Putting back the F in FAQ

Das Thema *"Häufigkeit"* in *"Häufig gestellte Fragen"*
Masterarbeit
Kristijan Madunić
Betreuer: Prof. Dr. Chris Biemann
In Zusammenarbeit mit Tim Neubacher und Dr. Frederik Janssen
November 2014

TECHNISCHE
UNIVERSITÄT
DARMSTADT

FB Informatik Sprachtechnologie

Putting back the F in FAQ

vorgelegte Masterarbeit von Kristijan Madunić

1. Gutachten: Prof. Dr. Chris Biemann
2. Gutachten: Tim Neubacher und Dr. Frederik Janssen

Tag der Einreichung: 17. November 2014

Erklärung zur Masterarbeit

Hiermit versichere ich, die vorliegende Masterarbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 17. November 2014

_____

Kristijan Madunić

Danksagung

Ich möchte mich bei Chris als auch bei Tim und Frederik bedanken, mir die Möglichkeit gegeben zu haben, die vorliegende Masterarbeit zu schreiben.

Ein großer Dank gebührt Chris für die hervorragende Betreuung dieser Arbeit durch sein Engagement, sein fachliches Wissen und seine Anregungen zur Erstellung dieser Arbeit.

Besonders möchte ich mich für die kompotente Begleitung meiner Arbeit aus Sicht der Studienberatung durch Tim und Frederik bedanken und vor allem für Ihre Evaluation des entwickelten Systems im Rahmen dieser Arbeit.

Meinen Freunden, Arbeitskollegen als auch Mitgründern Michael und Jens möchte ich für ihre Geduld und Rücksicht danken.

Meiner Mutter Mila möchte ich dafür danken, mich während meines ganzen Studiums unterstützt und mir den notwendigen Rückhalt gegeben zu haben.

Vor allem möchte ich meinen Dank gegenüber Marlene aussprechen. Sie war immer für mich da in schwieriegen und motivationsnotwendigen Zeiten.

Zusammenfassung

Viele Unternehmen und Organisationen haben heutzutage eine große Anzahl an Kunden oder Mitgliedern. Um die gewaltige Anzahl an Fragen, Problemen und Wünschen der Kunden zu bewältigen, haben die meisten Unternehmen einen Kundendienst als zentrale Anlaufstelle. In der Regel wird ein Helpdesk eingesetzt, eine softwarebasierte Lösung zur Unterstützung der Mitarbeiter im Kundendienst. Der Begriff Helpdesk umfasst verschiedene Systeme zur Unterstützung des Kundendienstes. Ein Helpdesk kann eine Software zum Monitoring der Anfragen, ein Frage-Antwort-System zur Unterstützung der Mitarbeiter bei der Beantwortung von neuen Fragen oder ein Frage-Antwort-System als Self-Service zur Reduzierung der Fragen im Kundendienst sein.
Die Studienberatung des Fachbereichs Informatik an der Technischen Universität Darmstadt ist eine zentrale Anlaufstelle für Studierende. Zur Unterstützung der Studienberatung und zur Bewältigung von Studierendenanfragen wurde im Rahmen dieser Arbeit für die Mitarbeiter der Studienberatung eine zentral zugängliche Webapplikation als Helpdesk entwickelt. Ein Ziel des entwickelten Helpdesks `InSight` war zum einen eine Möglichkeit für die Analyse der E-Mail-Anfragen zu bieten, um die aktuellen Supportwebseiten der Studienberatung auf die Bedürfnisse der Studierenden anzupassen. Aus den vorhandenen E-Mail-Anfragen und den Webseiten der Studienberatung wurden E-Mail-Dokumente und Webdokumente erstellt. Auf Grundlage der E-Mail-Dokumente und der Webdokumente wurden mittels der Latent Dirichlet Allocation jeweils zwei probabilistische Modelle erzeugt. Die Korpora sind anhand der probabilistischen Modelle miteinander verglichen worden, indem die Dokumente aus dem Webkorpus über das Modell des E-Mail-Korpus inferiert wurden. Die beiden Modelle können durch die jeweilige Visualisierung im Helpdesk `InSight` miteinander verglichen werden. Die Latent Dirichlet Allocation erzeugt kein hierarchisches Modell. Um innerhalb der einzelnen *Topics* eine weitere hierarchische Ebene zu gewinnen, wurden die Dokumente innerhalb eines *Topics* mittels der Analyse durch Skip-N-Gramme segmentiert.
Zum anderen war es das Ziel ein Frage-Antwort-System zu entwickeln, welches die Mitarbeiter in der Beantwortung von Fragen unterstützen soll. Das integrierte Frage-Antwort-Systen in `InSight` basiert auf einem mittels der Latent Dirichlet Allocation generierten Modell, welches sowohl aus dem E-Mail-Korpus als auch aus dem Webkorpus erzeugt wurde. Ein weiteres Frage-Antwort-System `iHelp` wird ebenfalls zum Vergleich in der vorliegenden Arbeit präsentiert, welches ebenfalls im universitären Bereich eingesetzt wird. Die Mitarbeiter der Studienberatung bewerteten die Antwortvorschläge des Frage-Antwort-Systems in `InSight` im Vergleich zu den Antwortvorschlägen einer naiven Volltextsuche mit Lucene. Das entwickelte Frage-Antwort-System lieferte bessere Antwortvorschäge als die naive Volltextsuche mit Lucene. Im Rahmen dieser Arbeit wurde ein Prototyp zur Datenanalyse und zur Beantwortung von E-Mail-Fragen entwickelt, welcher die Grundlage für viele weitere Forschungsthemen bietet.

Abstract

Nowdays many companies and organisations have a large number of customers or members. Most of the companies have a customer service as their central interface in order to manage the enourmous number of customer requests, problems and needs. To support employees working in the customer service a software based helpdesk is generally used. The term *"helpdesk"* includes various systems supporting the customer service. A helpdesk may be a software for monitoring the incoming requests, a question answering system for supporting employees answering new questions or a question answering system as a self-service for reducing the flood of questions in the customer service.
The student advisory office of the computer science department at Technische Universität Darmstadt is the central interface for students and their questions. In the scope of this work a helpdesk web application was developed, which is centrally accessible for the employees of the student advisory office, in order to support the student advisory office and to manage the students requests.
The aim of the developed helpdesk `InSight` at the one hand is to provide a possibility for analysing the email requests from students in order to adjust the current student advisory office support websites to the students needs. The email documents and the web documents were created from the past email requests as well as from the current websites of the student advisory office. Two topic models were generated by the latent Dirichlet allocation from the email coprus and the web corpus. The two corpora were compared with the generated topic models. The web documents were assigned to topics from the email topic model. The inference results and the two topic models are visualized through the developed helpdesk `InSight`. The generated topics by the latent Dirichlet allocation have no hierarchical structure. Each topic in the model was analyzed through skip N-grams to get another hierarchical layer for each topic.
At the other hand the second goal was to develop a question answering system which can support employees in answering questions. The question answering system is based on the latent Dirichlet allocation. The topic model was generated from the email corpus and the web corpus. Another question answering system `iHelp` is presented for the comparison to the developed one in the scope of this work. Both questions answering systems are used in the university environment. The employees of the student advisory service judged the answers from the developed question answering system in comparison to a full text search with Lucene. The developed question answering system in `InSight` gives better answer proposals as Lucene. In the scope of the presented work a prototype was developed which provides a basis for further research projects in this area.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einführung

Der Kundendienst oder der Kundenservice (Support) ist einerseits eine organisatorische Einheit (Abteilung) in einer Organisation oder in einem Unternehmen, anderseits eine Leistung als auch die Dienste dieser Abteilung oder auch des ganzen Unternehmens für ihre Mitglieder sowie Kunden. Jedes Unternehmen braucht einen Support für die Kommunikation im Lebenszyklus mit seinen Kunden. Es ist in der Regel die einzige und zentrale Schnittstelle zum Kunden. Die Kunden kommunizieren ihre Wünsche, Probleme, Fragen und Anregungen mit dem Support. Das Ziel des Supports ist einerseits einen hervorragenden und zufriedenstellenden Service zu leisten, um eine hohe Kundenzufriedenheit zu erreichen. Andererseits hat der Support eine zentrale Aufgabe der Wissenansammlung und Weiterleitung der Informationen an andere Unternehmenseinheiten, um das ganze Unternehmen in allen seinen Prozessen, Dienstleistungen und Produkten auf Basis der Kundeninformationen zu optimieren. Ein guter Support führt zu einer guten Reputation des Unternehmens und somit zu einem guten Ansehen. Mit der Kundenzufriedenheit und der Wissensweitergabe innerhalb des Unternehmens steigt die Kundenanzahl sowie der Erfolg eines Unternehmens. Die Kunden wollen ihre Angelegenheiten, die sie mit dem Support per E-Mail kommunizieren, sowohl schnell und ohne Komplikationen als auch mit souveräner Kompetenz zu ihrer Zufriedenheit erledigt haben. Selbst in den Universitäten spielt der Support eine zentrale Rolle, weil sich jedes Jahr viele neue Studierende in eine Univsersität immatrikulieren, die viele verschiedene Fragen haben und eine zentrale Anlaufstelle als Kommunikation brauchen. Im Laufe des Studiums werden Studierende oft mit Problemen oder Hindernissen konfrontiert. In solchen Situationen brauchen Studierende eine zentrale Einheit, an die sie sich wenden können, um beraten zu werden. Um alle Anforderungen des Supports zu erfüllen, müssen die Mitarbeiter im Support über alle möglichen Themen Bescheid wissen, die spezifischen Fragen beantworten können und alles mit Schnelligkeit bearbeiten. In der Regel ist eine Schulung der Mitarbeiter im Support in allen vorhanden Themen und der Wissensaneignung von allen möglichen Fragen zu den Themen, die Kunden stellen könnnten, nicht zumutbar als auch zeitlich und aus Kostengründen unwirtschaftlich. Zumeist wird der Support als Kostenstelle im Unternehmen angesehen, an der gespart werden sollte. Investitionen in die Schulung von Support-Mitarbeitern sind somit mit schwierigen kaum überwindbaren unternehmensinternen Hürden verbunden. Der Support braucht ein System, welches zentral für alle Mitarbeiter zugänglich ist. Dieses System sollte die Informationen in repräsentativer und zusammengefasster Form darstellen können, damit alle beteiligten Institutionen im Unternehmen aus diesen Informationen Wissen gewinnen, um die jeweiligen Prozesse, Dienstleistungen und Produkte zu optimieren. Außerdem sollte das System die Mitarbeiter in der Bewältigung von Kundenanfragen unterstützten, indem es dem Supportmitarbeiter bei der Lösung der Kundenprobleme hilft. Die zentrale Software, die im Support die aufgeführten Aufgaben erfüllen soll, wird in der Regel Helpdesk genannt.

Die heutigen Helpdesks erfüllen die oben aufgeführten Anforderungen der Wissenspeicherung, -verarbeitung und -darstellung sowie der Unterstützung der Support-Mitarbeiter durch Antwortforschläge auf neue Anfragen nicht in dem gewünschten Umfang. In vielen Fällen wird ein Helpdesk als allgemeine zentrale Erfassung, Protokollierung, Beobachtung und Überwachung von Kundenanfragen angesehen. Diese Aufgaben werden durch den Begriff Monitoring zusammengefasst. Diese Helpdesks bieten weder eine repräsentative noch hilfreiche Darstellungen der Daten zur Wissensextrahierung. Die Monitoringaufgaben werden manuell erfasst. Ein Beispiel für solch ein Helpdesk, welches im Großen und Ganzen eine zentrale Monitoring-Software ist, ist OTRS[1]. Es gibt weitere Helpdesks, die dem Mitarbeiter im Support durch Antwortvorschläge in der Kundenkommunikation als auch Problemfindung helfen. Das Problem dieser Helpdesks ist das Auffinden von relevanten Dokumenten aus der Vergangenheit, die zur aktuellen Anfrage passen. Die Suche nach relevanten Vorfällen aus der Vergangenheit erfolgt stichwortartig und anhand von markierten vordefinierten Wörtern, was in der Regel nicht zum gewünschtem

[1] https://www.otrs.com (23.10.2014)

Ergebnis führt. In einer Stichwortsuche steckt zu wenig semantische Information über die aktuelle Anfrage, um zufriedenstellende und nutzbare Suchergebnisse zu liefern. Ein Beispiel für solch ein Helpdesk ist Zendesk[2].
Das Ziel im Rahmen dieser Masterarbeit ist es zum einen ein Helpdesk für die Studienberatung des Fachbereichs Informatik an der Technischen Universität Darmstadt zu entwickeln, welches die E-Mail-Anfragen und die Webseiten der Studienberatung miteinander vergleicht, um so die Webseiten auf die Bedürfnisse der Studenten anzupassen und aktuell zu halten. Zum anderen soll ein Frage-Antwort-System entwickelt werden, welches den Mitarbeitern der Studienberatung bei der Beantwortung von neuen E-Mail-Anfragen hilft.
In Unterkapitel 1.1 wird die Notwendigkeit eines Helpdesk für die Studienberatung der Informatik an der Technischen Universität Darmstadt begründet. In Unterkapitel 1.2 folgt eine Zusammenfassung der Struktur der vorliegenden Arbeit.

1.1 Motivation

Die Universitäten erleben einen großen jährlichen Zuwachs an Studierenden, wobei das Budget für operative Tätigkeiten sehr begrenzt ist. Zu den operativen Tätigkeiten zählen die Betreuung und adäquate Hilfestellung für Studierende. Support-Mitarbeiter werden gebraucht, um das Wissen von allen allgemeinen Fragen bis hin zu spezifischen Vorlesungsfragen zu beantworten sowie die immer wachsende Flut an Studierendenanfragen adäquat zu bewältigen. Der Fachbereich Informatik an der Technischen Universität Darmstadt erhält viele E-Mail-Anfragen von Studierenden, welche manuell von Mitarbeitern beantwortet werden. Die Anzahl der notwendigen Mitarbeiter wächst im Allgemeinen mit der Anzahl der E-Mail-Anfragen, obwohl ein Großteil der Anfragen wiederkehrende Fragen sind. In der Regel findet man die Antwort zu den Fragen der Studierenden auf den Internetseiten der Studienberatung. Leider finden die Studierenden die gesuchten Informationen nicht auf den Internetseiten. Oft liefert die Suche über Google nicht die gewünschten Internetseiten als Suchtreffer, weil diese nicht anhand des Vokabulars der E-Mail-Anfragen suchmaschinenoptimiert sind. Die gesuchten Informationen auf den Internetseiten der Studienberatung sind nicht auf den ersten Blick ersichtlich, weil sich diese oft auf Unterseiten befindet, die zu tief in der Internetseitenhierarchie sind und somit nicht auf Anhieb gefunden werden oder ersichtlich ist, dass sich die gesuchte Information auf den entsprechenden Unterseiten befindet. Viele Informationen sind in PDF-Dokumenten niedergeschrieben, die auf den Internetseiten verlinkt sind, welche wiederum auch nicht auf Anhieb als mögliche Informationsquellen von Studenten identifiziert werden.
Das Ziel der Masterarbeit ist die Minimierung des Aufwands der Studienberatung zur Beantwortung der E-Mail-Anfragen. Hierzu wird eine Helpdesk entwickelt. Einerseits soll das Helpdesk ein Frage-Antwort-System beinhalten, damit E-Mail-Anfragen schneller beantwortet werden können. Das Frage-Antwort-System kann auch für einen Self-Service auf der Internetpräsenz der Studienberatung genutzt werden, welches im optimalen Fall zu einer Reduzierung der E-Mail-Anfragen führen sollte. Andererseits soll die Struktur, der Inhalt und die Hierarchie der Internetseiten verbessert werden, welches auch zu einer Abnahme an E-Mail-Anfragen führen sollte. Eine manuelle Analyse der Internetseiten auf Grundlage der E-Mail-Anfragen ist auf Grund der großen Datenmenge nicht zumutbar. In beiden Anwendungsfällen steht folgende Forschungsfrage im Vordergrund:

HELFEN AUTOMATISCH GENERIERTE TOPICS BEI DER ZUORDNUNG VON EMAILANFRAGEN ZU WEBDOKUMENTEN ?

[2] http://www.zendesk.de (23.10.2014)

1.2 Übersicht

Kapitel 2
Die Grundlage für die entwickelten und erforschten Ansätze im Rahmen dieser Arbeit sind probabilistische Modelle. Eines dieser statistischen Modelle ist die Latent Dirichlet Allocation (LDA), die in Kapitel 2 beschrieben ist und in diesem Zusammenhang verwendet wurde.

Kapitel 3
Eine Webapplikation, die als Self-Service zur Beantwortung von Studentenanfragen entwickelt wurde, ist in Kapitel 3 dargestellt. Die Rahmenbedingungen und Datengrundlage sind ähnlich zu der vorliegenden Arbeit.

Kapitel 4
In Kapitel 4 werden die einzelnen implementierten Komponenten, die Technologie und die Architektur des im Rahmen dieser Arbeit entwickelten Helpdesks `InSight` vorgestellt.

Kapitel 5 und 6
Die Korpora, die in dieser Arbeit miteinander verglichen werden, sind der E-Mail-Korpus und der Webseiten-Korpus. Die Verarbeitung und Aufbereitung der Dokumente für die weitere Nutzung werden jeweils in Kapitel 5 für Webseiten und in Kapitel 6 für E-Mail-Anfragen beschrieben.

Kapitel 7
Um im generisch erstellten probabilistischen Modell eine Hierarchieebene hinzuzufügen, wird in Kapitel 7 ein naiver Ansatz mittels Skip-N-Grammen vorgestellt.

Kapitel 8
Durch die Analyse der E-Mail-Anfragen soll Wissen extrahiert werden, welches zur Optimierung von den Supportwebseiten der Studienberatung führen soll. Wie genau die E-Mail-Anfragen mit den Webseiten verglichen werden, um Wissen zu extrahieren, ist in Kapitel 8 präsentiert.

Kapitel 9
Das entwickelte Frage-Antwort-System ist in Kapitel 9 dargestellt. Von der Fragenanalyse bis hin zur Antwortselektion werden alle relevanten Schritte aufgelistet.

Kapitel 10
In diesem Kapitel 10 wird der entwickelte Ansatz aus Kapitel 9 gegen eine naive Volltextsuche mit Lucene getestet.

Kapitel 11 und 12
Mögliche Erweiterungen und weitere Forschnungsthemen sind in Kapitel 11 zu finden. In Kapitel 12 folgt ein abschließende Aussage zum Helpdesk `InSight`, welches im Rahmen dieser Arbeit entwickelt wurde.

2 Latent Dirichlet Allocation

Probabilistic topic models are a suite of algorithms
whose aim is to discover the hidden thematic structure
in large archives of documents

David M. Blei [4]

In diesem Kapitel wird das probabilistische Topic-Modell, die Latent Dirichlet Allocation (LDA) [5], beschrieben. Das LDA dient in dieser Arbeit als Grundlage für den inhaltlichen Vergleich von zwei unterschiedlichen Korpora, dem E-Mail-Korpus und dem Webkorpus.

Aufgrund der immer stärker wachsenden Anzahl an textuellen digitalen Dokumenten ist es heutzutage nicht zumutbar diese manuell zu verarbeiten. Neue Methoden und Verfahren werden gebraucht, um die Menge an Dokumenten zu organisieren, zu durchzusuchen und zu verstehen [4]. Eine mögliche Lösung zu diesem Problem bieten probabilistische Topic-Modelle. Die probabilistischen Topic-Modelle bestehen aus einer Ansammlung von Algorithmen, die ohne Vorverarbeitungsschritte oder vorher manuell markierte Daten einsetzbar sind. Das Ziel dieser Topic-Modelle ist eine große Anzahl von Dokumenten anhand ihrer versteckten thematischen Struktur zu explorieren. Eines der einfachsten probabilistischen Topic-Modelle ist das LDA.
Die Ansammlung an statistischen eingesetzten Algorithmen soll durch die beobachteten Variablen, den sichtbaren Dokumenten und deren Wörtern, die unsichtbaren Variablen, die nicht sichtbaren thematischen Strukturen (Topics), erschließen. Ein Topic ist als eine Verteilung über ein fixes Vokabular definiert, das Vokabular beinhaltet alle Wörter aus dem gegebenen Korpus. Ein Datensatz aus Dokumenten und deren Wörtern dient als Eingabe im LDA. Die Ausgabe ist eine vorher festgelegte Anzahl an Topics und deren Topic-Wort-Verteilung als auch die Dokument-Topic-Verteilung. Im LDA wird davon ausgegangen, dass ein Dokument aus mehreren Topics generiert wurde. Durch die ermittelten Verteilungen des LDAs ist eine Zusammenfassung der Dokumente durch die Topics gegeben, Dokumente können anhand der Topics klassifiziert werden, Beziehungen zwischen den Dokumenten können anhand der Topics aufgedeckt werden und Frage-Antwort-Systeme können durch die semantischen Informationen optimiert werden. Das LDA basiert auf dem Latent Semantic Indexing (LSI) [6] und probabilsitischer Latent Semantic Analysis (pLSA) [7].

