Hauptseminar: Intelligente Systeme
Sprachliche Informationsverarbeitung
Institut für Linguistik, Universität zu Köln
Wintersemester 2006/2007

Exemplarbasierte Textklassifikation

Dozent: Prof. Dr. Jürgen Rolshoven
Referenten: Valentina Rahmanian, Fabian Steeg, Sonja Subicin

Inhalt

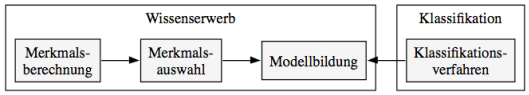
- Textklassifikation, Bezüge NLP, Korpuslinguistik
- Induktives, exemplarbasiertes maschinelles Lernen
- Paradigmen und Suffixbäume zur Modellbildung
- Evaluationsmaße, Ergebnisse, Verbesserungen
- Software Architecture for Language Engineering

Textklassifikation: Definition und Abgrenzung

- Definition Textklassifikation: Automatische Zuordnung von Texten zu vordefinierten Kategorien
- Ähnlich: Keyword-Extraktion, hier werden Wörter im Text auf ihre Relevanz für den Inhalt des Textes untersucht, erfordert kein Training, keine Beispiele
- Unterschied zur Textklassifikation: Die Klassen, denen Texte zugeordnet werden, müssen nicht als Wort im Text vorhanden sein (ein Artikel über Politik muss nicht das Wort *Politik* enthalten), erfordert Training, z.B. aus Beispielen: Induktives, exemplarbasiertes Lernen
- Mehr zu Texklassifikationverfahren z.B. in Goller et al. 2000

Aufbau von Textklassifikationssystemen

- Zwei Hauptkomponenten (Brückner 2001): **Wissenserwerb** (durch maschinelles Lernen oder manuell) und die eigentliche **Klassifikation**
- Der Wissenserwerb besteht aus **Merkmalsberechnung** (Ermittlung der Merkmale, anhand derer klassifiziert wird), **Merkmalsauswahl** (welche davon sind wirklich relevant) und **Modellbildung** (die Erstellung des fertigen Modells, anhand dessen klassifiziert wird, dieses wird vom Klassifikationsverfahren verwendet)

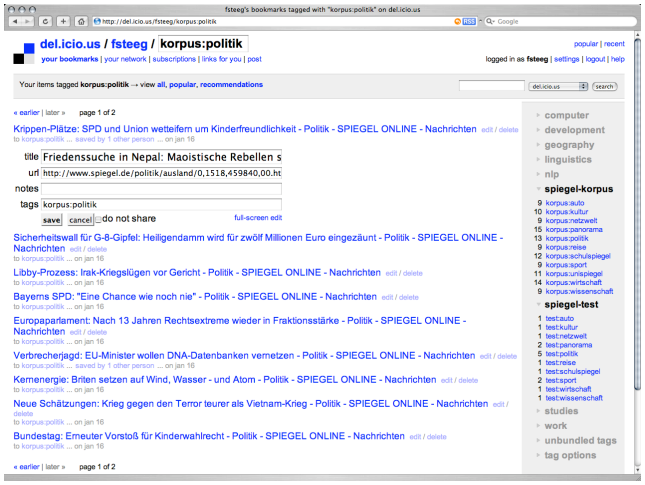


NLP und Korpuslinguistik

- Korpuslinguistik: Erstellung und Auswertung von Textkorpora (Korpora: "the raw fuel of NLP", McEnery 2003)
- Korpora nicht irgendwelche Texte, sondern auf eine bestimmte Fragestellung hin organisiert (*sampling frame*), und in Bezug auf diese ausgewogen und repräsentativ
- Wenn etwa Zeitungsartikel klassifiziert werden sollen, sollte das Korpus aus Zeitungsnachrichten bestehen, nicht aus Romanen oder Alltagskonversation; zudem aus unterschiedlichen Rubriken (dadurch ausgewogen, repräsentativ)
- Es gibt öffentliche, allgemeine Korpora ohne spez. sampling frame. Darüber hinaus steht mit dem Web eine Grundlage zur Zusammenstellung von domänenspezifischen Korpora bereit

