frei zugänglicher Webserver deutscher Hochschulen (Wätjen et al., 1998) in Form eines „sprachlichen Schnappschusses“ beinhalten wird. Die Beschränkungen hinsichtlich der akademischen Domäne und der Sprache resultieren aus der Fragestellung des eingangs angesprochenen Forschungsvorhabens.[1]

* Erschienen in: *Proceedings der GLDV-Frühjahrstagung 2001*, Henning Lobin (Hrsg.), Universität Gießen, 28.–30. März 2001, Seite 93–103. `http://www.uni-giessen.de/fb09/ascl/gldv2001/`

[1] Im kommerziellen Bereich des World Wide Web werden häufig professionelle Content Management Systeme einge-

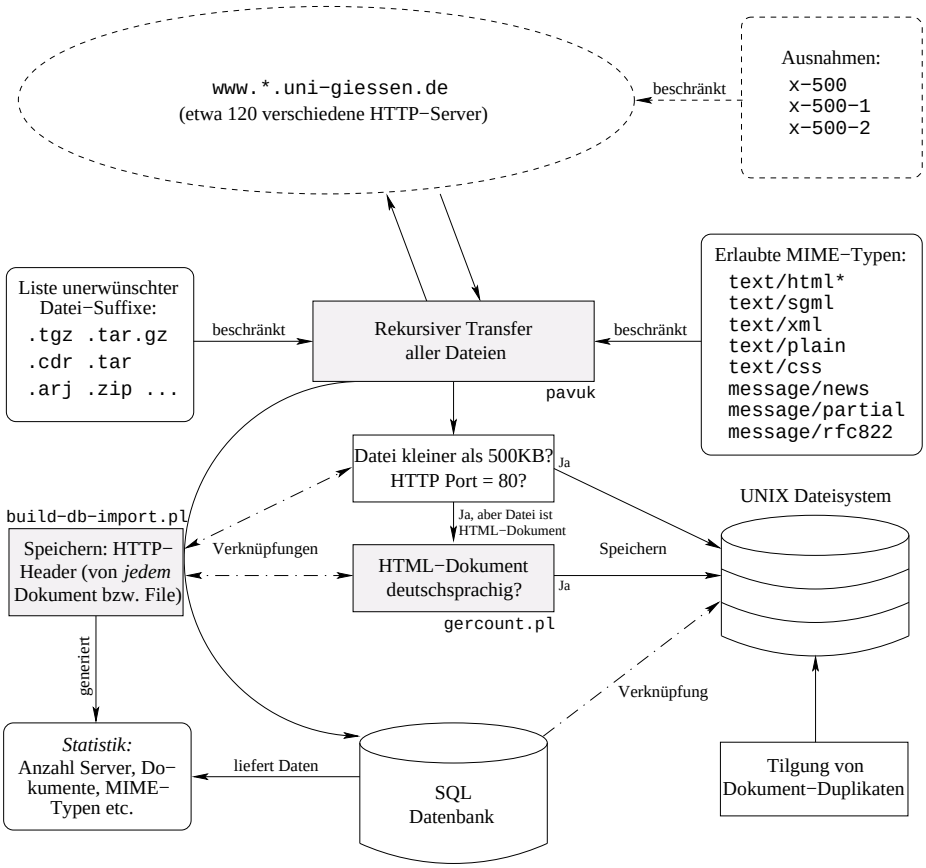

Abbildung 9.1.: Die Sammlung der im Korpus enthaltenen Daten unter Berücksichtigung der eingesetzten Beschränkungen am Beispiel der Domäne `*.uni-giessen.de`

9.2. Datensammlung und Datenerfassung

Die Datensammlung erfolgt automatisch auf der Grundlage einer Liste aller zentralen universitären Einstiegspunkte[2] der zu untersuchenden Domäne, wobei die in das Korpus aufzunehmenden Universitäten bzw. Universitätsstädte per Zufallsprinzip ermittelt werden.

Abb. 9.1 verdeutlicht den Ablauf der Datensammlung[3] sowie die verschiedenen Beschränkungen, die hinsichtlich der in das Korpus aufzunehmenden Daten spezifiziert wurden, am Beispiel der Webserver der Universität Gießen: Die erste der drei zentralen Komponenten stellt das frei verfügbare Werkzeug Pavuk[4] dar. Pavuk ermöglicht das automatisierte und rekursive Übertragen

setzt, um Webdokumente zu pflegen. Diese Template-basierten Systeme erzeugen oftmals identische hypertextuelle Strukturen und sind daher für den Skopus dieser Arbeit irrelevant: „Restriction on the academic part of the web has two reasons: It is relatively clear to define and [...] we can expect a relatively clear and (partially) consistent (pre-)structuring." (Müller, 1999)

[2] Fast alle Hochschulen betreiben neben einem zentralen Webserver, beispielsweise `www.uni-giessen.de`, weitere spezialisierte Server.