Notation und Terminologie

In diesem Abschnitt wird die im Folgenden benutzte Notation und Terminologie eingeführt sowie beschrieben, die auch in [8] verwendet wurde.

- T Die Anzahl der Topics.
- D Die Anzahl der Dokumente.
- V Die Größe des Vokabulars.
- N_d Die Anzahl der Wörter innerhalb eines Dokuments d.
- $\phi^{(z)}$ Die Topic-Wort-Verteilung für ein Topic z.
- $\theta^{(d)}$ Die Dokument-Topic-Verteilung für ein Dokument d.

- β Dirichletparameter zur Schätzung der Topic-Wort-Verteilungen ϕ.
- α Dirichletparameter zur Schätzung der Dokument-Topic-Verteilungen θ.

Generativer Prozess

Die Topic-Modell-Annahme geht davon aus, dass die Topics vor den Dokumenten entanden sind und die Dokumente anhand dieser Topics generiert wurden. Ein Topic ist eine Verteilung über ein fixes Vokabular an Wörtern. Jedes Topic weist eine andere Verteilung auf und es wird von einer festen Anzahl von T Topics ausgegangen. In jedem Dokument sind mehrere Topics vorhanden, die über die einzelnen Dokumente unterschiedlich verteilt sind. Der generative Prozess vom LDA ist folgendermaßen definiert:

1. Für alle Topics z aus T:
   a) Bestimmung der zufälligen Wortverteilung für z: $\Phi^{(z)} \sim Dirichlet(\beta)$
2. Für alle Dokumente d aus D:
   a) Bestimmung der zufälligen Topicverteilung für d: $\theta^{(d)} \sim Dirichlet(\alpha)$
   b) Für jedes Wort w_i aus d:
      i. Bestimmung eines zufälligen Topics aus der Dokument-Topic-Verteilung $\Theta^{(d)}$: $z_i \sim Multinomial(\theta^{(d)})$
      ii. Auswahl eines zufälligen Wortes aus der entsprechenden Topic-Verteilung $\Phi^{(z_i)}$: $w_i \sim Multinomial(\Phi^{(z_i)})$

Dieser generative Prozess spiegelt die Annahme wieder, dass Dokumente aus mehreren Topics generiert wurden und führt zur folgenden Gleichung:

$$p(w, z, \theta, \phi | \alpha, \beta) = p(\phi|\beta)p(\theta|\alpha)p(z|\theta)p(w|\phi_z) \tag{2.1}$$

Der generative Prozess als Plate-Notation ist in Abbildung 2.1 abgebildet.

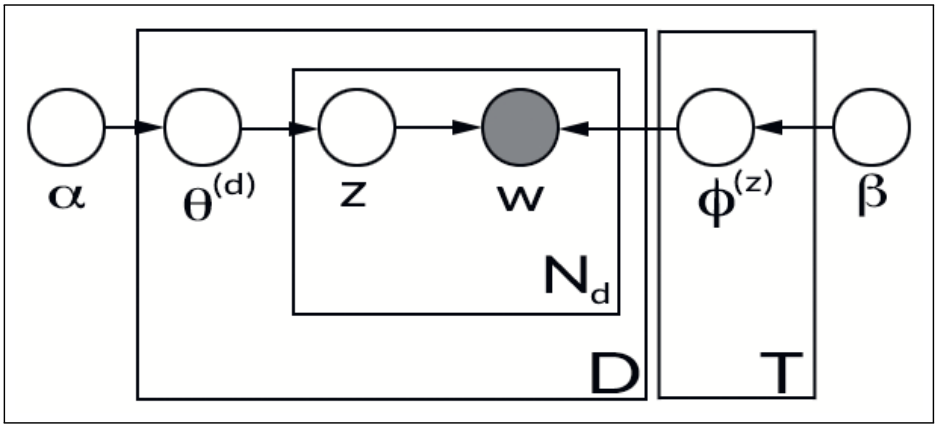

Abbildung 2.1: Plate-Notation des LDA Models

In [5] wurde die Schwierigkeit der Berechnung dieses generativen Models beschrieben, als Alternative wurde zur Schätzung die Mean Field Variational Inference benutzt. Weitere Arten von Inference sind in [9] und [10] präsentiert. In Rahmen dieser Arbeit wurde die Gibbs Sampling Methode aus [11] verwendet, die im Folgenden beschrieben wird.

Gibbs Sampling

Ist ein Datensatz mit einer Menge von Dokumenten gegeben und eine feste Anzahl an T Topics festgelegt, kann mit dem Gibbs Sampling Algorithmus sowohl die Topic-Wort-Verteilung als auch die Dokument-Topic-Verteilung gelernt werden. Das Gibbs Sampling ist ein Markov Chain Monte Carlo (MCMC) Algorithmus [12]. Im ersten Schritt werden die Dokumente und die Wörter in jedem Dokument zufällig zu den Topics verteilt. Im nächsten Schritt wird durch jedes Wort in jedem Dokument iteriert und eine neue Topic-Wort-Zuordnung für das jeweilige Wort im Dokument ermittelt, um sowohl die Topic-Wort-Verteilungen als auch die Dokument-Topic-Verteilungen zu verbessern. Durch die Wiederholung des Schrittes und Optimierung der Verteilungen konvergiert das Gibbs Sampling nach einer bestimmten Anzahl an wiederholten Schritten. Das Sampling eines neuen Topics j für das Wort i in einem Dokument d ist durch die Approximation in 2.2 formuliert, welche aus [8] übernommen wurde.

$$P(z_i = j | z_{-1}, w_i, d_i, \cdot) \propto \frac{C^{WT}_{w_i j} + \beta}{\sum_{w=1}^{W} C^{WT}_{wj} + W\beta} \times \frac{C^{DT}_{d_i j} + \alpha}{\sum_{t=1}^{T} C^{DT}_{d_i t} + T\alpha} \tag{2.2}$$

$z_i = j$ bezeichnet die Topic-Zuordnung j zum Wort i. z_{-i} sind die restlichen Wort-Topic-Zuordnungen. Der Punkt $\cdot$ steht für alle anderen observierten Variablen: w_{-i}, d_{-i}, α und β. α und β sind Dirichlet Parameter. C^{WT} ist eine Matrix mit der Wort-Topic-Zuweisung und C^{DT} ist die Matrix mit der Dokument-Topic-Zuweisung. C^{WT}_{wj} beinhaltet die Anzahl, wie oft das Wort w dem Topic j zugewiesen wurde, ohne das akutelle Wort w_i zu berücksichtigen. C^{DT}_{dj} beinhaltet die Anzahl, wie oft das Wort w dem Topic j innerhalb des Dokument d zugewiesen wurde, ohne die akutelle Zuweisung des Wortes w_i zu berücksichtigen. Der linke Teil der Multiplikation in Gleichung 2.2 ist die Wahrscheinlichkeit des Wortes w_i gegeben das Topic t. Der rechte Teil der Multiplikation ist die Wahrscheinlichkeit für das Topic j im aktuellen Dokument d_i bei gegebener aktueller Topic-Verteilung. Die Approximation in Gleichung 2.2 wird in einer Iteration für alle Wörter von allen Dokumenten durchgeführt. Nach einer gewissen unbekannten Anzahl von Iteration konvergiert die Approximation.
Der beschriebene Ansatz berechnet das warscheinlichste Topic für ein Wort. Für das LDA Modell ist sowohl die Topic-Wort-Verteilung $\phi^{(z)}$ als auch die Dokument-Topic-Verteilung $\theta^{(d)}$ von Bedeutung, diese können durch die Matrixen in Gleichung 2.2 gewonnen werden:

$$\phi^{(z)}_w \propto \frac{C^{WT}_{wz} + \beta}{\sum_{n=1}^{W} C^{WT}_{nz} + V\beta} \tag{2.3}$$

$$\theta^{(d)}_z \propto \frac{C^{DT}_{dz} + \alpha}{\sum_{m=1}^{D} C^{DT}_{mz} + T\alpha} \tag{2.4}$$

Mit dem vorgestellten Verfahren in diesem Kapitel werden aus Dokumenten in einem Korpus Themen generiert, über die in den Dokumenten geschrieben wurde. Mit dem LDA wird ein Topic-Modell aus dem E-Mail-Korpus und Topic-Modell aus dem Webkorpus erzeugt, die zwei Korpora werden mittels der generierten Modelle miteinander verglichen. Des Weiteren wird für das im Rahmen dieser Arbeit entwickelte Frage-Antwort-System im Helpdesk `InSight` ein probabilistisches Modell aus beiden Korpora generiert.

3 iHelp

In diesem Kapitel wird ein entwickelter webbasierter Self-Service iHelp in Form eines Frage-Antwort-Systems präsentiert. iHelp wird auch in einem universitären Umfeld mit gleicher Zielsetzung eingesetzt: Die Minimierung von Anfragen an den Support. Im Rahmen dieser Arbeit ist auch ein Frage-Antwort-System entwickelt worden, um die Mitarbeiter in der Beantwortung der Studierendenanfragen zu unterstützen, welches auch als Self-Service angeboten werden könnte. Nach der detaillierten Beschreibung und Funktionsweise von iHelp folgt am Ende des Kapitels eine kritische Anforderungsanalyse von iHelp zur Aufgabenstellung des Frage-Antwort-Systems im Rahmen dieser Arbeit.

Unter dem Begriff Helpdesk sind verschiedene Arten von Lösungen zur Unterstützung des Supports zu verstehen. Das Helpdesk-System iHelp [1] wurde als Online-Self-Service entwickelt. Das primäre Ziel von iHelp ist es, den Support-Aufwand von Mitarbeitern einer Universität durch ein webbasiertes automatisches Frage-Antwort-System zu minimieren. Die Studierenden geben ihre Frage in ein Eingabefeld ein, wie in Abbildung 3.1 dargestellt. Die Antwortvorschläge des Helpdesk-Systems sind in zwei Kategorien gegliedert. Einerseits in automatisch erstellte Zusammenfassungen *Reference Solutions* und andererseits in eine Auflistung von relevanten Vorfällen aus der Vergangenheit zur aktuellen Frage *Relevant Exisiting Cases*. Der Nutzer kann sich weitere relevante automatisch erstellte Zusammenfassungen anzeigen lassen oder kann einen relevanten Vorfall aus der Vergangenheit betrachten, um eine Lösung zu seinem eigenen Problem zu finden.

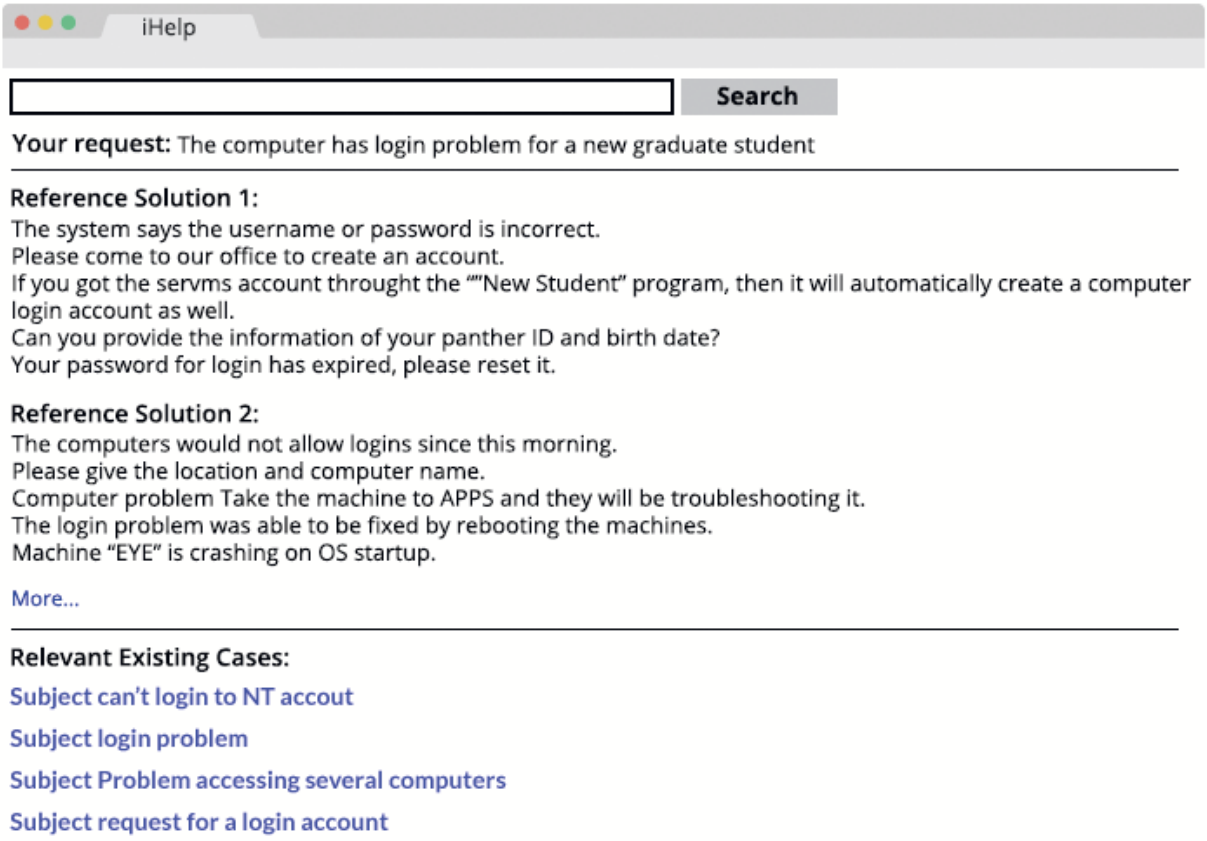

Abbildung 3.1: Benutzerschnittstelle von iHelp (nach Vorlage von Abbildung 2 aus [1])

Das Framework von iHelp ist in Abbildung 3.2 dargestellt. Die Wissensquelle für das System bilden die Kommunikationen aus der Vergangenheit *Past cases* zwischen den Studenten und den Mitarbeitern der Universität. Insgesamt wurden 30000 Dokumente verwendet. Ein Dokument ist die komplette Kommunikation mit der Anfrage eines Studierenden bis hin zur Antwort der Mitarbeiter der Universität über einen Vorfall.

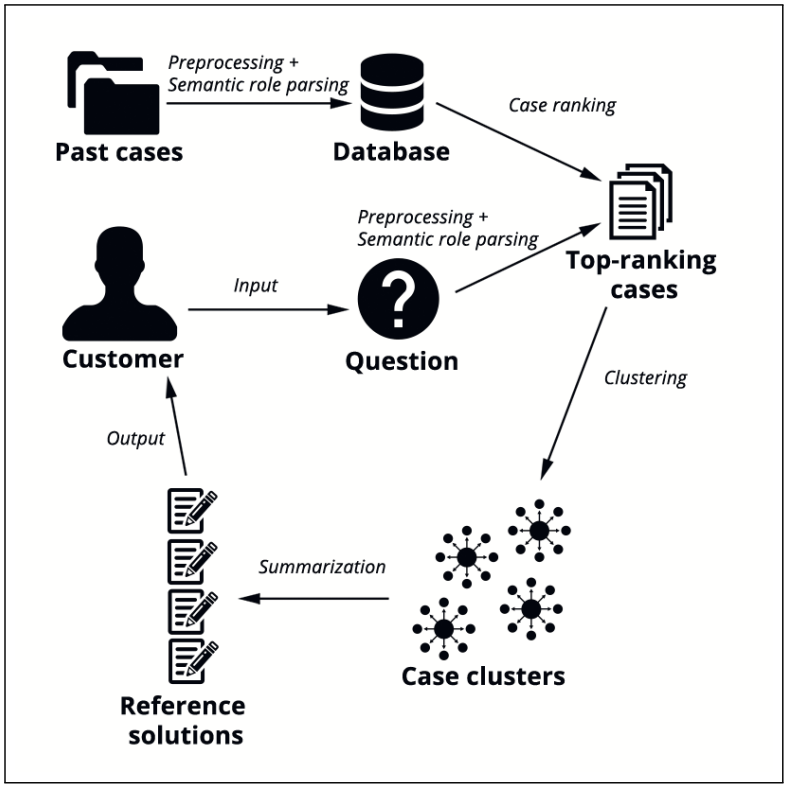

Abbildung 3.2: Framework von iHelp (nach Vorlage von Abbildung 1 aus [1])

Der dargestellte Prozess durch das Framework hat vier Hauptschritte, die in den folgenden Unterkapiteln detailliert behandelt werden. Zunächst werden im ersten Hauptschritt, der Vorverarbeitung der Dokumente, zwei wesentliche Schritte durchgeführt, bevor die Dokumente in die Datenbank gespeichert werden. Der erste Schritt ist eine allgemeine Vorverarbeitung der Dokumente, die die Entfernung von Formartierungszeichen und Stopwords beinhaltet. Der zweite Schritt wird *Semantic role parsing* genannt. Ziel des *Semantic role parsing* ist die Identifizierung der Beziehung des Prädikats mit den syntaktischen Konstituenten. Die gleichen Vorverarbeitungsschritte werden mit der Frage durchgeführt. Die Frage wird mit den vorhandenen Dokumenten verglichen und die Dokumente werden nach der Relevanz zur Frage im Schritt *Case ranking* sortiert. Alle relevanten Dokumente zu einer Anfrage nach dem *Case ranking* beinhalten Informationen zur Frage. Sie unterscheiden sich aber immer noch im Kontext. Zum Beispiel können bei einem Computer-Problem die Ursachen ganz unterschiedlich sein. Deswegen werden die Dokumente segmentiert (*Clustering*), um die Dokumente nach dem speziell inhaltlichen Kontex zu gliedern. Für jede Themengruppe (Segment) wird nach dem *Clustering* eine automatische Zusammenfassung *Summarization* erstellt, die dem Fragesteller unter anderem als Antwortmöglichkeiten *Reference solutions* zurückgegeben werden.

A semantic role is a description of the relationship that a constituent plays with respect to the verb in the sentence.

Arnold et al. [13] S. 205

Das *Semantic role parsing* ist Teil der Vorverarbeitung der Dokumente. Ein Dokument ist der ganze Konversationsverlauf einer Studentenanfrage mit der dazugehörigen Antwort der Mitarbeiter der Universität. Eine neue gestellte Frage im iHelp durchläuft auch erstmal den Schritt *Semantic role parsing* bevor sie mit den vorhandenen Dokumenten verglichen und die relevantesten Dokumente zur Frage herausgesucht werden.
Nachdem die Formatierungszeichen und Stopwörter aus den Dokumenten entfernt wurden, wird das Dokument in Sätze gegliedert. Für jeden Satz wird nun das *Semantic role parsing* durchgeführt. Die grundlegende Idee ist jedes Verb aus dem Satz mit den dazugehörigen sinngebenden Argumenten zu markieren. Das Subjekt im Satz ist in vielen Fällen der *Agent* oder *Experiencer*. Weitere sinngebende Argumente sind zum Beispiel *Patient*, *Instrument*, *Locative* usw.: Die Katze (*Agent*) jagt den Hund (*Patient*). Die dazugehörenden Argumente zu einem Verb geben somit zum Beispiel Antwort auf die Fragen: *wer?*, *wann?*, *was?*, *wo?* oder *warum?*. Die Markierung für jedes einzelne Verb im Satz wird *Frame* genannt. Pro Satz gibt es also genau so viele *Frames* wie Verben. Um diese semantischen Informationen aus dem Satz zu gewinnen wird in iHelp NEC SENNA [14] verwendet, welches auf der PropBank-Semantik-Annotation [15] basiert. Im Folgenden wird folgender Beispielsatz[1] betrachtet:

```
Google announced a new product yesterday.
```

SENNA's Ausgabe des aufgeführten Satzes ist:

Google	NNP	S-NP	S-ORG	-	S-A0
announced	VBD	S-VP	O	announced	S-V
a	DT	B-NP	O	-	B-A1
new	JJ	I-NP	O	-	I-A1
product	NN	E-NP	O	-	E-A1
yesterday	NN	S-NP	O	-	S-AM-TMP
.	.	O	O	-	O

Die Ausgabe bezieht sich auf das *Frame* mit dem Verb *announced*. Ein paar Annotationsbedeutungen sind im Folgenden aufgelistet:

```
V:          verb
A0:         acceptor
A1:         thing accepted
A2:         accepted-from
A3:         attribute
AM-MOD:     modal
AM-NEG:     negation
```

[1] Beispiel von http://nlpb.blogspot.de/2011/02/senna-fast-semantic-role-labeling-srl.html (20.10.2014)

Die extrahierten *Frames* durch das *Semantic role parsing* zu jedem Satz in den vorhanden Dokumenten werden in der Datenbank mitgespeichert. Die Frage durchläuft auch das *Semantic role parsing*. Auf Grundlage der gewonnen Frames durch die Frage und die Dokumente werden die relevantesten Dokumente zu einer Frage im Schritt *Case ranking* ermittelt, der im folgenden Unterabschnitt dargestellt wird.