Korpora und Delicious

- Delicious (http://del.icio.us/) ist ein Web-2.0 Tool zum *social bookmarking,* eines des frühesten Beispiele für *Tagging* (im Web-2.0-Kontext). Ermöglicht Klassifikation (*Tagging*) von Bookmarks und Bündelung von Klassen (in sog. *Bundles*)
- Solche Bündel könnten bei entsprechenden Inhalten den sampling frames entsprechen, so kann Delicious zum Aufbau von domänenspezifischen Korpora verwendet werden
- Del.icio.us stellt eine Web-basierte API bereit, für die es eine open-source Java-Bibliothek gibt: http://sourceforge.net/projects/delicious-java/
- Der eigentliche Inhalt der verlinkten Website kann mit korrigierenden HTML-Parsern ausgelesen werden (z.B. http://people.apache.org/~andyc/neko/doc/)

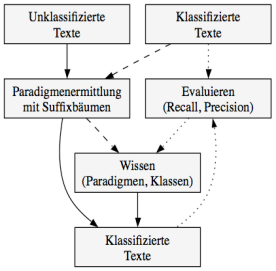


Unser Klassifikationsverfahren

- Grundgedanke: Texte haben das gleiche Thema, wenn sie die gleichen Wörter in den gleichen Kontexten enthalten (d.h. wenn sie ähnliche Paradigmen enthalten)
- z.B. wenn in einem Text *Chips* und *Eis* im gleichen Kontext auftauchen, geht es vermutlich um Essen. In einem Text über Mikrochips kommt das Wort *Chips* auch vor, allerdings nicht im gleichen Kontext wie *Eis*, daher sollte der Text nicht der gleichen Klasse zugeordnet werden, z.B.:
- *Hans mag Eis. Anna mag Chips nicht. Ich mag Chips gern.*
- *In Dresden produziert Infineon Chips. Die Produktion liegt zur Zeit auf Eis.*
- Die Paradigmen stellen das Wissen des Systems dar (eine symbolische Wissensrepräsentation)

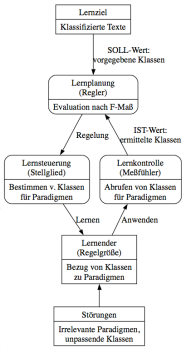
Übersicht

- Als Eingabe dienen schon klassifizierte Texte, daraus wird gelernt (gestrichelte Linie)
- Dann können unklassifizierte Texte klassifiziert werden (durchgezogene Linie)
- Zur Evaluation werden klassifizierte Texte neu klassifiziert und das Ergebnis mit der ursprünglichen Klassifikation verglichen (gepunktete Linie)



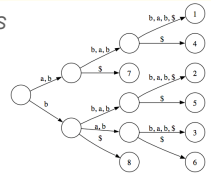
Wissenserwerb, Maschinelles Lernen

- Merkmalsberechnung: Erstellung von Paradigmen durch Suffixbäume
- Merkmalsauswahl: Die besten Paradigmen (Filtern durch Stopwort-Listen)
- Modellbildung: Bezug von Klassen zu Paradigmen
- Wissenserwerb ist eine Form von masch. Lernen
- Verbreitete Verfahren zum maschinellen Lernen verwenden numerische Repräsentationen der Merkmale, hier etwa: ob und wie sehr ein Paradigma für eine Klasse relevant ist (ein Merkmalsvektor pro Klasse)
- Induktives, exemplarbasiertes Lernverfahren: Lernen aus Beispielen, siehe Wörterbuch der Kognitionswissenschaft, Felix v. Cube, Kybernetische Grundlagen des Lernens (Abb.)