[3] Cowie et al. (1998) setzen ein ähnliches System ein: Mit Hilfe eines Erkenners, der 34 verschiedene Sprachen identifizieren kann, können für eine Sprache gezielt im World Wide Web verfügbare Texte gesammelt werden.

[4] Implementiert von Stefan Ondrejicka, `http://www.idata.sk/~ondrej/pavuk/`

von Dateien mithilfe des Hypertext Transfer Protocols (HTTP). Neben einer Startadresse – im Beispiel handelt es sich hierbei um `http://www.uni-giessen.de` – kann Pavuk mit vielfältigen Beschränkungen konfiguriert werden; die wichtigste Beschränkung bezieht sich jeweils auf die Angabe eines Musters derjenigen Adressen, die von Pavuk besucht werden sollen (im Beispiel `.uni-giessen.de`).[5] Drei Server wurden explizit ausgeschlossen, da sie ausschließlich die Daten des internen X.500-Verzeichnisses zur Pflege von Kontaktinformationen der Angehörigen der Universität Gießen enthalten. Damit nicht alle Dateien im Korpus abgelegt werden, spezifizieren wir eine Liste erlaubter MIME-Typen[6], die neben HTML-Dateien weitere für das Forschungsvorhaben wichtige Dokumenttypen enthält. Aufgrund der Tatsache, dass von sehr vielen Webservern Dokumente mit fehlerhaften Angaben bzgl. des `Content-Type` ausgeliefert werden, wird zusätzlich eine umfangreiche Liste nicht erwünschter Dateisuffixe spezifiziert.[7] Diese Beschränkungen gelten ausschließlich für die Ablage einer Datei im Korpus. Für *alle* referenzierten Dateien werden diejenigen Metadaten, die in Form des HTTP-Response Headers (vgl. Fielding et al., 1999, Kapitel 6) vom Webserver geliefert werden, zu statistischen Zwecken lokal gespeichert.[8]

Ein Dokument, das über einen erlaubten MIME-Typ verfügt, kleiner ist als 500 Kilobyte[9], auf dem HTTP-Standardport[10] ausgeliefert wurde und deutschsprachig ist (siehe Abschnitt 9.3, S. 97 für eine Beschreibung dieser zweiten zentralen Komponente), wird im Korpus gespeichert. Die HTML-Dokumente werden dabei bis auf eine Ausnahme nicht modifiziert: Damit ein Betrachten der Dokumente mit einem Webbrowser möglich ist, werden alle eingebetteten URLs so modifiziert, dass sie auf die – sofern im Korpus vorhanden – lokalen Dokumente bzw. die entfernten Dateien verweisen.

Ursprünglich war geplant, die Dateien in einer SQL-Datenbank abzulegen. Anfängliche Tests haben jedoch ergeben, dass bereits einige hunderttausend Dokumente die Performanz und Größe der Datenbank negativ beeinflussen, sodass von diesem Ansatz Abstand genommen werden musste. Stattdessen werden alle im Korpus enthaltenen Daten im UNIX-Dateisystem gespeichert[11], wobei die Verzeichnisstruktur der ursprünglichen Webserver beibehalten wird. Die SQL-Datenbank MySQL (vgl. DuBois, 1999) enthält lediglich die Metadaten des HTTP-Headers sowie einen Verweis auf den voll spezifizierten Dateinamen des Dokuments, sofern es im Korpus

[5] Vereinzelt werden weitere Bereiche manuell ausgeschlossen, etwa fehlerhafte Hyperlinks, die beim Retrieval der Daten zu Endlosschleifen geführt hätten, umfangreiche Datenbanken oder Verwaltungsdateien und -verzeichnisse, die zu Synchronisationszwecken von Microsoft Frontpage angelegt werden (`_vti_bin/*`, `_vti_cnf/*` etc.).

[6] Ein Webbrowser erkennt ein HTML-Dokument nicht an dem Suffix `.html` oder `.htm`, sondern an einer Angabe im HTTP-Header, die ein Dokument als `Content-Type: text/html` kennzeichnet, vgl. Fielding et al. (1999), Freed und Borenstein (1996).

[7] Beispielsweise versenden viele Webserver Archivdateien mit dem Suffix `.zip` fälschlicherweise als Dateien des MIME-Typs `text/plain`, korrekt wäre `application/zip`.

[8] Tab. 9.1, S. 96, stellt die beiden Datenbanktabellen zur Speicherung dieser Metadaten dar. Selten vorkommende Header wie etwa `Warning` oder `Upgrade` werden nicht berücksichtigt. Das Vorhandensein von Headern, die ausschließlich für statistische Untersuchungen interessant sind, wird mithilfe eines Booleschen Wertes markiert.