Case ranking

Im ersten Schritt *Semantic role parsing* wurden semantische Informationen durch den Semantic-Role-Parser NEC SENNA [14] aus den Dokumenten extrahiert. Bei einer neu gestellten Frage werden die gleichen Vorverarbeitungsschritte durchgeführt. Auf Basis dieser gewonnen Informationen werden die Dokumente nach ihrer Relevanz zum Fragesatz geordnet. Jeder Satz aus einem Dokument wird mit dem Fragesatz verglichen und dabei ein Sentence-Similarity-Score ermittelt. Dem Dokument wird der höchste ermittelte Sentence-Similarity-Score zugewiesen. Die Dokumente werden nach ihrem Document-Similarity-Score absteigend geordnet. Die Scores liegen zwischen 0 und 1.

Für jeden Satz S_i aus einem Dokument D_j wird der Sentence-Similarity-Score zu einer Frage F berechnet. Aus dem Schritt *Semantic role parsing* sind alle *Frames* f_{S_i} von S_i und f_F von F markiert und bekannt. Die Menge $\{r_1, r_2, ..., r_k\}$ beinhaltet alle gemeinsamen Rollen zwischen f_{S_i} und f_F. Die Rollen von Wörtern aus f_{S_i} und f_F werden durch die Datenbank WordNet [16] miteinander verglichen. Zwei Wörter stehen über ihre Rolle zueinander in Beziehung, wenn die Rollen identisch sind oder in einer Relation zueinander stehen wie Synoym, Hypernym, Hyponym, Meronym als auch Holynym. Die Funktion t_{sim} (3.1) überprüft, ob ein Wort existiert, das in beiden Mengen $T_{f_{S_i}}(r_i)$ und $T_{f_F}(r_i)$ enthalten ist und der gleichen Rolle zugewiesen wurde. Als Eingabe erhält die Funktion ein Wort $t_{ij}^{f_{S_i}}$ aus der Menge $T_{f_{S_i}}(r_i)$ und eine Rolle r_i. Die Ausgabe ist entweder 1, wenn $t_{ij}^{f_{S_i}}$ in $T_{f_F}(r_i)$ bei gleichem r_i exisitert, oder 0, falls dies nicht der Fall ist.

$$t_{sim}(t_{ij}^{f_{S_i}}, r_i) = \begin{cases} 1 \text{ , falls } t_{ij}^{f_{S_i}} \in T_{f_{S_i}}(r_i), \exists t_{ij}^{f_F} \in T_{f_F} \\ 0 \text{ , sonst} \end{cases} \tag{3.1}$$

Die Funktion T_sim (3.2) berechnet mit Hilfe der Funktion aus 3.1 die Ähnlichkeit zwischen $T_{f_{S_i}}(r_i)$ und $T_{f_F}(r_i)$ unter der Voraussetzung, dass $|T_{f_{S_i}}(r_i)| < |T_{f_F}(r_i)|$ gilt. Die Funktion iteriert über alle Wörter aus $T_{f_{S_i}}(r_i)$ und addiert 1 dazu, falls das entsprechende Wort unter der gleichen Rolle in $T_{f_F}(r_i)$ gefunden wurde. Am Ende wird durch die Anzahl der Wörter aus $T_{f_F}(r_i)$ geteilt.

$$T_{sim}(T_{f_{S_i}}(r_i), T_{f_F}(r_i)) = \frac{\sum_j t_{sim}(t_{ij}^{f_{S_i}}, r_i)}{|T_{f_F}(r_i)|} \tag{3.2}$$

Die Funktion f_{sim} (3.3) berechnet mit Hilfe der Funktion aus 3.2 die Ähnlichkeit zwischen f_{S_i} und f_F. Dabei wird über alle Rollen mit der Funktion aus 3.1 aufsummiert. Das Ergebnis wird durch die Anzahl der verschiedenen Rollen geteilt.

$$f_{sim}(f_{S_i}, f_F) = \frac{\sum_{i=1}^{k} T_{sim}(T_{f_{S_i}}(r_i), T_{f_F}(r_i))}{K} \tag{3.3}$$

Durch die Funktion f_{sim} (3.3) wird der Similarity-Score von zwei Frames unterschiedlicher Sätze berechnet. In dem vorliegenden Fall wird jeweils zwischen einem Satz aus einem Dokument und dem Fragesatz

der Similarity-Score berechnet. Der Sentence-Similarity-Score ergibt sich aus dem ermittelten Maximum der Frames eines Satzes S_i und der Frage F, wie in Gleichung 3.4 dargestellt.

$$Sim(S_i, F) = \max_{f_{S_i} \in S_i, f_F \in F} f_{sim}(f_{S_i}, f_F) \tag{3.4}$$

Das Ergebnis von Gleichung 3.4 liegt zwischen 0 und 1. Als Document-Similarity-Score eines Dokuments wird das ermittelte Maximum aus den iterierten Berechnungen der Gleichung Sim (3.4) ausgewählt. Die Dokumente d_p sind anhand ihres Document-Similarity-Scores zur Anfrage F absteigend sortiert, welches in folgender Gleichung 3.5 dargestellt ist:

$$Score(d_p, F) = \max_{S_i \in d_p} Sim(S_i, F) \tag{3.5}$$

Das *Case Ranking* berechnet für jedes Dokument bei einer gegebenen Frage den Document-Similarity-Score. Der Document-Similarity-Score drückt aus, wie gut der Inhalt eines Dokuments zu der gestellten Frage passt. Mit diesem Schritt werden die passendsten n Dokumente ermittelt. Für eine gestellte Frage kann es mehrere Antworten geben, die zu der Lösung des Problems führen. Im nächsten Schritt *Clustering* sollen die verschiedenen Antwortmöglichkeiten innerhalb der passendsten n Dokumente identifiziert werden, was im folgenden Unterabschnitt beschrieben ist.

Clustering

Im Schritt *Case ranking* wurden die Dokumente nach ihrer Relevanz zur Anfrage durch einen berechneten Score sortiert. In diesem Schritt *Clustering* werden die Dokumente in Segmente unterteilt, bevor im Folgenden für jedes Segment eine automatische Zusammenfassung für den Nutzer erstellt und als Antwortmöglichkeit präsentiert wird.
Eine Frage könnte zum Beispiel sein *'"mein Computer arbeitet nicht'*. Die möglichen Antworten in diesem Fall sind vielfältig, wie zum Beispiel Systemabsturz oder Festplattenfehler. Um die verschiedenen Antwortmöglichkeiten innerhalb der relevantesten Dokumente für eine Frage zu identifizieren, müssen die relevantesten Dokumente segmentiert werden. Die Segmentierung der relevantesten Dokumente zu einer Frage erfolgt in zwei Schritten. Im ersten Schritt werden die Ähnlichkeiten zwischen den Dokumenten durch ein Mixture-Language-Model [17] berechnet. Der zweiter Schritt ist die Segmentierung der Dokumente anhand der Ähnlichkeitsmatrix durch eine symmetrische nichtnegative Matrixfaktorisierung SNMF [18]. Diese zwei Schritte werden im Folgenden detailliert beschrieben:

1. **Ähnlichkeitsmatrix berechnen**
   Wie in Abbildung 3.3 gezeigt, besteht ein Dokument aus Wörtern, die drei Kategorien zugeordnet werden können. Bei dem gegebenen Beispiel: *"hi, a new student wants to create an e-mail account"*, sind Wörter wie *"an"* oder *"hi"* mit aller Wahrscheinlichkeit nach aus dem Englischen Θ_E. Wörter, die zur Anfrage Θ_R spezifisch gehören sind mit hoher Wahrscheinlichkeit zum Beispiel *"create"* oder *"account"*. Wörter, die eher zum Kontext Θ_C gehören, sind *"new"*, *"email"* oder *"account"*.

   Die Wahrscheinlichkeit für ein Wort aus Θ_E ist λ_E, für Θ_R ist sie λ_R und dementsprechend für Θ_C ist sie λ_C. In folgender Gleichung 3.6 ist die Wahrscheinlichkeit für ein Wort mit der Bedingung: $\lambda_E + \lambda_R + \lambda_C = 1$ formal beschrieben:

$$P(w_i|\Theta_E, \Theta_R, \Theta_C, \lambda_E, \lambda_R, \lambda_C) = \lambda_E P(w_i|\Theta_E) + \lambda_R P(w_i|\Theta_R) + \lambda_C P(w_i|\Theta_C) \tag{3.6}$$

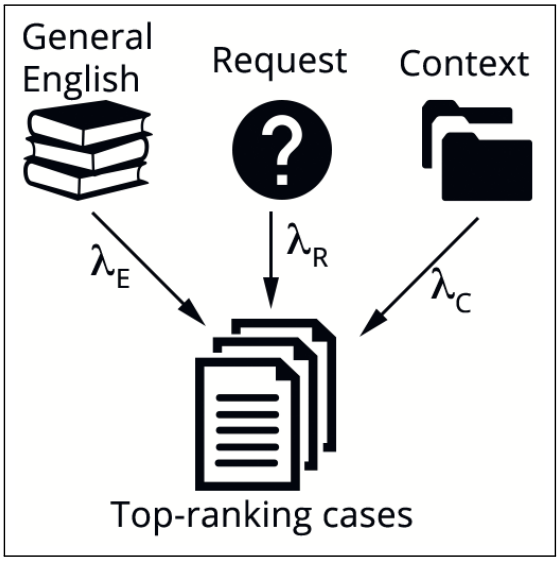

Abbildung 3.3: Mixture-Modell zur Ähnlichkeitsberechnung von Dokumenten (nach Vorlage von Abbildung 3 aus [1])

Die Ähnlichkeit von zwei Dokumenten d_i und d_j lässt sich nach der Gleichung 3.7 berechnen. KL ist die Kullback-Leibler-Divergenz. Θ_C ist das Modell für den Kontext. Die Modelle Θ_E, Θ_R und Θ_C können durch den Expection-Maximization-Algorithmus [19] trainiert werden.

$$S(d_i, d_j) = \frac{KL(\Theta_C(d_i), \Theta_C(d_j)) + KL(\Theta_C(d_j), \Theta_C(d_i))}{2} \tag{3.7}$$

Durch die Benutzung dieses Mixture-Language-Modells sollen Wörter aus dem generellem Englisch als auch frequente Wörter aus der Anfrage für die Betrachtung minimiert werden. Die so ermittelten einzelnen Scores für jeweils zwei Dokumente lassen sich in einer Ähnlichkeitsmatrix darstellen.

2. **Segmentierung der Dokumente**
   Durch die im ersten Schritt durchgeführte Berechnung der Ähnlichkeitsmatrix lassen sich die Dokumente mit dem SNMF-Algorithmus [18] segmentieren. Gegeben ist eine Matrix W (Ähnlichkeitsmatrix) und gesucht ist eine neue Matrix H, die die Gleichung 3.8 mit minimaler Belegung erfüllen soll, wobei $|W - HH^T||^2$ bzw. $||X||^2 = \sum_{ij} X_{ij}^2$ in Frobeniusnorm ist.

$$\min_{H \geq 0} F(W, H) = ||W - HH^T||^2 \tag{3.8}$$

Durch die Iterative Berechnung mit der Gleichung 3.9 bis zur Konvergenz wird die Gleichung 3.8 erfüllt.

$$H_{ik} \leftarrow H_{ik} \sqrt[4]{\frac{[WH]_{ik}}{[HH^T H]_{ik}}} \tag{3.9}$$

Die Anzahl der Segmente wurde durch die Methode in [20] bestimmt. Dabei wurden die Dokumente mehrmals mit einer zufälligen Anzahl an k Cluster segmentiert. Durch den Vergleich der Cluster untereinander wurde das k bestimmt. Je weniger die Cluster sich im Inhalt unterscheiden, desto besser ist das gewählte k.

In diesem Schritt wurden die relevantesten Dokumente zu einer Frage in Segmente unterteilt. Alle Dokumente innerhalb eines Segments beschreiben eine spezifische Lösung (Antwortmöglichkeit) zur gestellten Frage. Im nächsten Schritt wird zu jedem Segment eine automatische Zusammenfassung erstellt.

Summarization

Im Schritt *Clustering* wurden die relevantesten Dokumente zu einer Frage segmentiert, um die verschiedenen Antwortmöglichkeiten bezogen auf die Frage zu identifizieren. In diesem Schritt wird für die Dokumente eines Segments, die alle die gleiche Antwortmöglichkeit beschreiben, eine automatische Zusammenfassung erstellt.

Die Informationen innerhalb der Dokumente in einem Segment überlappen sich. Der Inhalt dieser Dokumente muss zusammengefügt werden, wobei rendundante Informationen identifiziert und entfernt werden. Relevante Informationen sollen durch markante Unterschiede im Inhalt der Dokumente erkannt und zusammengefasst werden [21]. Im Folgenden werden die einzelnen Schritte aufgelistet:

1. Alle Dokumente innerhalb eines Segments aus dem *Clustering*-Schritt in ein Dokument D zusammenfassen.
2. Alle Sätze S_i in D mit der Satz-Semantik-Analyse aus Unterabschnitt *Case ranking* vergleichen. Nach der semantischen Analyse der Sätze entsteht eine Ähnlichkeitsmatrix zwischen den Sätzen.
3. Die Sätze S_i in D durch den SNMF-Algorithmus in Segmente C_S unterteilen. Alle Segmente C_S mit weniger als drei Sätzen werden als Ausreißer betrachtet und entfernt.
4. Für jeden Satz S_i in jedem Segment C_k aus C_S die Relevanz für die Zusammenfassung berechnen.
   a) Für einen Satz S_i die Ähnlichkeit zu den restlichen Sätzen S_j im Segment C_k berechnen, die Funktion $Sim(S_i, F)$ ist in Gleichung 3.4 definiert:

$$F_1(S_i) = \frac{1}{N-1} \sum_{S_j \in C_k - S_i} Sim(S_i, S_j) \tag{3.10}$$

   Der Score von F_1 in Gleichung 3.10 wird durch das Aufsummieren der berechneten Ähnlichkeitswerte zu den restlichen Sätzen innerhalb des Segments C_k berechnet und am Ende durch die Anzahl N an Sätzen in C_k minus eins geteilt.

   b) Für jeden Satz S_i im Segment C_k die Relevanz zur Frage berechnen:

$$F_2(S_i) = Sim(S_i, F) \tag{3.11}$$

   c) Für einen Satz S_i wird der Score berechnet, wie wichtig S_i für die Zusammenfassung ist:

$$Score(S_i) = \lambda F_1(S_i) + (1-\lambda) F_2(S_i) \tag{3.12}$$

   λ ist ein Gewichtsparamter, welcher in diesem Fall mit 0.7 gesetzt ist.

5. Auswahl der relevantesten n Sätze aus D anhand des berechneten Scores aus Gleichung 3.12.

Durch die Satz-Semantik-Analyse und anschließende Segmentierung in Schritt 2 und 3 werden die unterschiedlichen Informationen innerhalb der Dokumente eines Segments aus dem Schritt *Clustering* identifiziert. Im Schritt 4 wird für jeden Satz S_i eines Segments an Sätzen aus Schritt 3 ein Score (Gleichung 3.12) berechnet, wie wichtig dieser jeweilige Satz für die Zusammenfassung ist. Nur die relevantesten Sätze zur gestellten Frage innerhalb *D* sollen in die Zusammenfassung aufgenommen werden.

Die Dokumente innerhalb eines Segments nach dem *Clustering* beschreiben alle die gleiche Antwort zu einer Frage. Unterschiedliche Segmente nach dem *Clustering* beschreiben verschiedene Antwortmöglichkeiten. Im Schritt *Summarization* wurde für jedes Segment automatisch eine Zusammenfassung erstellt, welche dem Fragesteller als Antwortmöglichkeiten vorgeschlagen werden.

Zusammenfassung

In diesem Kapitel wurde ein webbasierter Self-Service iHelp in Form eines Frage-Antwort-Systems als Helpdesk dargestellt, welches in einem universitären Umfeld eingesetzt wird. Das Ziel von iHelp ist auch die Minimierung von Support-Anfragen, indem es als webbasierter Self-Service von Studierenden zur Selbstrecherche genutzt werden kann. Als Eingabe erwartet das System eine Frage, die der Nutzer in ein Eingabefeld eingibt, wie in Abbildung 3.1 zu sehen ist. Die E-Mail-Anfragen erfordern eine andere Vorverarbeitung der Frage als die bis jetzt implementierte Vorverarbeitung in iHelp. Die generierten Antworten in iHelp werden aus alten E-Mail-Antworten der Support-Mitarbeiter generiert. Die Antworten im gegebenen Szenario dieser Arbeit sollen aus dem Inhalt der Webseiten der Studienberatung generiert werden. Der Semantic Role Parser ist eine Schlüsselkompnente in iHelp, den es in dieser Qualität für die deutsche Sprache nicht gibt. Somit kommt iHelp als Software im gegebenen Fall nicht in Frage.

4 InSight

In diesem Kapitel wird das im Rahmen dieser Arbeit entwickelte Helpdesk `InSight` vorgestellt und die einzelnen Komponenten des Softwaresystems als auch deren Zusammenstellung dargestellt. Im Unterkapite 4.1 werden die einzelnen Komponenten und deren Interaktion untereinander gezeigt. Im Unterkapitel 4.2 wird die verwendete Technologie beschrieben und die Architektur präsentiert.

4.1 Die Komponenten

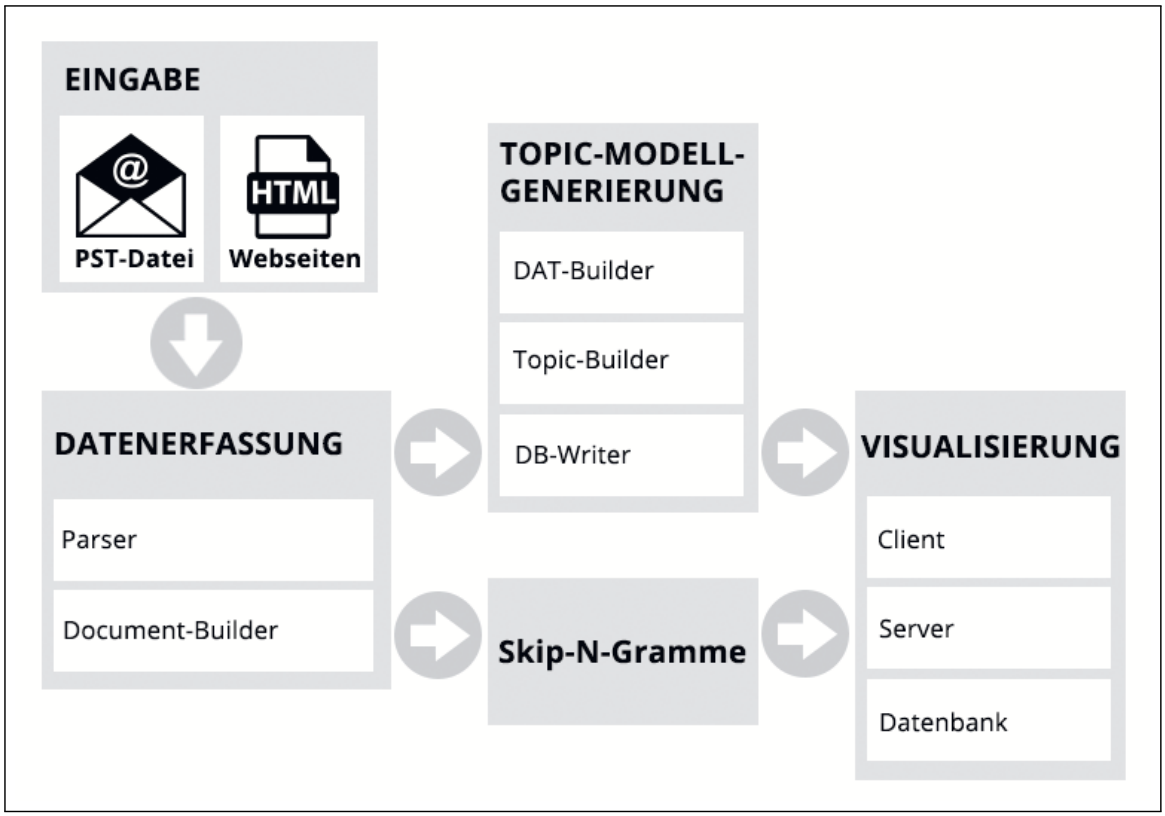

Abbildung 4.1: Übersicht über die Komponenten von `InSight`

Die Abbildung 4.1 zeigt die einzelnen Komponenten des entwickelten Helpdesk `InSight`. Als Eingabe erhält das System eine PST-Datei von vorhandenen E-Mail-Konversationen zwischen den Studierenden mit deren Anfragen und den Antworten der Studienberatung. Die Analyse der E-Mail-Konversationen durch das visuelle System beantwortet die Frage: *Was sind die Probleme, Anliegen und Wünsche der Studierenden über die Zeit verteilt?* Eine weitere Informationsquelle sind die Webseiten der Studienberatung. Aus dem Inhalt der Webseiten werden der Studienberatung Antworten für neue Fragen von Studierenden vorgeschlagen. Die Verarbeitung und Analyse der Webseiten beantwortet die Frage: *Was ist die Antwort auf eine neue Fragen der Studierenden?* Der erste Schritt ist die Datenerfassung, in welchem die PST-Datei und die Webseiten geparst und in einzelne Dokumente gegliedert werden. Der zweite Schritt ist die Generierung des probabilistischen Topic-Modells durch das JGibbLDA[1]. Vorher müssen die erfassten Dokumente aus dem Schritt der Datenerfassung in das Eingabeformat für das JGibbLDA umgewandelt werden. Der Prozess im JGibbLDA produziert Textdateien. Diese Textdateien werden geparst und in ein

[1] http://jgibblda.sourceforge.net (04.11.2014)

Datenbankmodell eingefügt. Weil das generierte Topic-Modell durch das JGibbLDA kein hierarchisches Modell erzeugt, wird durch die Skip-N-Gramme eine weitere Analyse über die Dokumente durchgeführt und eine weitere Hierarchieebene zum generierten Topic-Modell hinzugefügt. Die letzte Komponente ist die Visualisierung, die zur Interaktion mit einem Mitarbeiter der Studienberatung dient. Es beinhaltet die Möglichkeit zur Analyse der Dokumente aus dem E-Mail-Korpus und dem Webkorpus als auch ein Frage-Antwort-System für neue eintreffende Fragen von Studierenden. Die generierten Topic-Modelle durch das JGibbLDA sind wesentlicher Bestandteil in der Datenanalyse und im Frage-Antwort-System.
Die Limitierungen und Verbesserungen des Designs werden in Kapitel 11 besprochen, wie zum Beispiel die Optimierung der Software durch eine direkte Anbindung an den E-Mail-Server, damit keine PST-Dateien manuell eingefügt werden müssen.
In den folgenden Unterabschnitten werden die einzelnen Komponenten detailliert dargestellt.