Suffixbaum: Definition

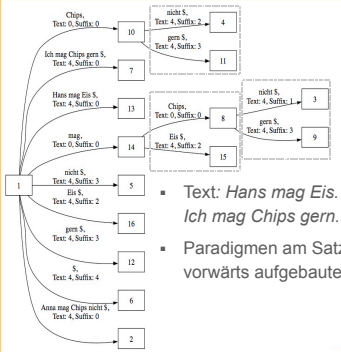
- Ein Suffixbaum *T* für einen String *S* mit *m* Symbolen ist ein gerichteter Baum mit *m* Blättern
- Jede Kante ist mit einem Teilstring von *S* beschriftet
- Jeder innere Knoten von *T* hat mindestens zwei Kinder, deren Kantenbeschriftungen nie mit dem gleichen Symbol beginnen. Für jedes Blatt *i* in *T* ergeben die Beschriftungen der Kanten auf dem Pfad von der Wurzel zu *i* aneinander gehängt das Suffix von *S*, das an Index *i* beginnt. Somit enthält *T* alle Suffixe von *S*, wobei mehrfach auftretende Teilstrings nur einmal in *T* enthalten sind (Quelle: Wikipedia, zu Details siehe Gusfield 1997)



Paradigmen in Suffixbäumen

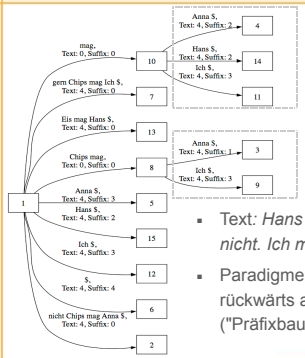
- Für die **syntaktische** Analyse mit Suffixbäumen: als Symbole nicht einzelne Buchstaben, sondern **Wörter**
- Zum Vergleich von Sätzen kann dann ein wortbasierter, über mehrere Sätze generalisierter Suffixbaum erstellt werden, d.h. der Baum enthält mehrere Sätze, wobei Satzgrenzen berücksichtigt werden
- Strukturalistische Ideen: syntagmatische (gemeinsames Auftreten) und paradigmatische Relationen (austauschbar im Kontext)
- Suffixbäume enthalten in ihrer Struktur **Paradigmen:** Beschriftungen der Kanten von inneren Knoten zu ihren Kindern stehen zueinander in paradigmatischer Beziehung
- Suffixbäume lassen sich effizient erstellen, daher können mit Suffixbäumen effizient Paradigmen aus großen Korpora erstellt werden

Suffixbaum I



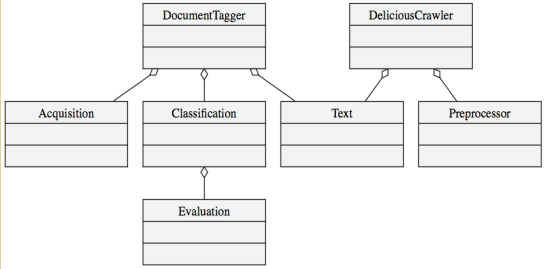
- Text: *Hans mag Eis. Anna mag Chips nicht. Ich mag Chips gern.*
- Paradigmen am Satzende werden im vorwärts aufgebauten Baum sichtbar

Suffixbaum II: Präfixbaum



- Text: *Hans mag Eis. Anna mag Chips nicht. Ich mag Chips gern.*
- Paradigmen am Satzanfang werden im rückwärts aufgebauten Baum ("Präfixbaum") sichtbar

Implementation: Klassen



- Die Programmstruktur spiegelt den allgemeinen Aufbau von Textklassifikationssystemen wie zuvor beschrieben dar, erweitert um Klassen zum Crawling und zur Vorverarbeitung (HTML, Filter)