[9] Durch diesen Schwellwert soll verhindert werden, dass sehr große Dateien unnötig viel Platz im Korpus einnehmen. Des Weiteren kommen derart umfangreiche Dateien in der Realität selten vor: Nur neun HTML-Dateien innerhalb der Domäne `.uni-giessen.de` sind größer als 500 Kilobyte.

[10] HTTP transferiert Dateien in der Grundeinstellung auf Port 80. Webserver, die ihre Daten auf anderen Ports anbieten, sind meist – so haben vorab durchgeführte Experimente ergeben – experimentelle, WAP- oder auch Proxy-Server und daher für das Vorhaben nicht relevant.

[11] Turau (1998) und Walker (1999) sprechen die Problematik von Duplikaten an: Da ein Dokument durchaus mehrere URLs besitzen kann, sollten Duplikate vermieden werden. Wir entfernen alle Duplikate mithilfe des von Phil Karn entwickelten Werkzeugs dupmerge (`http://people.qualcomm.com/karn/code/dupmerge/`), das Duplikate tilgt und durch Verweise auf *ein* verbleibendes Original ersetzt.

Datenbanktabelle `http_header`

Feld	Datentyp	Beispiel
ID	`int(10)`	`72926`
Partieller URI	`text`	`/~g91063/ps/textannotation.ps.gz`
Korpus-Datei	`text`	
Verweis zum Server	`int(10)`	`8`
HTTP Status Code	`smallint(5)`	`200`
Content-Length	`int(10)`	`156432`
Content-Type	`varchar(30)`	`postscript`
Content-Encoding	`tinytext`	`x-gzip`
Content-Language	`tinytext`	
Content-Location	`tinytext`	
Location	`tinytext`	
Date	`datetime`	`2001-01-16 15:48:19`
Expires	`datetime`	
Last-Modified	`datetime`	`1999-07-28 20:41:38`
WWW-Authenticate	*`boolean`*	`0`
Cache-Control	*`boolean`*	`0`
Content-MD5	*`boolean`*	`0`
Pragma	*`boolean`*	`0`
Set-Cookie	*`boolean`*	`0`

Datenbanktabelle `server_info`

Feld	Datentyp	Beispiel
ID	`int(10)`	`8`
Server-Name	`VARCHAR(10)`	`www.uni-giessen.de`
Port	`mediumint(5)`	`80`
Server-Typ	`tinytext`	`Apache/1.3.14 (Unix) FrontPage/3.0.4.2 PHP/4.0.4 mod_ssl/2.7.1 OpenSSL/0.9.6`
HTTP-Version	`char(5)`	`1.1`
Stadt	`char(3)`	`GI`

Tabelle 9.1.: Die Tabellenstrukturen `http_header` und `server_info` der Datenbank zur Aufnahme der Metadaten mit assoziierten Datentypen sowie vollständige Beispieldaten für die Adresse `http://www.uni-giessen.de/~g91063/ps/textannotation.ps.gz`

abgelegt wurde (vgl. Tab. 9.1). Zum Parsing der lokal abgelegten HTTP-Header, zur Erzeugung der in die Datenbank zu importierenden Dateien sowie für rudimentäre statistische Auswertungen (siehe Abschnitt 9.5) wird die dritte zentrale Komponente – das Perl-Skript `build-db-import.pl` – eingesetzt.[12]

Das Speichern der Dokumente im UNIX-Dateisystem – im Gegensatz zur Datenbank – bietet verschiedene Vorteile: Zum einen wird der Einsatz von Werkzeugen zur Analyse der Daten (z. B. sed-, awk- oder Perl-Skripte sowie SGML-Parser zur Validierung der HTML-Dokumente) ohne den Performanz-intensiven Umweg über die Datenbank ermöglicht. Zum anderen bleiben die Daten durch den in der Datenbank enthaltenen Verweis vollständig zugreifbar. Dies ist besonders wichtig für das PHP-basierte[13] Front-End, das für einen möglichst intuitiven Zugriff der im Korpus enthaltenen Daten entwickelt wird.

Aufgrund der datenbankbasierten Erfassung der Metadaten wird eine hohe Flexibilität garantiert: Entsprechende Retrieval- und Transformationsskripte vorausgesetzt, ist es möglich, automatisch TEI-Header (Sperberg-McQueen und Burnard, 1994) sowohl für einzelne Dokumente als auch für das gesamte Korpus (Dunlop, 1995, Walker, 1999) oder RDF-basierte Beschreibungen nach dem Dublin Core Standard (Weibel et al., 1999) zu generieren.