Datenerfassung

Der erste Schritt in der Datenvorverarbeitung ist die Datenerfassung und Datenextraktion aus den Informationsquellen. Die Schnittstelle zur Eingabe ist der Parser in der Datenerfassung. Es sind zwei Parser vorhanden: Ein PST-Parser, der zum Parsen von PST-Dateien dient, sowie ein HTML-Parser zum Crawlen von Webseiten. Der PST-Parser extrahiert die Nachricht aus dem E-Mail-Verkehr sowie wichtige Metainformationen zu einer Nachricht, wie zum Beispiel den Betreff und das Datum der Nachricht. Nur E-Mails mit Studierendenfragen werden berücksichtigt. Der HTML-Parser parst die relevanten Webseiten der Studienberatung und speichert diese mit der kompletten HTML-Syntax ab. Der E-Mail-Dokument-Builder entfernt Duplikate aus den E-Mails, fügt E-Mails einer E-Mail-Konversation zu einem E-Mail-Dokument zusammen und identifiziert die Sprache als auch die relevanten Inhalte einer E-Mail-Konversation. Der HTML-Dokument-Builder extrahiert aus jeder Webseite die relevanten Informationen und erstellt eins bis mehrere Dokumente aus einer Webseite. Redundante HTML-Dokumente werden gelöscht, da exakte Duplikate einen zu großen Einfluss auf das Ergebnis der Topic-Modell-Generierung haben. Eine detallierte Beschreibung der HTML Datenverarbeitung ist in Kapitel 5 zu finden. Die E-Mail Datenverarbeitung ist in Kapitel 6 genau beschrieben.

Topic-Modell-Generierung

Die generierten Topic-Modelle, die in Kapitel 2 beschrieben wurden, bilden die Grundlage zur Analyse der gegebenen Korpora und sind eine Kernkomponente im Frage-Antwort-System. Für die Topic-Modell-Generierung wird das JGibbLDA verwendet, welches eine Implementierung des LDA [5] ist. Die Software erstellt ein probabilistisches Modell, wie in [4] beschrieben. Die Dokumente aus der Datenerfassung müssen in das Eingabeformat für die Software gebracht werden. Die Ausgabe des Topic-Modells erfolgt in Textdateien. Diese Textdateien werden in ein Datenbankmodell überführt, damit die Visualisierung die Daten über die Datenbank bekommt und die Informationen darstellen kann. Die Topic-Modell-Generierung erstellt drei Modelle, die im Helpdesk `InSight` genutzt werden. Zum einem erstellt es ein Topic-Modell, in dem alle Dokumente von den E-Mails und den Webseiten enthalten sind. Dieses Mix-Topic-Modell wird im Rahmen des im Rahmen dieser Arbeit entwickelten Frage-Antwort-System benutzt, auf welches in Kapitel 9 eingegangen wird. Zudem werden noch ein E-Mail-Topic-Modell und ein Web-Topic-Modell erstellt, welche zur Analyse der Themen der Studierendenanfragen, dem Abgleich der Themen auf den Webseiten als auch deren Optimierung dienen. Das Thema der Wissensextraktion und Analyse der Daten mittels `InSight` wird in Kapitel 8 betrachtet.

Skip-N-Gramme

Das generierte Topic-Modell durch das LDA ist kein hierarchisches Modell, es wird eine Dokument-Topic-Verteilung für eine bestimmte Anzahl an *Topics* erzeugt. Um die relevantesten Dokumente innerhalb eines

Topics weiter zu untersuchen und zu unterscheiden, sind weitere zusätzliche Verfahren notwendig. Durch die Analyse der Dokumente anhand von Skip-N-Grammen, die eine Erweiterung von K-Skip-N-Grammen [22] sind, wird eine Hierarchieebene zum generierten Topic-Modell hinzugefügt. Für jedes *Topic* werden die relevantesten Dokumente mittels Skip-N-Grammen segmentiert. Der Prozess ist unabhängig von der Topic-Modell-Generierung und ist in Kapitel 7 detailliert beschrieben.

Visualisierung

Das Helpdesk `InSight` ist als eine Webapplikation mit einer Client-Server-Architektur und einer Datenbank entwickelt worden. Eine webbasierte Anwendung ist ohne Probleme auf allen Systemen als eine zentrale Anwendung auf allen Clients der Studienberatung ohne eine zusätzliche Installation einsetzbar. Für eine Webapplikation zur Datenanalyse und Interaktion wurden im Rahmen dieser Arbeit folgende aufgelistete Komponenten verwendet:

- **Datenbank**
  Die Software enthält eine relationale Datenbank in Tabellenform. Die Datenbank speichert die geparsten unverarbeiteten E-Mails und Webseiten, die erstellten E-Mail-Dokumente und HTML-Dokumente als auch die Ausgabe der Topic-Modell-Generierung. Die Zuordnung der Topic-Wort-Verteilung, Dokument-Topic-Verteilung, Dokument-Wort-Verteilung als auch die erstellten Skip-N-Gramme und deren Zuordnung zu Dokumenten als auch zu *Topics* werden in der Datenbank gespeichert.

- **Server**
  Der Server ist die Schnittstelle zwischen der visuellen Ausgabe beim Client und der Datenbank im Hintergrund. Der Server speichert keine Informationen, um die Zugriffsrate auf die Datenbank zu minimieren und die Antwortzeit der Operationen und Anfragen durch den Client zu beschleunigen. Der Funktionsumfang des Servers wurde gering gehalten, indem alle logischen Operationen auf den Client ausgelagert wurden und der Server nur als Kommunikationsschnittstelle zwischen dem Client und der Datenbank dient.

- **Client**
  Die Benutzerschnittstelle der Webapplikation von `InSight` bietet eine Menge an Interaktionsmöglichkeiten zur Wissensgewinnung auf Grundlage der Daten sowie ein integriertes Frage-Antwort-System. Die Daten können auf verschiedene Art und Weise exploriert werden. Details zur Interaktion und Visualisierung mit der Software sind in Kapitel 8 zu finden.

4.2 Die Technologie und Architektur

In diesem Kapitel wird die Architektur der Komponenten und die verwendete Technologie erläutert. Die Datenerfassung, die Topic-Modell-Generierung und der Server sind in Java[2] implementiert. Java ist eine verbreitete objektorientierte Programmiersprache mit vielen nützlichen Bibliotheken, die die Entwicklung von Software und die Verarbeitung von Daten erleichtert. In den folgenden Unterabschnitten wird die Technologie der einzelnen Komponenten dargestellt.

Datenerfassung

Die Abbildung 4.2 visualisiert den Ablauf in der Datenerfassung. In der Datenerfassung werden zwei Datenquellen als Informationsquellen genutzt: Die E-Mail-Kommunikation und die Webpräsenz der Studienberatung. Die Datenerfassung hat dementsprechend zwei Parser. Der erste Parser ist für das Parsen

[2] https://www.java.com/de/ (16.11.2014)

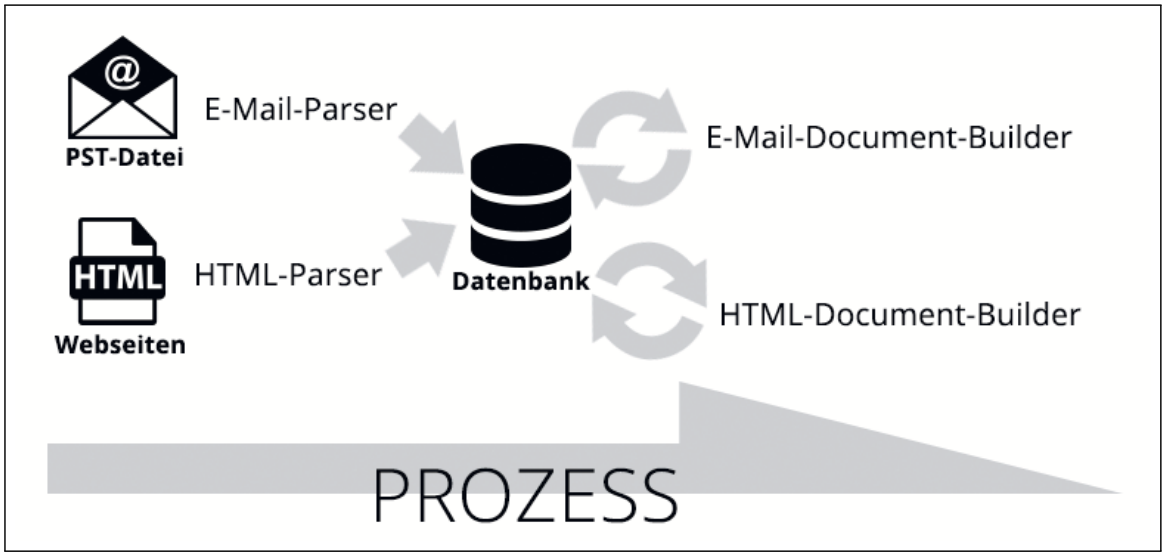

Abbildung 4.2: Ablauf in der Datenerfassung

des Inhalts der PST-Datei mit den E-Mails. Der zweiter Parser ist für das parsen von Webseiten der Studienberatung.

- **E-Mail-Parser**
  Für das Parsen der E-Mails aus der PST-Datei wird die Java-Bibliothek java-libst[3] verwendet. Die Bibliothek ermöglicht ohne andere zusätzliche Bibliotheken den Inhalt einer PST-Datei zu lesen. Jede einzelne E-Mail mit der Nachricht und deren Metainformationen wie zum Beispiel Betreff, Datum, Sender, Empfänger als auch der Verlauf zwischen den E-Mails ist durch die zur Verfügung gestellten Methoden im Interface der Bibliothek gegeben. Implementiert wurde eine Klasse, die mittels der Bibliothek java-libst[2] die einzelnen E-Mails durchläuft und alle notwendigen Informationen in die Datenbank zur Weiterverarbeitung speichert.

- **HTML-Parser**
  Als HTML-Parser wird die Bibliothek jsoup[4] verwendet. Die Bibliothek jsoup erwartet als Eingabe eine URL. Durch die zur Verfügung stehenden Methoden im Interface kann eine gecrawlte Webseite durch HTML-Elemente oder CSS-Klassen exploriert werden. Eine Klasse zum Crawlen der Webseiten der Studienberatung wurde implementiert. Diese Klasse nutzt jsoup[3], um die Startseite zu parsen und zu verarbeiten. Die Unterseiten werden durch die extrahierten Links einer Webseite identifiziert. Das Crawlen ist fertig, wenn keine neuen Unterseiten gefunden werden.

Nachdem die Daten durch die Parser aufgenommen und in die Datenbank gespeichert wurden, erzeugen die Document-Builder bereinigte und für die Weiterverarbeitung relevante Dokumente. In der Datenerfassung sind zwei Document-Builder implementiert. Der E-Mail-Document-Builder entfernt Duplikate aus den E-Mails, indem referenzierte oder beantwortete E-Mails, die im Verlauf der E-Mail-Kommunikation enthalten sind, entfernt werden. Außerdem werden relevante Inhalte der E-Mail sowie die Sprache des E-Mail-Textes identifiziert. Die detaillierte Beschreibung der Verarbeitung der E-Mails folgt in Kapitel 6. Der HTML-Document-Builder entfernt alle nicht notwendigen Informationen einer Webseite und unterteilt eine Webseite in mehrere HTML-Dokumente. Das genau Vorgehen wird in Kapitel 5 beschrieben. Im Folgenden werden einzelne Kompentenen der Datenerfassung aufgelistet und beschrieben.

Auğerdem werden relevante Inhalte der E-Mail sowie die Sprache des E-Mail-Textes identifiziert.

[3] https://code.google.com/p/java-libpst/ (25.10.2014)
[4] http://jsoup.org (25.10.2104)

Language Tagger

Die E-Mail-Anfragen von Studierenden werden in Englisch und Deutsch gestellt. Die Sprache der E-Mails wird mittels der Bibliothek language-detection[5] erkannt. Der Spracherkenner erkennt mit einer Genauigkeit von 99% insgesammt 49 verschiedene Sprachen.

Named Entity Tagger

Das Klassifizieren von Personen, Organisationen und Bezeichnungen im Bereich der natürlichen Sprachverarbeitung wird als Named Entitiy Recognition (NER) bezeichnet. Das Ziel von NER im Rahmen dieser Arbeit war es, Vorlesungsnamen oder sonstige relevante Bezeichnungen zu erkennen. Zwei verschiedene Ansätze wurden jeweils für den deutschen und englischen Text implementiert:

- **German Named Entity Tagger**
  In der deutschen Sprache werden Namen, Origanisationen und Bezeichnungen mit großen Anfangsbuchstaben meist hintereinander geschrieben. Dieser Sachverhalt wird ausgenutzt, um im Text die Named Entities zu erkennen. Dabei werden Phrasen, die aus einer Folge von großgeschriebenen Wörtern bestehen und ein Stopwort beinhalten, nicht berücksichtigt.

- **English Named Entity Tagger**
  In der englischen Sprache werden Namen, Origanisationen und Bezeichnungen mit Hilfe des POS Tagger identifiziert. Dabei wird eine Phrase, die aus einer Folge von Nomen besteht, als Named Entity markiert.

Stopword Tagger

Der Stopword Tagger erkennt und annotiert Stopwörter, Funktionswörter ohne semantischen Inhalt, die in der Regel nur eine grammatikalische Funktion im Satz beinhalten. Als Stopwörter werden meistens bestimmte als auch unbestimmte Artikel, Konjunktionen und häufige Präpositionen deklariert. In der Regel werden diese Wörter über eine vordefinierte Wortliste erkannt. Domainabhängige Stopwörter sind Wörter, die mit einer bestimmten Häufigkeit in einem Korpus vorkommen. Stopwörter werden im Allgemeinen am Anfang der Datenverarbeitung erkannt und in den folgenden Verarbeitungsschritten nicht mehr betrachtet.

Personal Name Tagger

Ein Personal Name Tagger funktioniert analog zum Stopword Tagger. Beim Personal Name Tagger werden persönliche Namen von Personen mittels einer vordefinierten Liste erkannt. Diese Komponente ist ein spezifischer Named Entity Tagger. Die Namen von Personen werden am Anfang der Datenerfassung erkannt und in den folgenden Schritten der Datenverarbeitung als auch Auswertung nicht mehr betrachtet.

Part of Speech (POS) Tagger

Der Prozess in dieser Komponente wird als Part Of Speech (POS) Tagging [23] bezeichnet, welches eine Zuordnung von Worten und Satzzeichen zu Wortarten unter Berücksichtigung des Kontexts herstellt. Mit dem POS Tagger wird für jedes Wort dessen Wortart bestimmt. Wortarten sind zum Beispiel Nomen, Verben, Artikel und viele mehr. Im Rahmen dieser Arbeit wird der Stanford Log-linear Part-Of-Speech Tagger[6] verwendet. Der POS Tagger wird zum einen in der Named Entity Recognition verwendet und zum anderen bei der Fragenanreicherung im Fragen-Antwort-System (Kapitel 9.2).

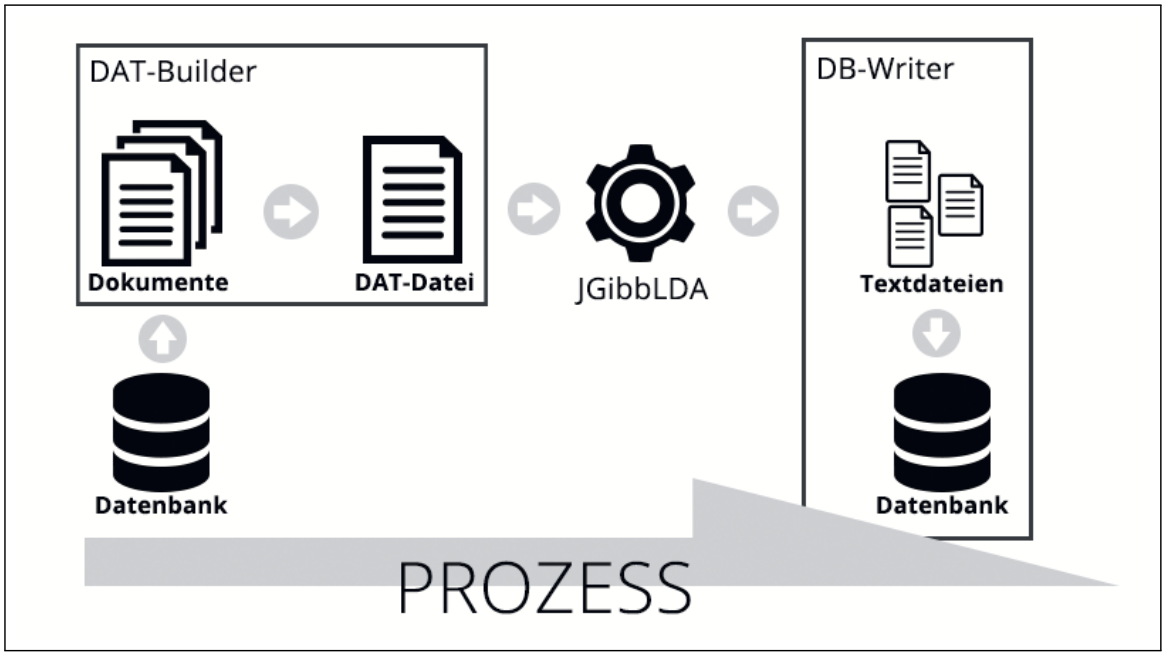

Abbildung 4.3: Ablauf in der Topic-Modell-Generierung

Topic-Modell-Generierung

Die Abbildung 4.3 visualisiert den Ablauf in der Topic-Modell-Generierung. Die generierten probabilistischen Modelle bilden die Grundlage zur Analyse der Dokumente in den Korpora und sind eine Kernkomponente im Frage-Antwort-System. Insgesamt werden drei verschiedene Topic-Modelle erzeugt: Ein E-Mail-Topic-Modell, ein Web-Topic-Modell und ein Mix-Topic-Modell. Die ersten beiden Topic-Modelle aus den E-Mails und den Webseiten werden für die Analyse der Daten benutzt. Das Mix-Topic-Modell wird im Frage-Antwort-System verwendet. Der DAT-Builder holt die relevanten Dokumente in bereinigter Form aus der Datenbank und fügt diese zu einer DAT-Datei zusammen. Die DAT-Datei muss im folgenden Format sein:

```
[M]
[document 1]
[document 2]
[document 3]
...
[document M]
```

In der ersten Zeile der Datei steht die Anzahl der Dokumente M. Ein Dokument i ist durch die einzelnen Wörter definiert:

$$[\text{document } i] = [\text{word}_{i1}]\ [\text{word}_{i2}]\ \ldots\ [\text{word}_{iN_i}]$$

Die einzelnen Wörter müssen durch ein Leerzeichen getrennt sein. Die Java Implementierung JGibbLDA[7] wurde als Umsetzung für das LDA [5] benutzt. Die Ausgabe vom JGibbLDA[5] sind folgende Textdateien:

[5] https://code.google.com/p/language-detection/ (25.10.2014)
[6] http://nlp.stanford.edu/software/tagger.shtml (05.11.2014)
[7] http://jgibblda.sourceforge.net (25.10.2014)

- `<Dateiname>.others`
  Die Datei beinhaltet die Parameter, anhand welcher das probabilistische Modell erzeugt wurde.