Implementation: Vorgehen

- **Wissenserwerb**
  1. Merkmalsberechnung: Paradigmen ermitteln im Suffixbaum
  2. Merkmalsauswahl: Stopwort-Listen-Filter: Paradigmen filtern
  3. Modellbildung: Paradigmen und für diese relevante Klassen (z.B. [chips, eis] --> [essen], [chips, monitore] --> [computer])
- **Klassifikation**
  1. Paradigmen im zu klassifizierenden Text ermitteln
  2. Berechne für jede mögliche Klasse die beste Übereinstimmung eines der für die Klasse relevanten Paradigmen mit einem Paradigma im zu klassifizierenden Text (z.B. [chips, eis] im Modell, [chips, cola] im Text, Übereinstimmung: 50%)
  3. Wähle alle Klassen mit einer Übereinstimmung über einem bestimmten Schwellenwert als relevante Klasse für den zu klassifizierenden Text (für Versuch 50%)

Evaluation: Maße

- Precision: Wie viele der ermittelten Klassen sind korrekt
- Recall: Wie viele der zu ermittelnden Klassen wurden auch tatsächlich ermittelt
- F-Maß: Einheitliche Betrachtung von Precision und Recall; sog. *gewichtetes harmonisches Mittel*, trad: (2*R*P) / (R+P)
- Evaluation der Qualität eines klassifizierten Dokuments (für mehrere Kategorien): Vergleich von ermittelten Klassen und den menschlich vergebenen (wie viele der zu ermittelten wurden auch ermittelt; wie viele der ermittelten sind korrekt)
- Die Qualität das Verfahrens: Mittelwerte für eine Menge klassifizierter Dokumente

Evaluation: Ergebnisse

Text	Recall	Precision	F-Maß
1	0	0	0
2	1	0,25	0,4
3	0	0	0
4	1	0,25	0,4
5	1	0,5	0,67
6	1	0,5	0,67
7	1	1	1
8	1	0,5	0,67
9	1	0,5	0,67
10	1	0,5	0,67
11	1	0,35	0,5
12	1	0,5	0,67
13	1	0,5	0,67
14	1	0,25	0,4
15	1	0,25	0,4
16	1	1	1
Summe	14	6,85	8,78
Mittel	0,88	0,43	0,55

- Test mit kleinem Spiegel-Online-Korpus aus 120 Artikeln, Test mit 16 Artikeln, 10.000 Merkmalen (Paradigmen)
- Ergebnisse bei Recall schon überraschend gut, durch geringe Precision insgesamt verbesserungswürdig
- Ergebnisse bei allgemeineren Korpora mit vielen Kategorien schlechter (vgl. sampling frame)

Mögliche Verbesserungen

- Qualität der Paradigmen (Filtern, Mehrwort-Paradigmen, nur mit bestimmten Wörtern, Zusammenfassen)
- Qualität und Quantität der Korpora, HTML-Parsing: z.B. nicht nur Inhalt von Paragraph-Elementen
- Ändern des Algorithmus: Schwellenwert anders; automatisch anpassen; N beste (z.B. 5%) Paradigmen statt festen Wert
- Sprachspezifisch: Stemming (alle Formen eines Wortes bei der Bildung der Paradigmen berücksichtigen), POS-Tagging (POS berücksichtigen: *fliegen* als Verb oder Nomen nicht im gleichen Paradigma), Semantische Relationen (z. B. auch Hyperonyme der Wörter im Paradigma berücksichtigen), all dies z.B. mit WordNet
- Fast alles davon (Paradigmen, Vorverarbeitung, Stemming, Tagging) erfordert die Integration mit anderen Komponenten: SALE

Software Architecture for Language Engineering (SALE)

- Language Engineering: *"Production of software systems that involve processing human language with quantifiable accuracy and predictable development resources."* (Cunningham 1999), *dt.: "Die Herstellung von Software-Systemen, welche die Verarbeitung natürlicher Sprache mit quantitativ bestimmbarer Präzision und verlässlichen Entwicklungsressourcen beinhalten."*
- LE ist verwandt mit den Bereichen CL, NLP und AI, hat aber einen anderen Fokus und setzt anderen Prioritäten
- SALE: Konstruktion einer softwaretechnischen **Infrastruktur** für die Verarbeitung von Sprache ("Werkzeugkasten" zur Konstruktion von Systemen und Experimenten)