9.3. Automatische Erkennung der Sprache

Da das Korpus ausschließlich deutschsprachige Dokumente enthalten soll, ist ein automatisches Verfahren zur Erkennung der Sprache erforderlich. Im Bereich der *Language Identification* (Muthusamy und Spitz, 1998) dominieren einerseits statistische Methoden, andererseits Wortlisten-basierte Ansätze (siehe Langer, 2001). Aufgrund der Beschränkung hinsichtlich der deutschen Sprache, ist im vorliegenden System keine Differenzierung bezüglich der Erkennung möglichst vieler unterschiedlicher Sprachen notwendig. Stattdessen ist eine Klassifizierung ausreichend, ob es sich bei der Sprache um *deutsch* oder *unbekannt* handelt.

Verschiedene Experimente mit im Quelltext verfügbaren Werkzeugen[14] haben gezeigt, dass diese Systeme entweder unbefriedigende Kategorisierungsergebnisse liefern oder schlicht zu langsam ablaufen. Gerade die Performanz spielt jedoch bei der Sammlung der Daten eine wesentliche Rolle, da für *jedes* besuchte HTML-Dokument eine Überprüfung der Sprache notwendig ist.

Prinzipiell können – neben der Analyse des in einem HTML-Dokument enthaltenen Textes – weitere Informationen zur Identifikation der Sprache eingesetzt werden. Hierzu gehören etwa die in der jeweiligen Adresse enthaltene Top-Level Domäne (`.de`, `.fr`, `.es` etc.) oder eine explizite Angabe der Sprache innerhalb eines `<meta>` Tags. In der Realität sind derartige Informationen kaum nutzbar: Die Top-Level Domäne kann lediglich einen sehr groben Hinweis auf den Standort des Servers geben; explizite Sprachangaben in Form von `<meta>` Tags waren bei vorab

[12] Eine Verarbeitung der in Fielding et al. (1999) spezifizierten HTTP-Response Header ist prinzipiell mit einfachen Mitteln zu realisieren, jedoch senden viele Server – etwa der Microsoft Internet Information Server (Versionen 3.0 und 4.0) – vereinzelte Header-Zeilen, die nicht dem HTTP-Standard entsprechen. Weitere Server – z. B. der CERN Server in der Version 3.0 – scheinen nicht Jahr-2000 kompatibel zu sein und senden fehlerhafte Datumsangaben. Diese nicht RFC 2616 konformen Zeilen werden mithilfe von Ausnahmeregeln verarbeitet.

[13] Die serverseitig ausgeführte Skriptsprache PHP (PHP: Hypertext Preprocessor, `http://www.php.net`) verfügt über eine Schnittstelle zur Datenbankkommunikation und wird eingesetzt, um HTML-Inhalte dynamisch generieren zu können.

[14] Beispielsweise das n-Gram basierte TextCat, implementiert von Gertjan van Nord, `http://odur.let.rug.nl/~vannoord/TextCat/`

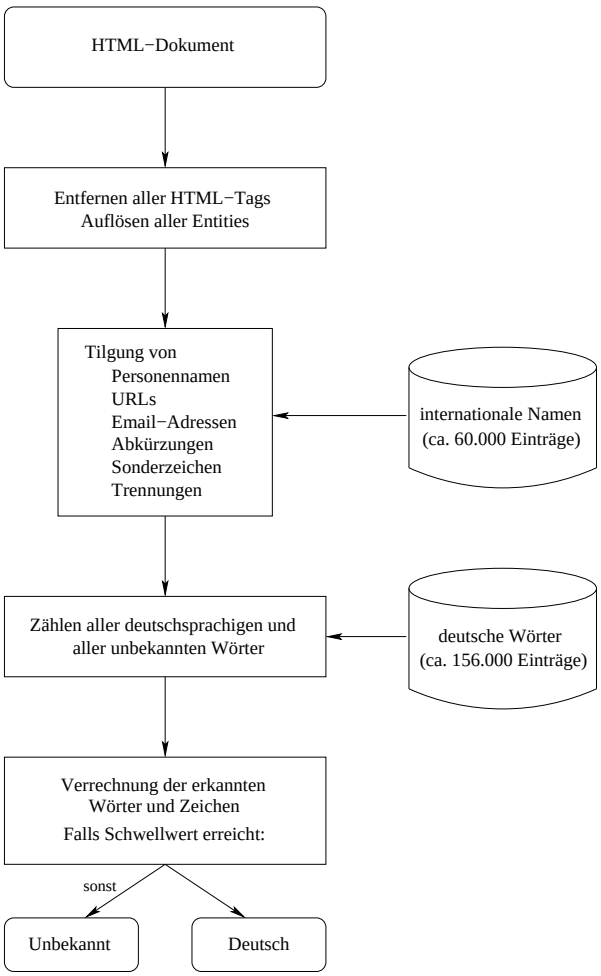

Abbildung 9.2.: Die Erkennung der Sprache eines HTML-Dokuments

durchgeführten Tests kaum verfügbar (vgl. hierzu auch O'Neill et al., 1998), weshalb der hier beschriebene Ansatz zur Identifizierung der Sprache ausschließlich auf der Analyse des in einem Dokument enthaltenen Textes basiert.