- `<Dateiname>.phi`
  Die Datei enthält die Topic-Wort-Verteilung. Jede Zeile ist ein *Topic* und jede Spalte ist eine Wort aus dem Vokabular. In den Zellen ist die Wahrscheinlichkeit für ein Wort, mit der es zu dem entsprechenden *Topic* gehört.

- `<Dateiname>.theta`
  Die Datei enthält die Dokument-Topic-Verteilung. Jede Zeile ist ein Dokument und jede Spalte ist ein *Topic*. In jeder Zelle ist die Wahrscheinlichkeit für ein Dokument, mit der es zu dem entsprechenden *Topic* gehört.

- `<Dateiname>.tassign`
  Die Datei beinhaltet die Topic-Zuweisung jedes Wortes innerhalb eines Dokuments und zwar für jeweils jedes Dokument. Jede Zeile ist ein Dokument. In jeder Zeile ist *Wortindex:TopicIndex* für alle Wörter des Dokuments in der jeweiligen Zeile enthalten.

- `<Dateiname>.twords`
  In der Datei stehen die wahrscheinlichsten Wörter für jedes *Topic*.

Nach der Topic-Modell-Generierung werden die Topic-Wort-Verteilungen aus der Datei *.phi* und die Dokument-Topic-Verteilungen aus der Datei *.theta* in ein relationales Datenbankmodell übernommen. Des Weiteren werden die Dokumente in der Datenbank, aus denen die DAT Datei erzeugt wurde, mit der Ausgabe in den Dateien *.phi* und *.theta* über die Datenbank verbunden. Nach diesem Schritt sind alle notwendigen Informationen zur Visualisierung der Modelle in der Datenbank enthalten, sowie das entsprechende Mix-Topic-Modell generiert, welches im Frage-Antwort-System verwendet wird.

Visualisierung

Die Visualisierung ist die Benutzerschnittstelle zur Interaktion mit dem Helpdesk `InSight`, welche aus mehreren Technologien besteht. Die Schnittstelle soll die Analyse von generierten Topic-Modellen und den Vergleich zwischen diesen ermöglichen. Das Frage-Antwort-System für die Studienberatung ist auch in dieser Schnittstelle integriert. In den folgenden Unterabschnitten wird auf die einzelnen Kompenenten der Visualisierung eingegangen.

- **Datenbank**
  Die ganzen Daten, die während dem gesamten Prozess zwischengespeichert werden, sowie die Ausgabeinformationen aus der Topic-Modell-Generierug werden in einer MySQL-Datenbank[8] gespeichert. Die Datenbank befindet sich auf dem gleichen Server wie auch die Webapplikation. Nach dem Speichern der Informationen der Ausgabe der Topic-Modell-Generieurung ist das System zur Analyse und Antwortgenerierung bereit. Es werden keine neuen Daten in die Datenbank bis zur nächsten Topic-Modell-Generierung gespeichert.

[8] http://www.mysql.de (25.10.2014)

- **Server**
  Als Server wird ein Tomcat[9] verwendet. Das ist ein Standadserver für Java Servlets als auch für Webseiten und Servers, die in Java geschrieben sind. Zur Entwicklung von Webanwendungen, die auf einem Tomcat-Server laufen, wird die Groovy/Grails Tool Suite (GGTS)[10] verwendet. Das ist eine Erweiterung der Entwicklungsumgebung Eclipse[11], welche zum Standard in der Java-Entwicklung herangewachsen ist. In GGTS kann in der Sprache Groovy entwickelt werden, welche eine Erweiterung der Sprache Java ist und im Java-Compiler kompiliert wird. Daneben bietet es fertige Modelle zur einfachen Umsetzung von Java basierender Webentwicklung. Die Integration und Erweiterung durch Bibliotheken ist in GGTS genauso leicht wie unter Eclipse-Projekten in Java. Die komplette Business-Logik der Webseiten ist im Client integriert, so dass der Server nur für die Kommunikation zwischen Client und Datenbank zuständig ist.

- **Client**
  Die Client-Seiten sind in HTML implementiert. Die Business-Logik ist in Javascript umgesetzt, damit der Server bei einer größeren Nutzerzahl mit vielen Client-Anfragen nicht überlastet wird. Als Javscript Framework zur Verwaltung der HTML-Seiten wird AngularJs[12] eingesetzt. Für die Visualisierung und Interaktion mit den dargestellten Grafiken wird das Javascript Framework Highcharts[13] verwendet.

In diesem Kapitel wurden sowohl die einzelnen Komponenten als auch die Technologie zur Datenverarbeitung und zur Topic-Modell-Generierung vorgestellt. Durch die Visualisierung eines generierten E-Mail-Topic-Modells aus dem E-Mail-Korpus und eines Web-Topic-Modells aus den Webseiten wird das Ziel verfolgt, die zwei Korpora miteinander zu vergleichen und die Webseiten anhand der gewonnenen Informationen zu optimieren. Die Webseiten sollen an die Bedürfnisse der Studierenden angepasst werden, um somit die E-Mail-Anfragen zu reduzieren. Des Weiteren wird durch das graphische User-Interface (GUI) eine Schnittstelle zum Frage-Antwort-System erstellt.

[9] http://tomcat.apache.org (25.10.214)
[10] http://spring.io/tools/ggts (25.10.2014)
[11] https://www.eclipse.org/home/index.php (25.10.2014)
[12] http://angularjs.org (25.10.2014)
[13] http://www.highcharts.com (25.10.2014)

5 HTML Datenverarbeitung

In diesem Kapitel werden die Webseiten der Studienberatung als Informationsquelle dargestellt und die Datenerfassung aus diesen im Detail erläutert. Im ersten Unterkapitel 5.1 wird allgemein die Struktur und das Ziel der Datenverarbeitung beschrieben. Im nächsten Unterkapitel 5.2 wird die Durchführung der Datenerfassung genauer erläutert.

5.1 Ausgangslage und Ziel der Datenverarbeitung

Die Studienberatung des Fachbereichs Informatik an der Technischen Universität Darmstadt betreut und pflegt Webseiten. Auf diesen Webseiten find sich zum einem FAQs (Frequently Asked Questions) wieder, wie in Abbildung 5.1 als Screenshoot[1] abgebildet. In dieser Abbildung 5.1 sind FAQs in der Kategorie Allgemein zu sehen, weitere existierende FAQ-Kategorien der Studienberatung sind Anwendungsfächer, Bachelorstudium, Internationales Studium und Masterstudium.

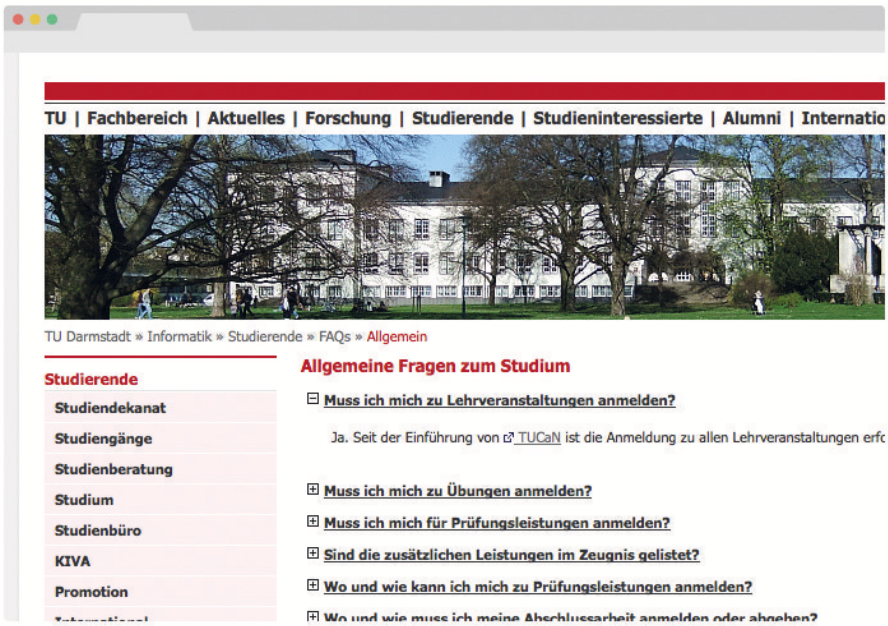

Abbildung 5.1: FAQ-Webseite der Studienberatung (Screenshoot[1])

Ein Teil des HTML-Codes ist in Abbildung 5.2 skizzenhaft abgebildet. Die einzelnen Fragen sind mit HTML-Überschrift-Syntax $h3$ umgesetzt. Sowohl die einzelnen Fragen als auch die Abschnitte lassen sich durch die Trennung der HTML-Überschrift-Syntax $h1$ bis $h7$ identifizieren. Die Frage steht innerhalb der HTML-Überschrift-Syntax, die Antwort folgt danach. Die wichtigen Informationen auf einer Webseite der Studienberatung sind immer innerhalb eines HTML-Elements div eingeschlossen, das durch die CSS-Klasse $zentrale\_spalte$ oder $content$ definiert ist.

Die vorgeschlagenen Antworten zu einer Frage im Frage-Antwort-System im Helpdesk `InSight` stammen aus Dokumenten, die durch die Webseiten der Studienberatung generiert wurden. Dies betrifft alle englischen und deutschen Webseiten, die folgende URL-Präffix haben (Webseiten, die eigentlich eine PDF sind, werden nicht betrachtet):

[1] https://www.informatik.tu-darmstadt.de/de/studierende/faqs/allgemein (24.10.2014)

```
<html>
<head>...</head>
<body>
    ...
    <div class="zentrale_spalte">
    ...
    <h3>Muss ich mich zur Lehrveranstaltung anmelden?</h3>
    <p>Ja, Seit der Einführung von <a href="...">TUCaN</a> ist die
    Anmeldung zu allen Lehrveranstaltungen erforderlich</p>
    ...
    </div>
</body>
</html>
```

Abbildung 5.2: Skizzenhafter HTML-Code-Ausschnitt der Webseite aus Abbildung 5.1

- https://www.informatik.tu-darmstadt.de/de/studierende/
- https://www.informatik.tu-darmstadt.de/de/international/
- https://www.informatik.tu-darmstadt.de/en/students/
- https://www.informatik.tu-darmstadt.de/en/international/

Alle echten Webunterseiten der eben aufgeührten URLs wurden gecrawlt und verarbeitet. Webseiten, die keine FAQs darstellen, sind in der Regel so aufgebaut wie die exemplarisch als Screenshoot präsentierte Webseite[2] in Abbildung 5.3.

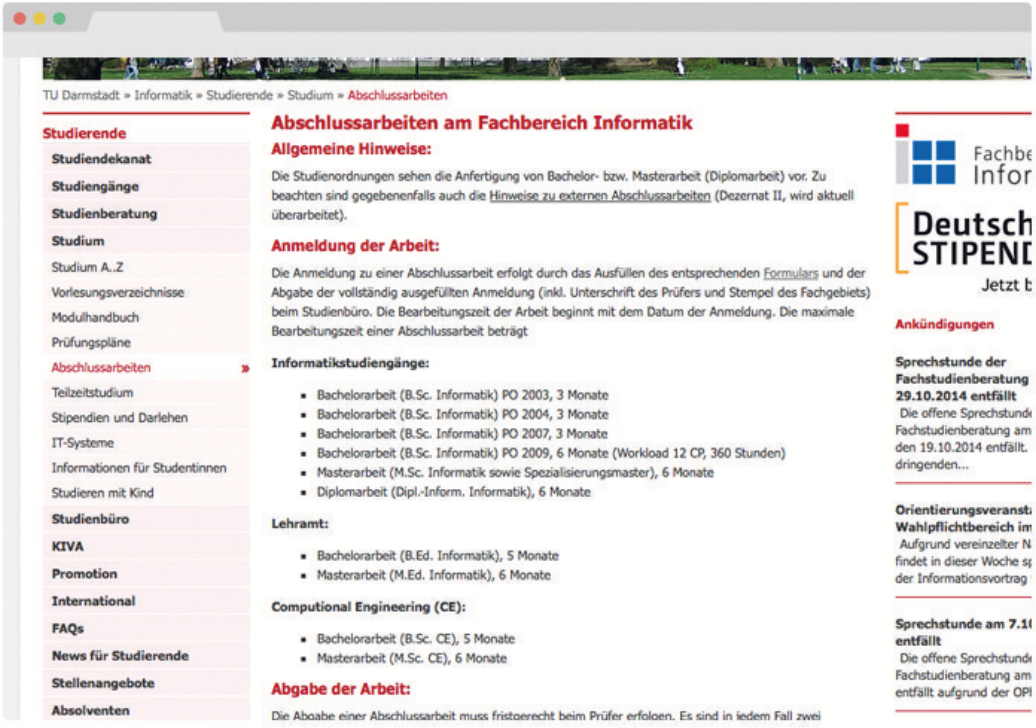

Abbildung 5.3: Eine exemplarische Webseite der Studienberatung (Screenshoot[2])

Die Webseite enthält Informationen über Abschlussarbeiten im Fachbereich Inormatik an der Technischen Universität Darmstadt. Alle Seiten sind nach dem gleichen Prinzip wie die eben dargestellte

[2] https://www.informatik.tu-darmstadt.de/de/studierende/studium/abschlussarbeiten/ (25.10.2014)

FAQ-Seite aufgebaut. Der relevante Inhalt ist nach den einzelnen Überschriften (In HTML-Syntax als *h*1 bis *h*6) auf der Webseite getrennt und thematisiert. Ein Dokument entspricht einem Textsegment, welches durch eine Überschrift separiert ist. Zum Beispiel wird das Textsegment unter der Überschrift *Allgemeine Hinweise* als ein Dokument erkannt und bekommt als Titel alle hierarchisch darüber stehenden Überschriften. In diesem Fall heißt der Titel des extrahierten Dokuments *Abschluss am Fachbereich Informatik Allgemeine Hinweise*.

5.2 Durchführung der Datenverarbeitung

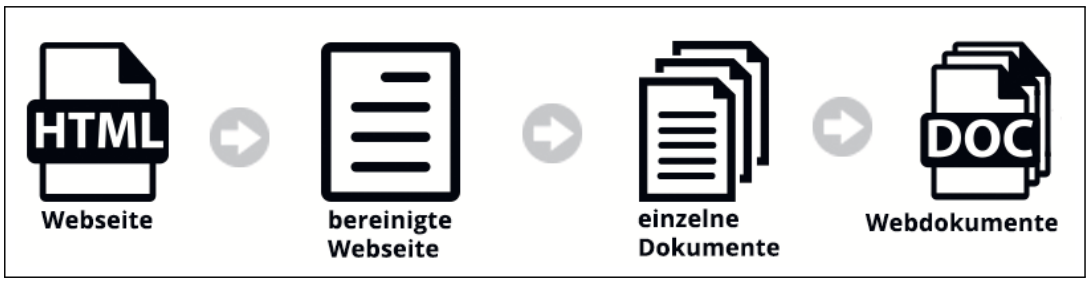

Abbildung 5.4: Einzelne Schritte in der Datenverarbeitung der Webseiten

Der Prozess in Abbildung 5.4 stellt die Ausgabe der wesentlichen drei Schritte in der Datenverarbeitung der Webseiten dar. Im ersten Schritt wird die Webseite bereinigt, damit nur die relevanten Informationen enthalten sind:

1. Der HTML-Body-Inhalt der Webseite wird extrahiert.
2. Der Inhalt innerhalb des HTML-Tags div mit dem CSS-Klassennamen $zentrale\_spalte$ oder $content$ wird aus dem Ergebnis aus Schritt 1 extrahiert.
3. Alle HTML-Kommentrare, HTML-Syntax und Javascript oder PHP-Code werden aus der Webseite entfernt. Es bleiben nur noch die einzelnen Überschriften mit der HTML-Syntax und dem restlichen Text enthalten.

Nach diesen drei Unterschritten entsteht eine bereinigte Webseite, die nur noch relevante Informationen für die Weiterverarbeitung enthält. Im Folgenden wird die bereinigte Webseite in einzelne Dokumente gegliedert. Jeder separierte Textabschnitt durch eine Überschrift in der HTML-Syntax wird zu einem Dokument mit Titel, Text und der URL, aus dem das Dokument erzeugt wurde. Wie der Titel bestimmt wird, ist in Abschnitt 5.1 beschrieben. Für jedes Dokument wird der Text des Dokuments bereinigt, welcher für die Weiterverarbeitung zur Topic-Modell-Generierung genutzt werden kann. Die einzelnen Schritte der Textbereinigung sind im Folgenden beschrieben:

1. Alle Zeilenumbrüche durch Leerzeichen und alle mehrfachen Leerzeichen durch eines ersetzen.
2. Identifizieren von Entitäten.
3. Entfernung von allen Zeichen, die nicht zu einem Wort gehören.
4. Entfernung von Stop-Wörtern durch eine allgemeine Stopwortliste.
5. Entfernung von Stop-Wörtern, die in mehr als 1/3 der Dokumente vorkommen.
6. Entfernung von URLs oder E-Mail-Adressen.
7. Entfernung von Namen durch eine Namensliste.

Nach diesen sieben Unterschritten entsteht ein bereinigtes Dokument. Zu jedem Dokument ist der Titel, der orginale Text, der bereinigte Text und die Sprache bekannt. In Tabelle 5.1 sind dies Daten noch mal aufgelistet. Die Sprache wird durch die URL einer jeden Webseite identifiziert.

Attribut	**Bedeutung**
Titel	Der Titel wird durch die einzelnen Überschriften der einzelnen Textabschnitte auf einer Webseite gewonnen.
Originaler Text	Der originale Text ist der unverarbeitete Text jedes Textabschnittes auf einer Webseite.
Url	Die Url gehört zur Webseite, aus dem das Dokument gewonnen wurde.
Bereinigter Text	Der bereinigte Text wurde aus dem originalem Text gewonnen und enthält nur relevanten Informationen für die Verarbeitung durch die Algorithmen.

Tabelle 5.1: Die Attribute eines Webdokuments

6 E-Mail Datenverarbeitung

In diesem Kapitel werden der Aufbau einer PST-Datei, das Extrahieren der E-Mails aus dieser und die Datenerfassung im Detail erläutert. Im ersten Unterkapitel 5.1 wird allgemein die Struktur und das Ziel der Datenverarbeitung beschrieben. Im nächsten Unterkapitel 5.2 wird genau die Durchführung der Datenerfassung präsentiert.

6.1 Ausgangslage und Ziel der Datenverarbeitung

Die E-Mails der Studienberatung werden aus dem E-Mail Client in eine PST-Datei extrahiert. Diese PST-Datei dient als Eingabeformat für das im Rahmen dieser Arbeit entwickelte Helpdesk `InSight`. Bei einer PST-Datei handelt es sich um einen Container für E-Mails aus dem E-Mail Client Microsoft Outlook[1] und Exchange[2]. PST steht für *Personal Storage* also für *Persönlicher Speicher*. In einer PST-Datei ist jede einzelne E-Mail mit ihren Metainformationen gespeichert. Eine Auflistung aller angebotenen und benutzten Attribute aus der PST-Datei für eine E-Mail sind in Tabelle 6.1 aufgelistet.

Attribut	**Bedeutung**
`Body`	Der Inhalt der E-Mail, die eigentliche übermittelte Nachricht.
`SenderName`	Der Name des Senders einer Nachricht.
`RecievedByName`	Die Aufzählung der Empfänger einer Nachricht durch ihre Namen.
`OriginalDisplayCc`	Die Aufzählung der Empfänger, die in Kopie standen, einer Nachricht durch ihre Namen.
`DescriptorNodeId`	Der Universally Unique Identifier (UUID) einer E-Mail, der eindeutige Schlüssel, durch den eine E- Mail eindeutig identifiziert werden kann.
`Subject`	Der Betreff der E-Mail.
`MessageDeliveryTime`	Das Datum und die genaue Zeit, wann die E-Mail übermittelt wurde.
`ReplyToId`	Falls die E-Mail eine Nachricht auf eine vorangegangene E-Mail-Konversation ist, referenziert das Attribut auf die vorangegangene E-Mail in der PST Datei.