Drei Arten von Infrastruktur

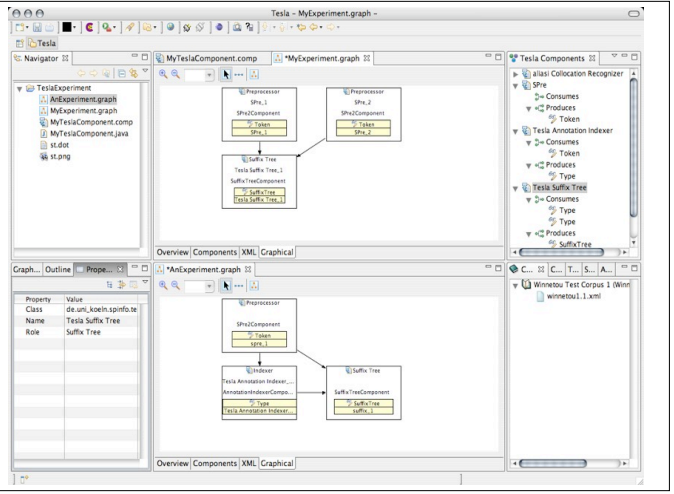
- **Framework** (Platform, Component System): Objektorientierte Klassenbibliothek, die für einen bestimmten Bereich konstruiert wurde und für die Lösung von Problemen in diesem Bereich erweitert oder zugeschnitten werden kann
- Alle Software-Systeme haben eine explizite oder implizite **Architektur**. Explizit (Reference Architecture, Domain-Specific-Architecture): Evtl. allg. Standards entsprechend, auf mehrere Systeme gerichtet, implizit (*software architecture for a family of application systems*)
- Eine Implementation einer Architektur, die z. B. graphische Werkzeuge für die Erstellung und das Testen von Systemen bereitstellt, ist eine **Entwicklungsumgebung**

SALE: Eigenschaften

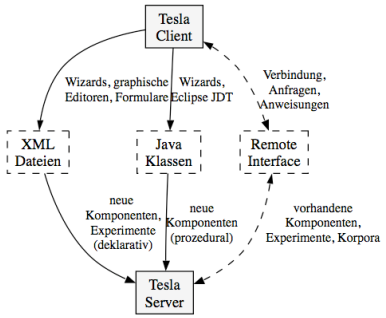
- Strikte Trennung systemnaher Aufgaben von den Datenstrukturen und Algrorithmen zur Sprachverarbeitung
- Bereitstellung von standardisierten Mechanismen für die Sprachdaten-Übertragung zwischen Komponenten und Verwendung offener Standards (Java, XML) als zugrundeliegende Plattform zur Reduzierung von Integrationskosten
- Bereitstellung eines erweiterbaren Basis-Sets von Komponenten zur Sprachverarbeitung, dessen Komponenten ggf. vom Benutzer (z. B. zu Vergleichszwecken) ausgetauscht werden können
- Bereitstellung einer Entwicklungsumgebung oder eines Sets von Werkzeugen zur Erleichterung der Modifikation und Implementation von sprachverarbeitenden Komponenten und Anwendungen.
- Wiederverwendbarkeit, Messung der Qualität von Ergebnissen