Vor der Verarbeitung des Dokuments durch den Erkenner werden zunächst mithilfe des in einem Shell-Skript eingesetzten Webbrowsers Lynx[15] alle HTML-Elemente entfernt und Entitäten aufgelöst (z. B. `&ouml;` → ö in der Latin 1 Kodierung). Daraufhin werden durch den in Perl implementierten Erkenner `gercount.pl` mithilfe regulärer Ausdrücke diejenigen Token des Eingabestroms entfernt, die vollkommen unabhängig von der benutzten Sprache sind. Hierbei handelt es sich beispielsweise um URLs, Email-Adressen, verschiedene Sonderzeichen und Eigennamen.[16] In der folgenden Verarbeitungsstufe werden für alle deutschsprachigen und alle unbekannten Token Frequenzlisten berechnet. Die Wortliste, die für diesen Zweck eingesetzt wird,

[15] Erhältlich unter `http://lynx.isc.org`.

[16] Aus einem Projekt zur Eigennamenerkennung standen Namenslisten zur Verfügung, die für diese Aufgabe eingesetzt werden konnten, vgl. Kober et al. (1998).

umfasst etwa 156 000 Einträge und wurde halbautomatisch auf ihre Korrektheit überprüft: Einerseits wurde die Liste mithilfe anderssprachiger Wortlisten, die in vielfältigen Ausprägungen im Internet erhältlich sind, abgeglichen, d. h. Einträge, die sowohl in der anderssprachigen Liste und der deutschsprachigen Liste enthalten waren, wurden entfernt; andererseits wurden große Teile der Liste manuell überprüft. Sobald die Anzahl der deutschsprachigen und die Anzahl der unbekannten Wörter vorliegen, werden die Häufigkeiten sowohl bzgl. der Anzahl der *Wörter* als auch *Zeichen*[17] eines Wortes in Beziehung zur Länge des bereinigten Dokuments (s. o.) gesetzt und der prozentuale Anteil erkannter Wörter berechnet. Dieser beträgt für sehr viele bislang verarbeitete Dokumente zwischen 85 und 100%. Die Festlegung des Schwellwertes, der die Grenze zwischen *deutsch* und *unbekannt* bestimmt, gestaltete sich schwierig, da verschiedene Grenzfälle – etwa sehr viele fachsprachliche Termini, englischsprachige Navigationsleisten etc. – den prozentualen Anteil erkannter Wörter teilweise erheblich reduziert haben. Aus diesem Grund arbeitet das System derzeit mit einem Schwellwert von 40%. Dieser ist klein genug, um die angesprochenen Grenzfälle korrekt zu klassifizieren, und er ist groß genug, um Dokumente unbekannter Sprache nicht in das Korpus aufzunehmen. Abb. 9.2 veranschaulicht diesen Prozess der Spracherkennung.

Zur Evaluation des Ansatzes wurden aus einer Liste aller Webdokumente, die innerhalb der Domäne `*.uni-giessen.de` angeboten werden, 150 Adressen zufällig ausgewählt: Von den 150 Dokumenten wurden 144 korrekt klassifiziert, 97 als *deutsch* und 47 als *unbekannt*. Ein Dokument konnte nicht klassifiziert werden, da es nicht vom Typ `text/html` war[18] (Recall: 99,3%, Precision: 96,64%). Fünf Dokumente wurden falsch klassifiziert: Bei vier Dokumenten lag Bilingualität vor, d. h. die Dokumente enthielten sowohl deutschsprachige als auch ähnlich viele anderssprachige Abschnitte. Derartige Eingaben kann der Erkenner – bedingt durch den Ansatz – nicht korrekt klassifizieren. Das verbleibende Dokument besteht lediglich aus fünf Token, die in dem Dokument die Beschreibung eines Fotos darstellen und die – bis auf zwei Artikel – allesamt als sehr fachsprachlich bezeichnet werden können, sodass auch in diesem Fall die Erkennung scheitern muss, da die fachsprachlichen Wörter nicht in der Wortliste des Systems enthalten sind. Nicht enthaltene deutschsprachige Wörter werden, je länger eine Eingabe ist, durch die jeweils erkannten Wörter kompensiert, d. h. der hier beschriebene Ansatz ist für extrem kurze – weniger als etwa zehn Wörter umfassende – fachsprachliche Dokumente nicht geeignet.