Tabelle 6.1: Angebotene und benutzte Informationen aus der PST-Datei bezogen auf eine E-Mail

Der `Body` und das `Subject` sind der relevante Inhalt einer E-Mail zur Weiterverarbeitung. Um einen zeitlichen Verlauf der Topics darstellen zu können ist die `MessageDeliveryTime` notwendig. Zur genau-

[1] http://www.microsoft.com/de-de/outlook-com/ (27.10.2014)
[2] http://office.microsoft.com/de-de/exchange/ (27.10.2014)

en Identifikation einer E-Mail dient die `DescriptorNodeId`. Durch die `ReplyId` werden referenzierte E-Mails identifiziert und entfernt, damit keine Duplikate im E-Mail-Korpus vorhanden sind. Die Attribute `SenderName` und `RecievedByName` dienen dazu E-Mails auszuschließen, die keine Studierendenanfragen sind. Dies betrifft allgemeine Rund-E-Mails aus E-Mail-Verteilern. E-Mails mit nur informellem Inhalt, deren Absender meistens mit *noreplay@...* gekennzeichnet ist, werden nicht berücksichtigt. Weitere Sonderaktionen, die über E-Mails kommuniziert werden, wurden auch nicht beachtet. Eine Sonderaktion ist zum Beispiel die Bücheraktion am Fachbereich Informatik an der Technischen Universität Darmstadt, die immer mit einer gesonderten dafür speziellen E-Mail-Adresse mitgeteilt wird. Aus den einzelnen E-Mail-Konversationen, die durch Studierendenanfragen entstehen, sollen einzelne Dokumente entstehen. Ein exemplarisches Gesprächsmodell zwischen einem Studierenden und der Studienberatung ist in Abbildung 6.1 zu sehen.

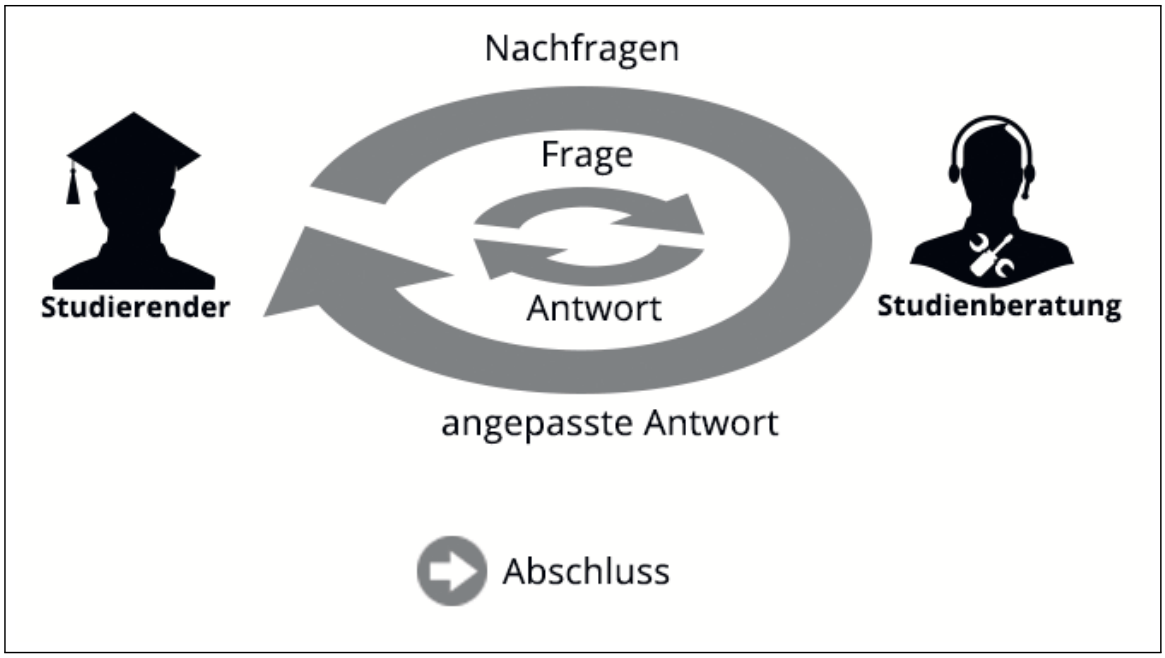

Abbildung 6.1: Exemplarisches Gesprächsmodell zwischen einem Studierenden und der Studienberatung

Stellt ein Studierender eine Frage an die Studienberatung, bekommt dieser in der Regel eine Antwort. Es gibt mehrere Möglichkeiten, wie diese Kommunikation fortgesetzt werden könnte:

- Der Studierende stellt eine neue Frage, welche wiederum von der Studienberatung beantwortet wird.
- Der Studierende hat die Antwort nicht verstanden und fragt noch einmal nach. Die Studienberatung passt die Antwort an das Verständnis des Studierenden an.
- Die zwei eben beschreibenen Sachverhalte, der Studierende fragt nochmal nach oder stellt eine neue Frage, können sich in beliebiger Reihenfolge wiederholen.

Die Kommunikation endet entweder nach einer Antwort der Studienberatung oder der Studierende bedankt sich noch einmal bei der Studienberatung, worauf gegebenenfalls die Studienberatung noch ein einziges mal auf die Danksagung reagiert, indem diese zum Beispiel schreibt: *Kein Problem*. Jede Kommunikation zwischen einem Studierenden und der Studienberatung, die durch das Verlinkungssystem der PST-Datei identifiziert werden kann, wird als ein E-Mail-Dokument aufgefasst. Ein E-Mail-Dokument kann verschiedene Kommunikationverläufe, wie eben gezeigt, enthalten.

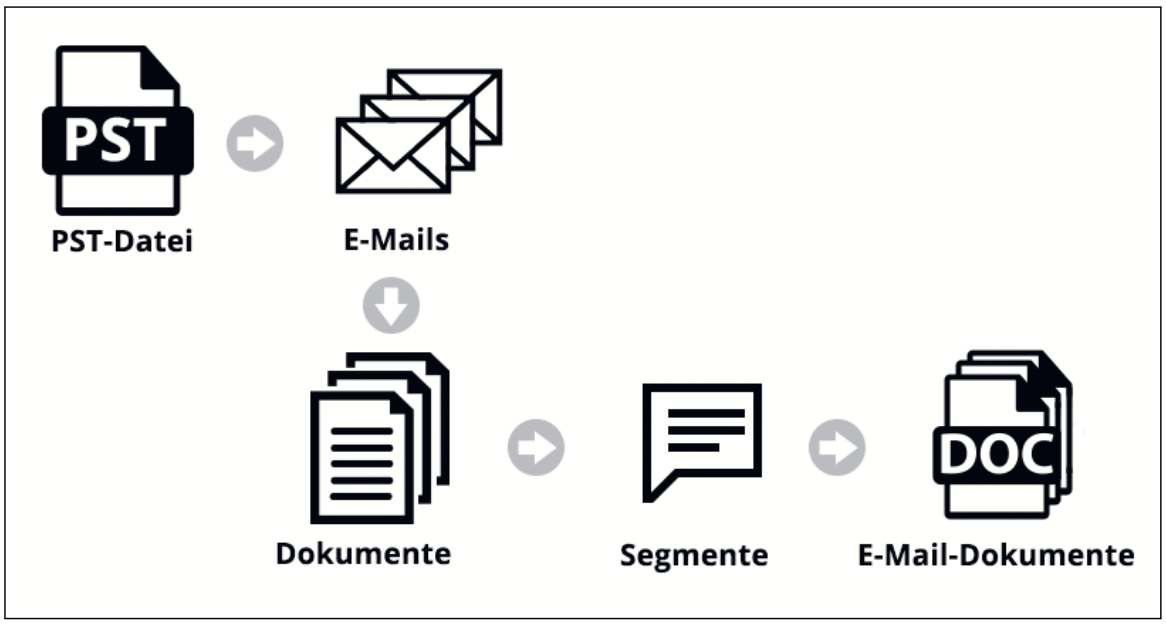

Abbildung 6.2: Einzelne Schritte in der Datenverarbeitung der E-Mails

Der Prozess in Abbildung 6.2 stellt die wesentlichen Schritte und deren Ausgabe in der Datenverarbeitung der E-Mails dar. Die vier Hauptschritte werden im Folgenden aufgelistet und detailliert dargestellt:

1. Im ersten Schritt wird die PST-Datei gecrawlt und jede einzelne E-Mail wird mit den Attributen aus Tabelle 6.1 zwischengespeichert. Dabei werden die E-Mails, die keine Studierendenanfragen sind, welche im ersten Abschnitt 6.1 beschrieben wurden, nicht verarbeitet.

2. Im zweiten Schritt werden aus den E-Mails Dokumente erzeugt und Duplikate gelöscht. Duplikate entstehen, indem man auf eine E-Mail antwortet. Dabei wird die ursprüngliche E-Mail separat behalten und als Anhang zur neuen E-Mail hinzugefügt. Alle E-Mails, auf die durch die `ReplyId` einer anderen E-Mail referenziert wird, sind beantwortete E-Mails, also Duplikate, die gelöscht werden müssen.

3. Im dritten Schritt sind Dokumente mit jeweils einem E-Mail-Verlauf enthalten. Ein E-Mail-Verlauf besteht aus einigen semantisch unwichtigen Informationen. Semantisch unwichtige Informationen sind zum Beispiel die Anrede, die Grußformel und die Signaturen sowie die Disclaimer. Jedes Dokument wird in Abschnitte (Segmente) unterteilt. Die Segmentierungsschritte sind in den folgenden Unterpunkten beschrieben:

   a) Identifizierung von Named Entities im Dokument.

   b) Entfernung von Zitatzeichen (>) im Dokument.

   c) Das Dokument wird in Segmente unterteilt. Die Segmente werden erkannt, indem diese durch zwei Zeilenumbrüche getrennt sind.

   d) Die Anrede wird gesondert segmentiert. Ein Segment wird in Zeilen unterteilt und für jede Zeile wird überprüft, ob diese mit Anredewörtern anfängt gefolgt von einem Interpunktionszeichen.

   Für jedes Segment wird die Sprache identifiziert. Das Dokument kriegt die Sprache zugewiesen, welche die Mehrheit durch die Segmente hat. Relevante Segmente für die Weiterverarbeitung werden markiert. Die unrelevanten Segmente werden durch Regeln identifiziert. Die Anrede, die

Grußformel, die Singnaturen als auch die Disclaimer enthalten bestimmte Sonderzeichen, Wort-Zeichen-Muster als auch Wörter, die jeweils typisch für diese Abschnitte in der E-Mail sind.

4. Im letzten Schritt werden die relevanten Segmente eines E-Mail-Dokuments zusammengefügt und einige weitere Verarbeitungsschritte auf diesem Text durchgeührt:
   a) Alle Zeilenumbrüche durch Leerzeichen und alle mehrfachen Leerzeichen durch eines ersetzen.
   b) Entfernung von allen Zeichen, die nicht zu einem Wort gehöhren.
   c) Entfernung von Stop-Wörtern durch eine allgemeine Stopwortliste.
   d) Entfernung von Stop-Wörtern, die in mehr als 1/3 der Dokumente vorkommen.
   e) Entfernung von URLs oder E-Mail-Adressen.
   f) Entfernung von Namen durch eine Namensliste.

Nach dem Prozess der Verarbeitung entstehen E-Mail-Dokumente mit den Attributen, die in der Tabelle 6.2 aufgeführt sind.

Attribut	**Bedeutung**
Betreff	Der Betreff der E-Mail.
Originaler Text	Der komplette Text im E-Mail-Verlauf im rohen Zustand.
Absender	Der Absender der E-Mail.
Empfänger	Die Liste der Empfänger der E-Mail.
Kopie	Die Empfänger der E-Mail, die in Kopie waren.
Datum	Das Datum, an dem die E-Mail im Postfach eingetroffen ist.
UUID	Ein eindeutiger Schlüssel für das E-Mail-Dokument.
Sprache	Die Sprache des E-Mail-Verlaufs.
Bereinigter Text	Der Text für die Weiterverarbeitung nach dem Verarbeitungsprozess.
Segmente	Alle Segmente eines E-Mail-Dokuments mit den Informationen: Sprache, Text, Relevanz und Position (an welcher Position das Segment im Dokument eingeordnet wird).

Tabelle 6.2: Die Attribute eines E-Mail-Dokuments

7 Skip-N-Gramme

Die Generierung probabilistischer Topic-Modelle durch das LDA [5] erzeugt keine hierarchische Struktur aus den Dokumenten eines Korpus. Für eine fest vorgegebenen Anzahl an *Topics* generiert das LDA eine Dokument-Topic-Verteilung. Des Weiteren wird eine Topic-Wort-Verteilung über das fixe Vokabular des Korpus generiert. Die signifikantesten *w* Wörter aus einem Topic bilden jeweils den Topic-Namen. Durch die Dokument-Topic-Verteilung werden die Dokumente mit unterschiedlicher Wahrscheinlichkeit den einzelnen *Topics* zugeordnet und lassen sich anhand dieser Topics analysieren. In vielen Fällen sind die Topic-Namen zu generell, wie zum Beispiel das *Topic* {*anmelden, Prüfungen, Prüfung*} aus Abbildung 8.2. Der Topic-Name lässt vermuten, dass sich die Dokumente um Fragen handeln, die sich mit Prüfungsameldungen beschäftigen. Die Dokumente behandeln nicht alle die gleiche Anmeldung oder Prüfung in diesem *Topic*. Um die Daten einen Schritt weiter zu analysieren, müssten zu einem *Topic* weitere Informationen aufgelistet sein, die Aufschluss über die verschiedenen Untertehmen in einem *Topic* geben. Ein weiterer Topic-Name, welcher noch weniger Aufschluss gibt, ist das *Topic* {*Veranstaltung, Vorlesung, Einführung*} aus Abbildung 8.2. Es können nur Vermutungen angestellt werden, welche genauen Sachverhalte das *Topic* abdeckt. Es lässt sich vermuten, dass es sich um einige Einführungsveranstaltungen handelt, aber was genau das Problem ist oder welche Fragen aufkommen ist nicht ersichtlich. Um die aufgeführten Problematiken des generierten probabilistischen Topic-Modells durch das LDA zu lösen, wurden die einzelnen Dokumente innerhalb eines *Topics* mit einem weiteren Verfahren durch Skip-N-Gramme analysiert und gegliedert. Die Erstellung von Skip-N-Grammen durch die relevantesten Dokumente innerhalb eines *Topics* ermöglicht es die Dokumente innerhalb eines *Topics* genauer zu analysieren und mehr Wissen zu extrahieren als auch einen naiven Ansatz zur Ersetzung des Topic-Namens durch eine Menge von relevanten K-Skip-Grammen aus dem jeweilgen *Topic*. Mit den erstellten Skip-N-Grammen gewinnen einzelne Topics eine weitere hierachische Ebene, die Dokumente innerhalb eines *Topics* werden anhand der Skip-N-Gramme gruppiert. Im ersten Unterkapitel 7.1 werden die K-Skip-N-Gramme beschrieben, auf die die Skip-N-Gramme basieren. Im zweiten Unterkapitel 7.2 wird das Verfahren zur Erstellung der Skip-N-Gramme beschrieben und wie die Skip-N-Gramme in das Helpdesk `InSight` integriert werden.

7.1 K-Skip-N-Gramme

Die Skip-N-Gramme basieren auf K-Skip-N-Grammen [22], die auf der Idee von N-Grammen [24] entstanden sind. In diesem Unterkapitel werden die K-Skip-N-Gramme als auch N-Gramme beschrieben.
Mit N-Grammen (Unigramme, Bigramme, Trigramme, etc.) kann ein vorliegender Text in Fragmente zerlegt werden. Ist zum Beispiel der Satz in 7.1 gegeben, entstehen Unigramme, Bigramme und Trigramme wie in Tabelle 7.1 dargestellt.

"I saw the man with the telescope" (7.1)

Bei den N-Grammen bezeichnet das N in unserem Fall die Anzahl der Wörter, die in ein Fragment aufgenommen werden. Eine Erweiterung der N-Gramme sind die K-Skip-N-Gramme. Die Zahl K gibt an, wieviele Wörter maximal bei der Konstruktion der N-Gramme übersprungen werden dürfen. Bei einem $K = 3$ dürfen zwischen den einzelnen Wörtern des N-Gramms bis zu drei Wörter dazwischen liegen. Zum Satz 7.1 sind die möglichen 2-Skip-Bigramme und 2-Skip-Trigramme in Tabelle 7.2 aufgelistet. Die Formel für die K-Skip-N-Gramme ist in Gleichung 7.2 beschreiben.

$$\{w_{i_1}, w_{i_2}, w_{i_3}, ..., w_{i_n} | \sum_{j=1}^{n} i_j - i_{j-1} < k\} \tag{7.2}$$

N-Gramm	Menge
Unigramme	{ I, saw, the, man, with, the , telescope }
Bigramme	{ I saw, saw the, the man, man with, with the , the telescope }
Trigramme	{ I saw the, saw the man, the man with, man with the, with the telescope }

Tabelle 7.1: N-Gramme aus dem Satz 7.1

K-Skip-N-Gramm	Menge
2-Skip-Bigramme	{ I saw, I the, I man, saw the, saw man, saw with, the man, the with, the the, man with, man the, man telescope, with the, with telescope, the telescope}
2-Skip-Trigramme	{ I saw the, I the man, I man with, I man the, I man telescope, I saw man, I saw with, I the with, I the the, saw the man, saw man with, saw with the, saw with telescope, saw the with, saw the the, saw man the, saw man telescope, the man with, the with the, the the telescope, the with telescope, the man the, the man telescope, man with the, man the telescope, man with telescope, with the telescope }

Tabelle 7.2: 2-Skip-N-Gramme aus dem Satz 7.1

7.2 Skip-N-Gramme in InSight

Im vorherigen Unterkapitel wurden die K-Skip-N-Gramme [22] beschrieben. Bei einem Satz mit sieben Wörtern wie in 7.1 entstehen eine Menge 2-Skip-Trigramme, wie in Tabelle 7.2 zu sehen ist. Ein weiteres Problem neben der großen Menge an 3-Skip-Trigrammen ist die Relevanz des Inhalts dieser 3-Skip-Trigramme. Nicht jedes 3-Skip-Trigramm gibt einen sinnvollen Aufschluss über den Inhalt des gegebenen Satzes, ein Beispiel hierfür ist das 2-Skip-Trigramm aus Tabelle 7.2: *the with the*. Das sind zwei Probleme der K-Skip-N-Gramme, die deren Anwendung auf längere Texte erschweren. Eine mögliche Lösung bieten Skip-N-Gramme, die eine Erweiterung der K-Skip-N-Gramme sind. Die Skip-N-Gramme haben keine Eingränzung K, wie viele Wörter (Tokens) maximal übersprungen werden dürfen. Um die große Anzahl an Skip-N-Grammen zu unterbinden, die durch ein maximal großes K entstehen würden, wird das Skip-N-Gramm aus den häufigsten Skip-(N-1)-Grammen erzeugt. Das N gibt die Anzahl der Länge eines Fragments an.
Im ersten Unterabschnitt wird der Generierungsprozess von Skip-N-Grammen dargestellt und im zweiten Unterabschnitt folgt die Anwendung und Integration in das im Rahmen dierser Arbeit entwickelte Helpdesk `InSight`.