SALE: Implementierungen

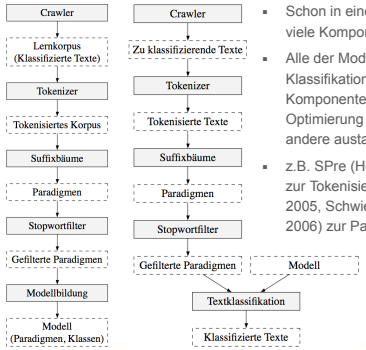
- Eine SALE, die alle drei Arten von Infrastruktur bereitstellt, ist bei der Sprachlichen Informationsverarbeitung in Arbeit: Das *Text Engineering Software Laboratory*
- Tesla: http://www.spinfo.uni-koeln.de/space/Forschung/Tesla
- Implementiert als Client-Sever-Anwendung, mit einem Java EE Application Server, in dem die sprachverarbeitenden Komponenten als Java Beans laufen, mit einem in die Entwicklungsumgebung Eclipse integrierten Client
- Andere vergleichbare Systeme: GATE (http://www.gate.ac.uk/) oder UIMA (http://www.research.ibm.com/UIMA/)



Tesla: Aufbau



Unsere Komponenten bei Wissenserwerb u. Klassifikation



- Schon in einem so kleinen Programm viele Komponenten identifizierbar
- Alle der Modellbildung und der Klassifikation vorgelagerte Komponenten könnte man zur Optimierung des Verfahrens gegen andere austauschen
- z.B. SPre (Hermes und Benden 2005) zur Tokenisierung, SOG (Schwiebert 2005, Schwiebert und Rolshoven 2006) zur Paradigmenbildung

Literatur

- **Brückner, T. (2001): 'Textklassifikation', in K. U. Carstensen, C. Ebert, E. Endriss, S. Jekat, R. Klabunde & H. Langer (eds.),** ***Computerlinguistik und Sprachtechnologie*****, Spektrum, Heidelberg, Berlin, pp. 442–447.**
- **Cunningham, H. & K. Bontcheva (2006): 'Computational Language Systems, Architectures', in K. Brown, A. H. Anderson, L. Bauer, M. Berns, G. Hirst & J. Miller (eds.),** ***The Encyclopedia of Language and Linguistics*****, second edn., Elsevier, München.**
- **Goller, C., J. Löning, T. Will & W. Wolff (2000): 'Automatic document classification: A thorough evaluation of various methods', 7.** ***Internationales Symposium für Informationswissenschaft*****.**
- **Gusfield, Dan (1997):** ***Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology*****, Cambridge University Press.**
- **Hermes, Jürgen & Christoph Benden (2005): "Fusion von Annotation und Präprozessierung als Vorschlag zur Behandlung des Rohtextproblems" in: B. Fisseni, H.-C. Schmitz, B. Schröder und P. Wagner (Hrsg.):** ***Sprachtechnologie, mobile Kommunikation und linguistische Ressourcen*****. Beiträge zur GLDV-Tagung 2005 in Bonn. Frankfurt a.M. u.a.: Lang. Sprache, Sprechen und Computer Bd. 8: S. 78-90.**
- **McEnery, T. (2003): 'Corpus Linguistics', in R. Mitkov (ed.),** ***The Oxford Handbook of Computational Linguistics*****, Oxford Handbooks in Linguistics, Oxford University Press, Oxford, pp. 448–463.**
- **Schwiebert, Stephan (2005): 'Entwicklung eines agentengestützten Systems zur Paradigmenbildung'. In: in: B. Fisseni, H.-C. Schmitz, B. Schröder und P. Wagner (Hrsg.):** ***Sprachtechnologie, mobile Kommunikation und linguistische Ressourcen*****. Beiträge zur GLDV-Tagung 2005 in Bonn. Frankfurt a.M. u.a.: Lang. Sprache, Sprechen und Computer Bd. 8: S. 633-646.**
- **Schwiebert, Stephan und Jürgen Rolshoven (2006): 'SOG: Ein selbstorganisierender Graph zur Bildung von Paradigmen'. In: Rapp, Reinhard, Sedlmeier & Zunker-Rapp:** ***Perspectives on Cognition*****. A Festschrift for Manfred Wettler. Lengerich: Pabst Science Publishers.**