9.4. Datenauswertung und Datenannotation

Sobald das Korpus in seiner vollständigen Form vorliegt, können zahlreiche Merkmale der Dokumente untersucht und in eigenständigen Tabellen, die mit den jeweiligen Dokumenten verknüpft sind, in der Datenbank abgelegt werden. Derartige Merkmale sind etwa die Anzahl der HTML-Elemente, die Anzahl der Wörter pro Dokument, die Anzahl und die Länge der Sätze, die durchschnittliche Länge der Dokumente eines Servers etc. (vgl. Abschnitt 9.6).

Auch eine Up-Konvertierung der HTML-Dokumente in ein SGML-basiertes Format ist prinzipiell möglich, jedoch mit einem erheblichen Aufwand verbunden, wie Walker (1999) anhand des Zielformats TEI Lite zeigt. An dieser Stelle können einige der Probleme nur angerissen werden: Es existieren strukturelle, typographische, referentielle und funktionale HTML-Elemente, die eine eindeutige Abbildung auf TEI-Elemente erschweren: „Despite its apparent simplicity,

[17] Damit ein Treffer wie „Abgeordnetenentschädigungsgesetzes“ eine bessere Bewertung bekommt als etwa „absolut“.

[18] Derartige Dokumente werden bei der Korpuserstellung nicht an das Werkzeug zur Identifizierung der Sprache weitergereicht, der Recall beträgt also im Rahmen der Produktionsumgebung zwangsläufig immer 100%.

Web document parsing and re-encoding is not trivial.“ (Walker, 1999, S. 189). Im SGML-Sinne nicht wohlgeformte HTML-Dokumente müssen vor der Konvertierung in eine wohlgeformte Struktur transformiert werden, was nicht immer fehlerfrei vonstatten geht.[19] Auch die hypertextuelle Struktur einer *Gruppe* von Dokumenten ist problematisch: Wo beginnt ein Dokument, wo endet es?

9.5. Statistische Auswertung

Derzeit ist das Korpus zu etwa 20% gefüllt; die obere Grenze wird lediglich durch den verfügbaren Festplattenplatz markiert, der etwa 30 Gigabyte beträgt. Tab. 9.2 (S. 101) bietet eine Übersicht über den aktuellen Umfang des Korpus sowie über die Gesamtgröße der bislang besuchten Webserver. Des Weiteren werden die vier häufigsten MIME-Typen mit der jeweiligen Anzahl der gefundenen Dateien dieses Typs notiert (vgl. O'Neill, 1997, Pitkow, 1998, Turau, 1998).

9.6. Weitergehende Analysen

Neben dem eingangs angesprochenen Forschungsvorhaben sind weitergehende Analysen der Daten denkbar. Die Arbeit an den im folgenden skizzierten Teilprojekten wird nach Fertigstellung des Korpus aufgenommen.

Wir werden Analysen durchführen, die sich an der Forschung der *computervermittelten Kommunikation* orientieren und sprachliche Phänomene in Webdokumenten untersuchen, wobei wir auch den durchschnittlichen „Grad der konzeptionellen Mündlichkeit vs. Schriftlichkeit“ (Haase et al., 1997) für unterschiedliche Dokumentgruppen ermitteln werden (Rehm, 2001). Ähnlich gelagerte Forschungsvorhaben können das Korpus über das WWW benutzen, was ein weiterer Grund für die Entwicklung der PHP-basierten Navigationsoberfläche ist. Somit können Interessierte auf ein zentrales Korpus zugreifen und dabei Kommunikationsforen auf dem Korpus-Server nutzen (vgl. Rehm und Lobin, 2000).

Es werden statistische Analysen über die *Häufungen verschiedener Dateiformate* in der Domäne der akademischen Webserver durchgeführt, etwa die Anzahl der Java-Applets, Anzahl der angebotenen PDF- oder Postscript-Dokumente, Einsatz proprietärer HTML-Elemente etc. (Pitkow, 1998, Turau, 1998).

Analysen haben ergeben, dass Webdokumente häufig modifiziert werden. Die Erstellung eines Monitor-Korpus (Kennedy, 1998, S. 61), dem regelmäßig neue Versionen bereits im Korpus enthaltener Dokumente hinzugefügt werden, erlaubt mittels automatischer Dokumentvergleiche Rückschlüsse über die im akademischen Bereich durchgeführten Revisionen sowie bei ausreichend häufig durchgeführten Erhebungen evtl. empirische Erkenntnisse bzgl. der Frage, ob digitale Kommunikationsmedien einen Sprachwandel hervorrufen (Haase et al., 1997).

Ein weiteres Ziel der Arbeit ist das Einreichen des Korpus beim *Internet Archive* (`http://www.archive.org`), das sich zur Aufgabe gemacht hat, das gesamte Internet (incl. FTP-Server, World Wide Web und Usenet) in regelmäßigen Abständen zu traversieren und zu archivieren.