Generierung von Skip-N-Grammen

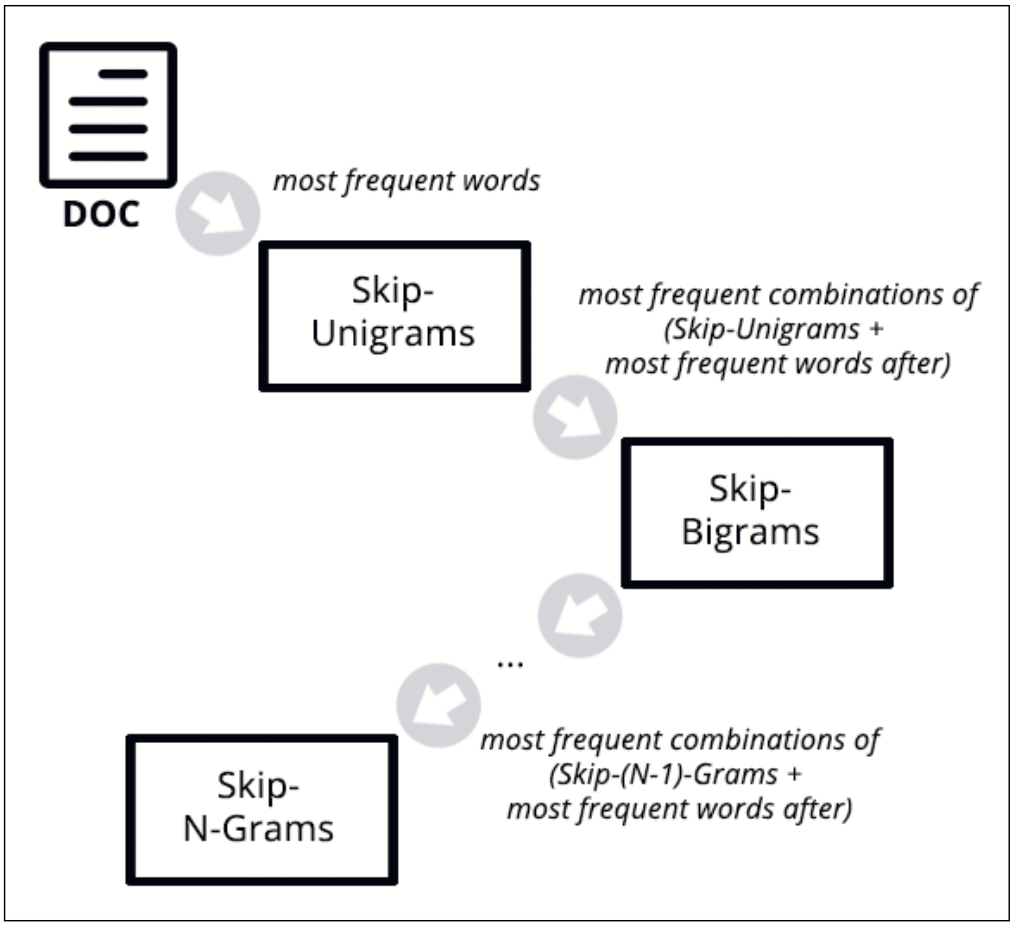

Abbildung 7.1: Generierungsprozess von N-Key-Phrases

Der Generierungsprozess von Skip-N-Grammen wird in Abbildung 7.1 visuell dargestellt. Die Skip-Unigramme sind die häufigsten n Wörter innerhalb eines Dokuments. Jetzt werden K-Skip-N-Gramme [22] gebildet. Wobei K maximal ist, so dass es alle Wörter innerhalb eines Dokuments aufspannt und N ist 2. Das erste Wort in diesen Skip-N-Grammen muss aus der Menge der Skip-Unigramme sein. Dadurch wird die Menge der möglichen Skip-Gramme eingegrenzt und die Gramme beinhalten nur relevante Wörter bezogen auf ihre Frequenz. Aus den Top k Kombinationen aus Skip-Unigrammen und einem Wort, welches im Dokument nach einem jeweiligen Skip-Unigramm vorkommt, werden die Skip-2-Gramme gebildet. Dieser Prozess wird analog bis zum einen gewählten N fortgesetzt.

Integration in InSight

Die einzelnen Schritte in der Skip-N-Gramm-Generierung sind in Abbildung 7.2 dargestellt. Für die Erstellung der Skip-3-Gramme werden die bereinigten Texte der E-Mail-Dokumente benutzt. Es werden Skip-3-Gramme, wie im vorherigen Unterkapitel 7.1 beschrieben, erstellt. Dabei werden in jedem Schritt jeweils die Skip-N-Gramme für den Folgeschritt ausgewählt, wenn deren Häufigkeit im Korpus mehr als 100 aufweist. Im letzten Schritt wird die Zuordnung der Skip-3-Gramme den Dokumenten, in denen sie vorkommen, zugeordnet und in die relationale Datenbank gespeichert.

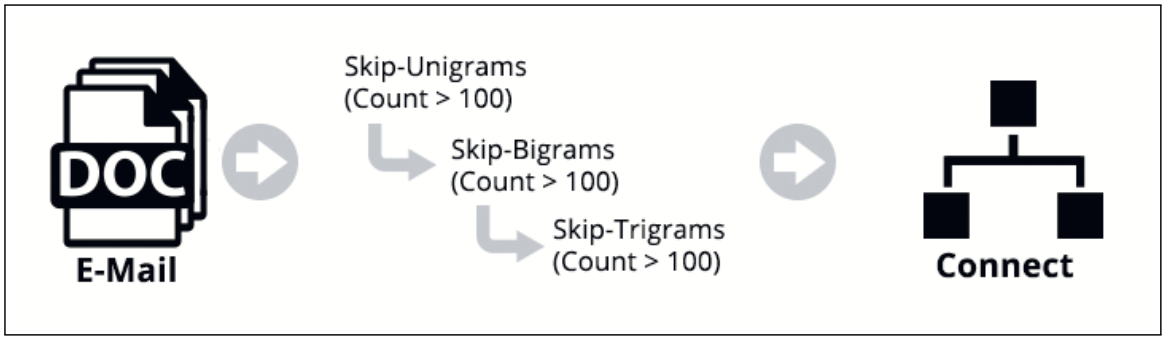

Abbildung 7.2: Generierungsprozess von 3-Key-Phrases in Insight

Der beschrieben Prozess in Abbildung 7.2 wird für jedes *Topic* und dessen relevantesten Dokumente durchgeführt. Dabei ensteht eine weitere Hierarchieebene zum gegebenen Topic-Modell, wie in Abbildung 7.3 skizziert. In Abbildung 8.5 sind die 10 relevantesten Skip-Trigramme für das *Topic* {*offenen, Sprechstunde, vereinbaren*} dargestellt. Durch die aufgelisteten Skip-Trigramme können die Dokumente innerhalb des *Topics* in weitere Segmente unterteilt werden.

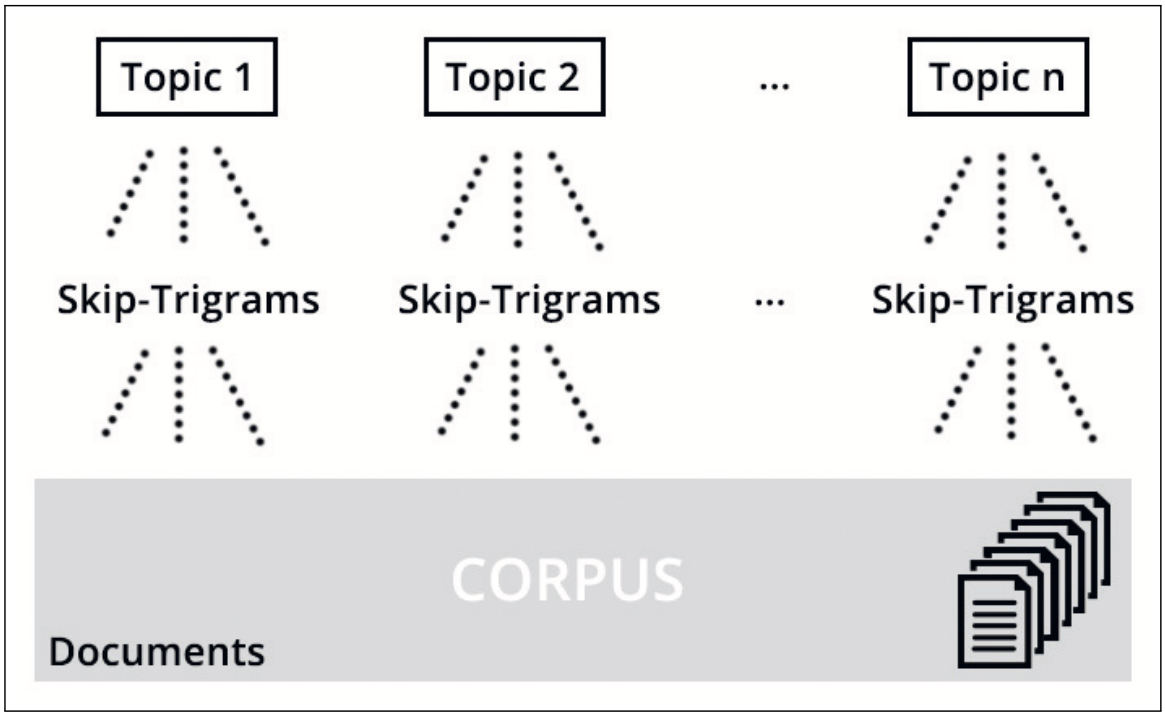

Abbildung 7.3: Skip-Trigramme als weitere Ebene zu *Topics*

8 Interaktion und Wissensextraktion mit InSight

In diesem Kapitel wird das User-Interface und die Interaktionsmöglichkeiten mit dem Helpdesk `InSight` präsentiert, das im Rahmen dieser Arbeit entwickelt wurde. Die Software wurde als eine gemeinsam nutzbare Webapplikation entwickelt. In der Abbildung 8.1 ist die Startseite mit dem Menü der Software dargestellt. Mit dem System können die einzelnen Datensätze über die Zeit verteilt analysiert werden. Der Menüpunkt *GERMAN EMAILS* leitet zum Datensatz mit den E-Mail-Anfragen weiter. Das Analysieren und Visualisieren der einzelnen Datensätze wird im ersten Unterkapitel 8.1 beschrieben. Entsprechend können die Datensätze der englischsprachigen E-Mails und die extrahierten Webdokumente in beiden Sprachen auf die gleiche Art und Weise wie die deutschsprachigen E-Mails durchforscht und so miteinander verglichen werden. Durch den vergleich können Themen erkannt werden, die eventuell in den E-Mail-Anfragen aufkommen, aber zum Beispiel in den FAQs auf den Webseiten nicht abgedeckt werden. Die Darstellung einzelner *Topics* (Themen) innerhalb eines Datensatzes wird im Unterkapitel 8.2 präsentiert. Die FAQ-Seiten, die unter dem Menüpunkten *GERMAN FAQ* und *ENGLISH FAQ* zu finden sind, werden im Unterkapitel 8.3 dargestellt. Das sind Seiten, die als interaktiver Merkzettel dienen, um relevante und häufig gestellte E-Mail-Anfragen zu speichern und die entsprechende Frage und Antwort zu dieser E-Mail passend zu formulieren. Die Benutzerschnittstelle des Frage-Antwort-Systems wird im Unterkapitel 8.4 beschrieben, zu finden unter dem Menüpunkt *QUESTION ANSWERING*.

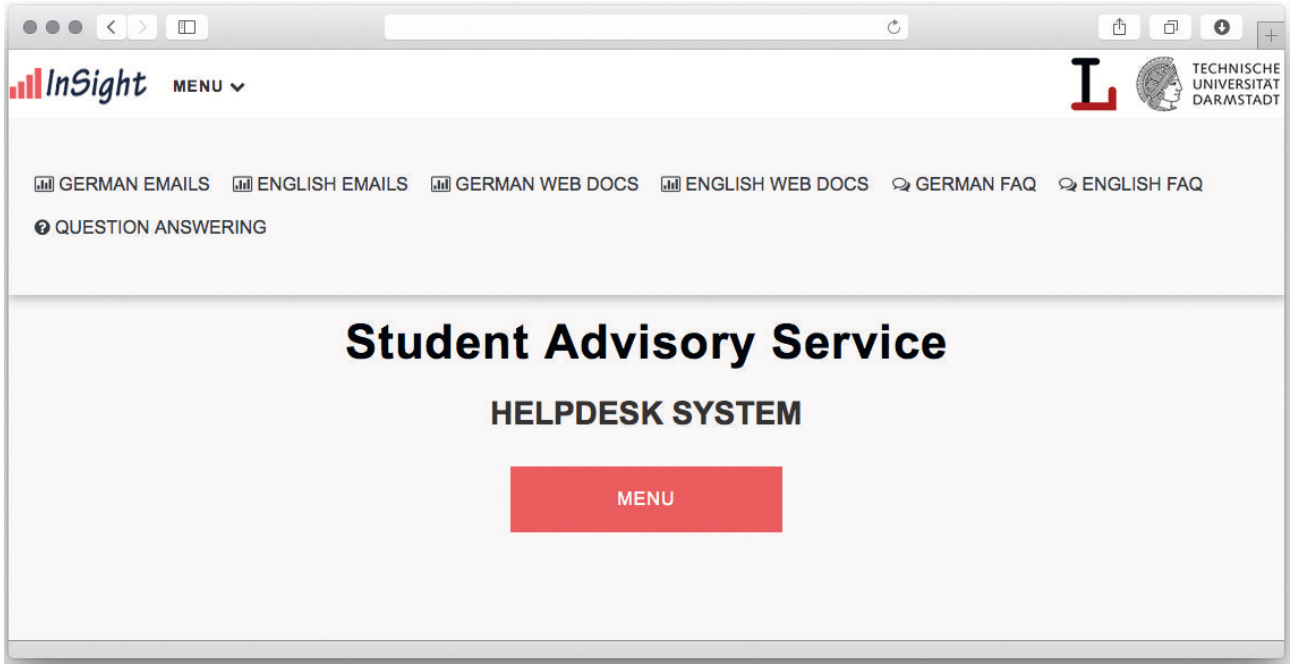

Abbildung 8.1: Das Menü im Helpdesk `InSight`

8.1 Datensatzvisualisierung

In diesem Unterkapitel wird die Visualisierung der Datensätze präsentiert. In Abbildung 8.2 ist der Datensatz mit den deutschsprachigen E-Mail-Anfragen dargestellt. Im linken Teil der Webseite im Screenshot in der Abbildung 8.2 ist die Verteilung der *Topics* visualisiert. In den folgenden Abschnitten werden die einzelnen Abschnitte der Webseite im Detail beschrieben.

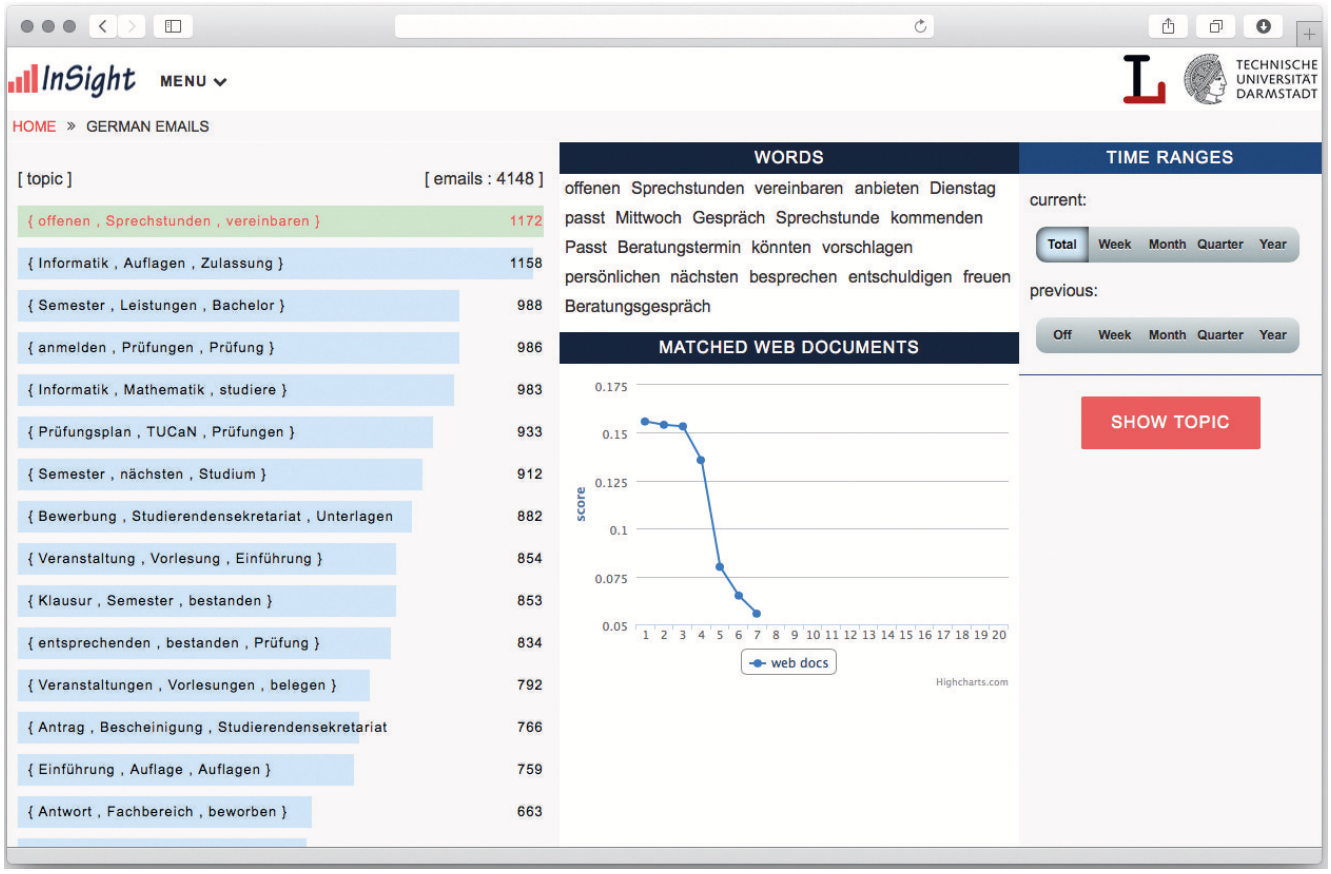

Abbildung 8.2: Datensatzvisualisierung in `InSight`

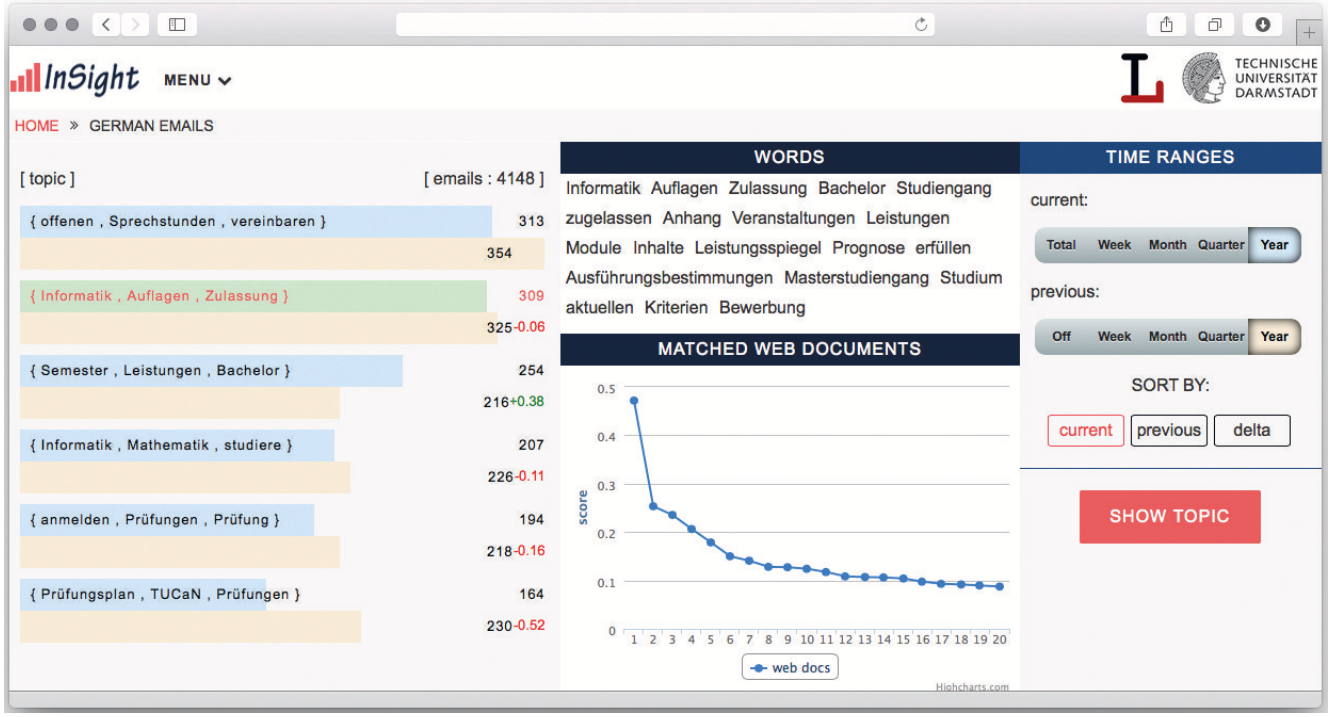

Abbildung 8.3: Zeitverlaufvergleich innerhalb eines Datensatzes bezüglich der Themen in `InSight`

Topics-Darstellung

Die Darstellung der Topic-Namen durch die signifikantesten Wörter für jeweils ein *Topic* und den Anteil an zugeordneten Dokumenten durch die Balken ist an die Visualisierung von Topic-Modellen aus [25] angelehnt. Insgesamt sind 4148 identifizierte E-Mail-Anfragen aus den Jahren 2010 bis 2013 im System enthalten. Nach der Topic-Model-Generierung ist eine Topic-Dokument-Verteilung zwischen jedem Topic und Dokument vorhanden. Jedes Dokument ist mit einer Wahrscheinlichkeit einem *Topic* zugeordnet. Um die Dokumente zwischen den *Topics* zu trennen, wird ein Threshold mit 0.015 festgelegt. Nur Dokumente, die über dem Threshold mit ihrer Wahrscheinlichkeit zum *Topic* liegen, werden als Dokumente des jeweiligen *Topics* angesehen. Eine klare Trennung der Dokumente zu den einzelnen *Topics* wurde nicht vorgenommen. Deswegen liegt die Aufsummierung der Dokumente aus den einzelnen *Topics* deutlich über 4148 E-Mail-Dokumenten. Für das LDA-Verfahren [5] wurde $n = 100$ als Parameter definiert. Das n legt die Topic-Anzahl fest. Bei der Wahl und Analyse der Topic-Anzahl ist folgender Sachverhalt aufgefallen: *Die Anzahl der Topics ändert nur marginal und nicht von Bedeutung an der Topic-Wort-Verteilung der Top-Topics*. Die Balken visualisieren optisch die Dokumentverteilung der einzelnen Topics. Beim Klicken auf einen Topic-Namen werden die relevantesten 20 Wörter, die ein *Topic* beschreiben und aus der generierten Topic-Wort-Verteilung stammen, angezeigt. Das aktuelle angezeigte *Topic* {*offenen, Sprechstunden, vereinbaren*} ist mit einem grünen Balken hinterlegt und roter Schrift gekennzeichnet.