[19] Das von Dave Raggett entwickelte Werkzeug tidy (`http://www.w3.org/People/Raggett/tidy/`) ist allerdings in der Lage, viele HTML-Fehler automatisch korrigieren zu können.

Stadt	Anzahl Server	Dokumente gesamt	Dokumente im Korpus	Die vier häufigsten MIME-Typen	
Ulm	167	237 272 (6 253 MB)	53 880 (311 MB)	text/html	113 751
				image/gif	70 420
				image/jpeg	37 954
http://www.uni-ulm.de				text/plain	4 701
Leipzig	160	349 423 (9 841 MB)	115 776 (702 MB)	text/html	203 485
				image/gif	67 710
				image/jpeg	42 935
http://www.uni-leipzig.de				text/xml	14 336
Cottbus	126	139 692 (4 329 MB)	22 029 (172 MB)	text/html	69 196
				image/gif	35 323
				image/jpeg	24 716
http://www.tu-cottbus.de				application/pdf	2 237
Gießen	119	314 621 (5 789 MB)	97 124 (635 MB)	text/html	160 965
				image/gif	50 438
				image/jpeg	30 298
http://www.uni-giessen.de				application/pdf	2 805
Trier	119	83 919 (3 142 MB)	24 765 (158 MB)	text/html	49 476
				image/gif	15 981
				image/jpeg	11 269
http://www.uni-trier.de				application/postscript	2 048
Eichstätt	15	71 424 (952 MB)	14 678 (86 MB)	text/html	43 210
				image/gif	22 188
				image/jpeg	3 580
http://www.ku-eichstaett.de				text/plain	695
Witten-Herdecke	13	11 993 (362 MB)	4 899 (56 MB)	text/html	6 683
				image/gif	3 398
				image/jpeg	1 177
http://www.uni-wh.de				application/pdf	488

Tabelle 9.2.: Der Umfang des Korpus sowie Angaben über die Häufungen von Dateitypen in den jeweiligen Universitätsnetzen (Stand: Anfang Februar 2001)

Literaturverzeichnis

Berners-Lee, Tim; Cailliau, Robert; Groff, Jean-François und Pollermann, Bernd (1992): "World-Wide Web: The Information Universe". *Electronic Networking: Research, Applications and Policy* 1 (2).

Cowie, Jim; Ludovik, Evgeny und Zacharski, Ron (1998): "An Autonomous, Web-based, Multilingual Corpus Collection Tool". In: *Proceedings of the International Conference on Natural Language Processing and Industrial Applications*. Moncton, S. 142–148. Online verfügbar: `http://crl.nmsu.edu/~raz/langrec/nlpia.htm`.

DuBois, Paul (1999): *MySQL*. Indianapolis: New Riders.

Dunlop, Dominic (1995): "Practical Considerations in the Use of TEI Headers in a Large Corpus". *Computers and the Humanities* (29): S. 85–98.

Fielding, R.; Gettys, J.; Mogul, J. C.; Frystyk, H.; Masinter, L.; Leach, P. und Berners-Lee, T. (1999): "Hypertext Transfer Protocol – HTTP/1.1". Network Working Group – Request for Comments (RFC) 2616. Online verfügbar: `http://www.rfc-editor.org`.

Freed, N. und Borenstein, N. (1996): "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies". Network Working Group – Request for Comments (RFC) 2045. Online verfügbar: `http://www.rfc-editor.org`.

Haase, Martin; Huber, Michael; Krumeich, Alexander und Rehm, Georg (1997): "Internetkommunikation und Sprachwandel". In: *Sprachwandel durch Computer*, herausgegeben von Weingarten, Rüdiger, Opladen: Westdeutscher Verlag, S. 51–85.

Hawking, David; Craswell, Nick und Harman, Donna (1999): "Results and Challenges in Web Search Evaluation". In: *The Eigth International World Wide Web Conference*, herausgegeben von Tang, E. International World Wide Web Conference Committee, Foretec Seminars, National Research Council Canada, Toronto. Online verfügbar: `http://www8.org/w8-papers/2c-search-discover/results/results.html`.

Kennedy, Graeme (1998): *An Introduction to Corpus Linguistics*. Studies in Language and Linguistics. London, New York: Longman.

Kober, Katharina; Krumeich, Alexander; von der Landwehr, Klaus; Langer, Hagen und Rehm, Georg (1998): "Projektbericht pronto: Probleme der Eigennamenerkennung". Institut für Semantische Informationsverarbeitung, Universität Osnabrück. Unveröffentlichtes Manuskript. Siehe `http://www.cl-ki.uni-osnabrueck.de/pronto/`.

Langer, Stefan (2001): "Sprachen auf dem WWW". In: *Proceedings der Frühjahrstagung der Gesellschaft für Linguistische Datenverarbeitung*, herausgegeben von Lobin, Henning. Justus-Liebig-Universität Gießen. In diesem Band.

Müller, Martin (1999): "Inducing Conceptual User Models". In: *Proceedings of ABIS-99 (Adaptivität und Benutzermodellierung in interaktiven Softwaresystemen)*, herausgegeben von Wrobel, Stefan. Gesellschaft für Informatik, Magdeburg. Online verfügbar: `http://www-mmt.inf.tu-dresden.de/joerding/abis99/proceedings.html`.

Muthusamy, Yeshwant K. und Spitz, Lawrence (1998): "Automatic Language Identification". In: *Survey of the State of the Art in Human Language Technology*, herausgegeben von Cole, Ronald; Mariani, Joseph; Uszkoreit, Hans; Varile, Giovanni Battista; Zaenen, Annie und Zampolli, Antonio, Cambridge: Cambridge University Press, S. 314–317. Online verfügbar: `http://cslu.cse.ogi.edu/HLTsurvey/HLTsurvey.html`.

O'Neill, Edward T. (1997): "Characteristics of Web Accessible Information". In: *63rd IFLA General Conference*. International Federation of Library Associations and Institutions, Kopenhagen. Online verfügbar: `http://www.ifla.org/IV/ifla63/63onee.htm`.

O'Neill, Edward T.; Lavoie, Brian F. und McClain, Patrick D. (1998): "Web Characterization Project – An Analysis of Metadata Usage on the Web". In: *Annual Review of OCLC Research 1998*, Dublin: OCLC Online Computer Library Center. Online verfügbar: `http://www.oclc.org`.

Pitkow, James E. (1998): "Summary of WWW Characterizations". In: *Seventh International World Wide Web Conference*. WWW7 Consortium, Brisbane. Online verfügbar: `http://www7.scu.edu.au/programme/fullpapers/1877/com1877.htm`.

Raggett, Dave; Hors, Arnaud Le und Jacbos, Ian (1999): "HTML 4.01 Specification". Technische Spezifikation, World Wide Web Consortium. Online verfügbar: `http://www.w3.org/TR/html401/`.

Rehm, Georg (2001): "Medienkonvergenz der etwas anderen Art – Eine empirische Analyse über den Einfluss ursprünglich E-Mail basierter Pseudo-Korrelate auf den Stil persönlicher Homepages". In: *Kommunikationsform E-Mail*, herausgegeben von Ziegler, Arne. Im Erscheinen.

Rehm, Georg und Lobin, Henning (2000): "From Open Source to Open Information – Collaborative Methods in Creating XML-based Markup Languages". In: *Proceedings of Electronic Publishing 2000*. International Federation for Information Processing and International Council for Computer Communication, Kaliningrad, Svetlogorsk. Online verfügbar: `http://www.uni-giessen.de/~g91063/pdf/open-information.pdf`.

Runkehl, Jens; Schlobinski, Peter und Siever, Torsten (1998): *Sprache und Kommunikation im Internet – Überblick und Analysen*. Opladen, Wiesbaden: Westdeutscher Verlag.

Sperberg-McQueen, C. M. und Burnard, Lou (Herausgeber) (1994): *Guidelines for Electronic Text Encoding and Interchange*. Chicago, Oxford: University of Chicago, University of Oxford. Version P3, 2 Bände.

Storrer, Angelika (1999): "Was ist eigentlich eine Homepage? Neue Formen der Wissensorganisation im World Wide Web". *Sprachreport* (1): S. 2–8. Online verfügbar: `http://www.ids-mannheim.de/grammis/storrer.html`.

Turau, Volker (1998): "Eine empirische Analyse von HTML-Dokumenten im WWW". Technischer Bericht 0198, Fachhochschule Wiesbaden, Wiesbaden. Online verfügbar: `http://www.informatik.fh-wiesbaden.de/~turau/`.

Walker, Derek (1999): "Taking Snapshots of the Web with a TEI Camera". *Computers and the Humanities* (33): S. 185–192.

Wätjen, Hans-Joachim; Diekmann, Bernd; Möller, Gerhard und Carstensen, Kai-Uwe (1998): "Bericht zum DFG-Projekt: GERHARD – German Harvest Automated Retrieval and Directory". Technischer Bericht, Bibliotheks- und Informationszentrum (BIS) der Carl von Ossietzky Universität Oldenburg. Online verfügbar: `http://www.gerhard.de/info/index_de.html`.

Weibel, S.; Kunze, J.; Lagoze, C. und Wolf, M. (1999): "Dublin Core Metadata for Resource Discovery". Network Working Group – Request for Comments (RFC) 2413. Online verfügbar: `http://www.rfc-editor.org`.