Topic-Matching mit Webdokumenten

Das Liniendiagramm in der Mitte in der Abbildung 8.2 zeigt die Zuordnung der Webdokumente absteigend geordnet nach ihrer Relevanz zum jeweiligen *Topic* an. Hierbei wurde das E-Mail-Topic-Model verwendet und die Webdokumente wurden über das E-Mail-Topic-Model inferiert. Die Inference der Dokumente wurde in Anlehnung an [26] durchgeführt. Die Inference beim LDA [5] erzeugt in jedem Inference-Schritt während der Inference eine Topic-Wort-Zuordnung für jedes bekannte Wort im Dokument. Diese Zuordnung der Topic-Wort-Verteilung über alle Inference-Schritte wird getrackt. Anhand dieser Verteilung wird das Dokument einem *Topic* zugeordnet. Die detaillierte Beschreibung des Inference-Prozesses ist in Kapitel 9 zu finden. Diese Inference führt dazu, dass nicht jedes Dokument zu einem *Topic* und nicht jedes *Topic* zu einem Dokument zugeordnet wird, wie es der Fall nach der vorgeschlagenen Inference von [5] wäre. Somit sind dem *Topic* {*offenen, Sprechstunden, vereinbaren*} nur 7 Webdokumente von insgesamt 243 vorhandenen deutschsprachigen Webdokumenten durch die Inference zugeordnet. Der Verlauf dieser 7 Webdokumente deutet an, dass das Topic auf den Webseiten relativ zentral auf einigen wenigen Webseiten zu finden ist. Dies sollte das Ziel der Informations- und Themenverwaltung für die Support-Webseiten sein, damit die Studierenden nicht durch mehrere Seiten navigieren müssen, um die gewünschten Informationen zu finden, dadurch überfordert sind und gar nicht die gewünschte Information finden. Die genaue Zuordnung von einem *Topic* zu Webdokumenten und zu Webseiten ist in der Topic-Ansicht zu sehen, welche im folgenden Unterkapitel 8.2 beschrieben wird. Im Vergleich dazu ist in Abblildung 8.3 die Zuordnung von Webdokumenten zum *Topic* {*Informatik, Auflagen, Zulassung*} abgebildet. In diesem Fall ist zwar ein Webdokument ziemlich passend zum *Topic*, aber die Informationen zu diesem Thema sind in sehr vielen Dokumenten verteilt. Hier sollte weiter durchforscht werden, welche Dokumente aus welchen Webseiten genau zum jeweiligen *Topic* zugeordnet sind, um eventuell eine Reorganisation der Webseiten in Bezug auf das *Topic* durchzuführen.

Zeitverlaufanalyse

Eine weitere Möglichkeit ist es die *Topics* der E-Mail-Anfragen in Bezug auf deren Verlauf über die Zeit zu analysieren und die Unterschiede festzustellen. In Abbildung 8.3 ist die Verteilung der Themen im aktuellen Jahr im Vergleich zum Vorjahr dargestellt. Die *Topics* sind absteigend nach deren Häufigkeit im aktuellen Jahr sortiert. Weitere Sortiermöglichkeiten sind nach dem letzten Jahr oder nach der Differenz zwischen den beiden Jahren bezogen auf den prozentualen Anteil des *Topics* zur Gesamtmenge. Zum Beispiel ist der prozentuale Anteil von *Topic x* im vorherigen Jahr 10% im Vergleich mit den anderen *Topics* im letzten Jahr und im aktuellem Jahr 15%. So ist das *Topic* um +5% zum vorherigen Jahr ge-

stiegen. In der Ansicht wäre dies eine grüne Zahl mit dem Wert von 0.05. Für jedes Dokument ist das Eingangsdatum gespeichert. Anhand des Datums und dem gesetzten Threshold ist eine Zuordnung der Dokumente zu den *Topics* über die Zeit möglich.

8.2 Topic-Visualisierung

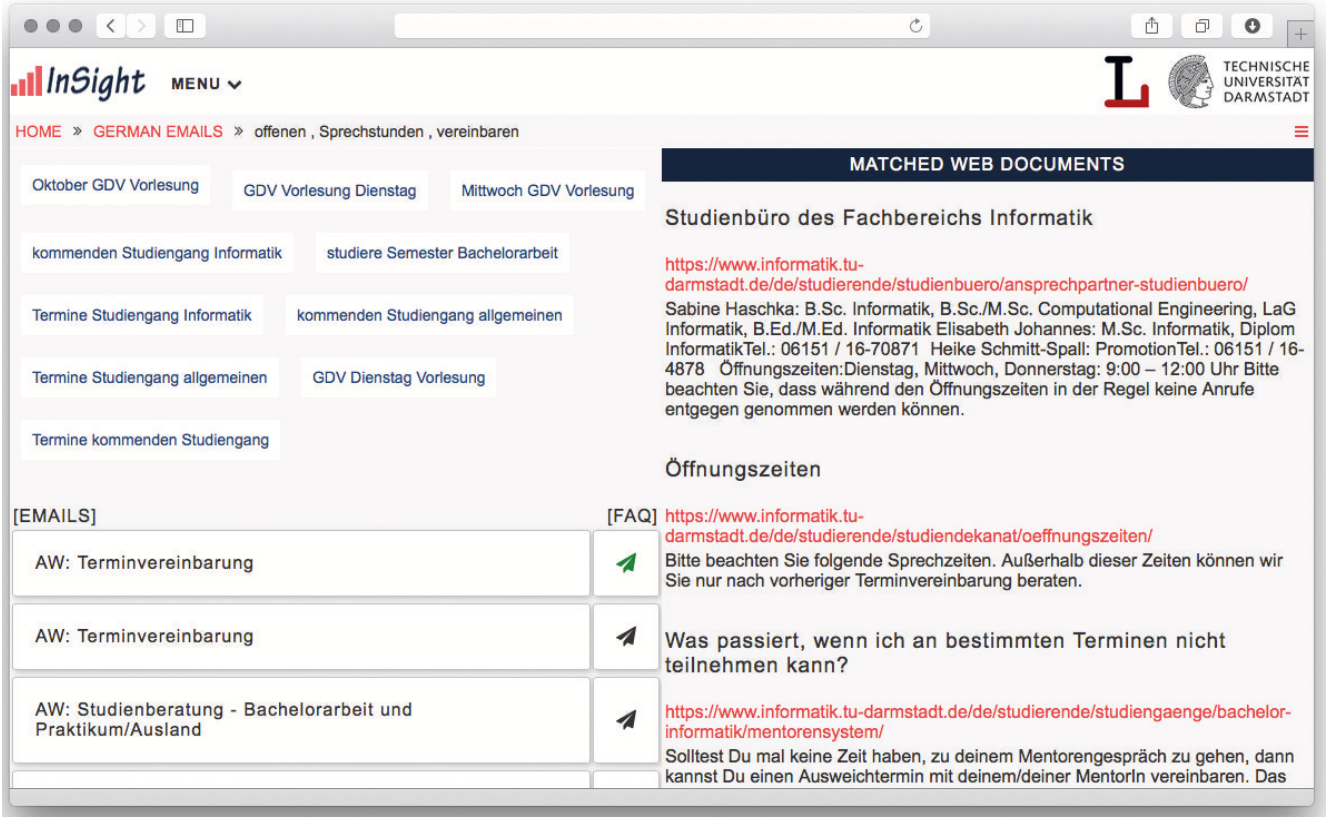

Abbildung 8.4: Topic-Darstellung im Helpdesk `InSight`

Wird in Abbildung 8.2 auf den Button *SHOW TOPIC* geklickt, öffnet sich die Topic-Ansicht in Abbildung 8.5, welches das *Topic* {*offenen, Sprechstunden, vereinbaren*} darstellt. Anhand der Skip-Trigramme, die in blauer Schrift mit weißem Hintergrund hinterlegt sind, lassen sich die Dokumente in einer weiteren Ebene eingrenzen. In der Abbildung sind jeweils die Betreffe der E-Mails zu sehen. Wird auf einen Betreff geklickt, öffnet sich der komplette E-Mail-Verlauf. Wird eine E-Mail als repräsentativ für das aktuelle *Topic* oder als relativ häufige Anfrage durch den Benutzer identifiziert, kann diese E-Mail durch einen Klick auf das Icon mit dem Papierflieger in einen Merkzettel verschoben werden. Dieser Merkzettel ist unter dem Menüpunkt *GERMAN FAQ* zu finden, welcher in Kapitel 8.3 dargestellt wird. Falls sich eine E-Mail in der FAQ-Liste befindet, ist das Icon mit einer grünen Farbe hinterlegt. In der zweiten Hälfte der Topic-Ansicht sind die passendsten Webdokumente zum Topic aufgelistet. Zu einem Webdokument wird der Inhalt, der Titel als auch die URL der Webseite angezeigt, aus der das Webdokument gewonnen wurde. Sind zum Beispiel die passendsten Webdokumente aus verschiedenen Webseiten generiert und jedes Webdokument ist hoch relevant für das *Topic*, ist das ein Indiz dafür, dass die Information auf den Webseiten zu sehr verteilt sind. Eine Reorganisation des Topics auf den Webseiten wäre empfehlenswert.

8.3 FAQ-Listen-Management

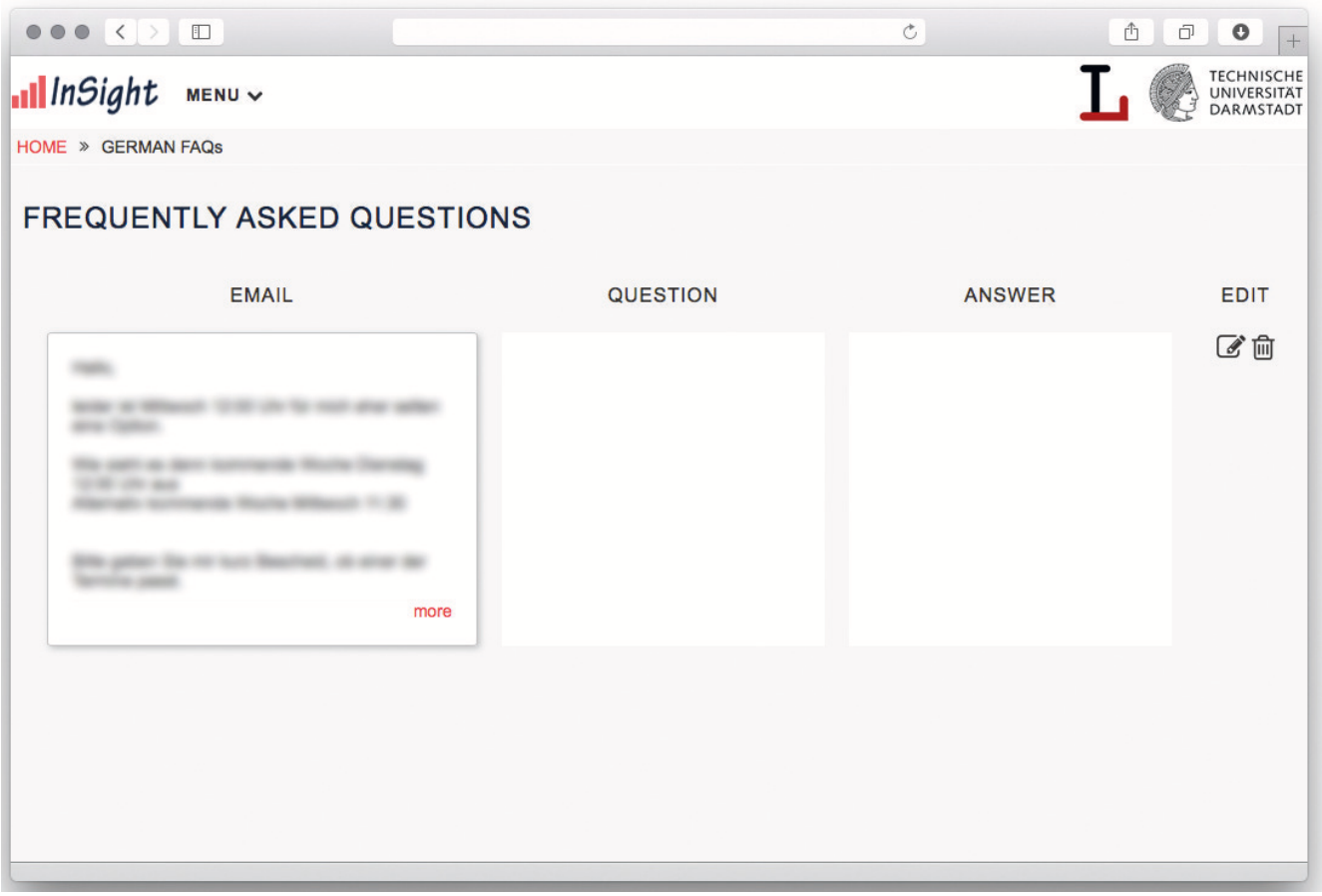

Abbildung 8.5: FAQ Merkzettel im Helpdesk `InSight`

Der FAQ Merkzettel in Abbildung 8.5 dient als ein Merkzettel für die E-Mails, die in den *Topics* als relevante oder häufig gestellte Anfragen identifiziert wurden und zwischengespeichert werden sollen. Gekennzeichnete E-Mails, wie in Abbildung 8.5, werden im FAQ Merkzettel angezeigt und sind in der Topic-Ansicht mit einem grünen Icon vermerkt. Die E-Mails werden in einer Liste untereinander aufgelistet. Zu jeder E-Mail kann, nachdem auf den Button zum Editieren geklickt wurde, die passende Frage und Antwort formuliert werden, die so in die richtige FAQ-Liste auf der FAQ-Webseite der Studienberatung aufgenommen werden soll. Im Editiermodus für eine E-Mail öffnen sich die Textfelder für die jeweilige E-Mail wie in Abbildung 8.6 gezeigt ist. Es erscheint ein Speicher-Button, durch welchen das Editieren geschlossen und gespeichert werden kann. Durch einen Klick auf den Mülleimer wird die E-Mail aus dem Merkzettel entfernt.

Abbildung 8.6: Editiermodus einer E-Mail im FAQ Merkzettel im Helpdesk `InSight`

8.4 User-Interface des Frage-Antwort-System

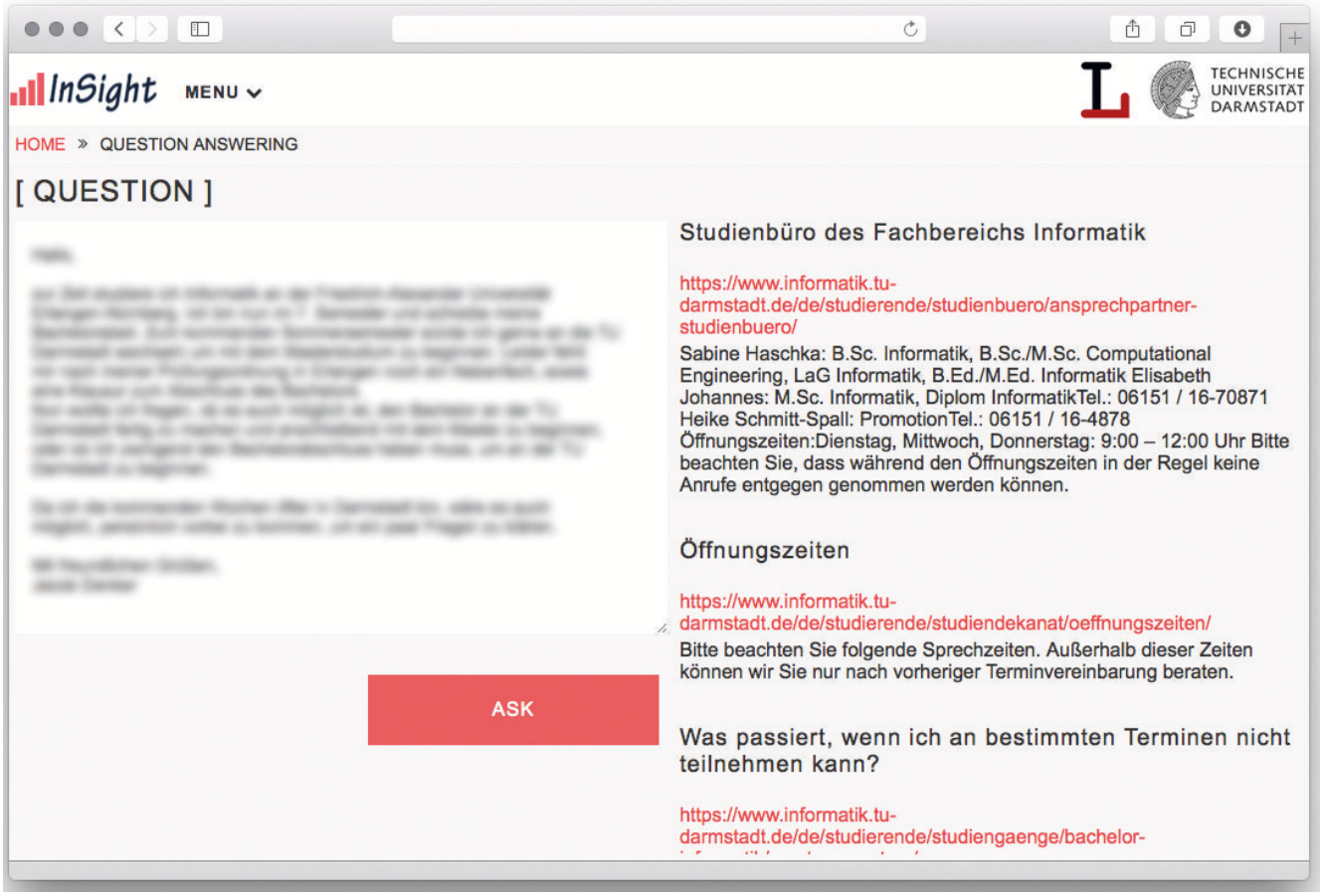

Abbildung 8.7: User-Interface des Frage-Antwort-Systems im Helpdesk `InSight`

Neben der Analyse der E-Mail-Anfragen von Studierenden und der Optimierung von Webseiten ist ein weiteres Ziel im Rahmen dierser Arbeit die Mitarbeiter bei der Beantwortung der Studierendenanfragen zu unterstützen. Dafür wurde das Frage-Antwort-System in das Helpdesk `InSight` integriert. Die Benutzerschnittstelle des Frage-Antwort-Systems ist in Abbildung 8.7 zu sehen. Die ganze E-Mail kann einfach aus dem E-Mail-Client in das Frageeingabefeld reinkopiert werden. Dabei wurde ein größeres Eingabefeld entworfen, um die ganze E-Mail-Anfrage auf einem Blick im Eingabefeld zu sehen. Die E-Mail wird vorverarbeitet und die entsprechenden wichtigen Informationen extrahiert. Die genaue Antwortgenerierung wird im folgenden Kapitel 9 beschrieben. Wird zum Beispiel ein Problem beschrieben und zum jeweiligen Problem nach einem persönlichem Gespräch gefragt, sind die möglichen Antwortvorschläge in der Abbildung 8.7 aufgelistet. Eine Antwort ist jeweils ein Webdokument mit Titel, URL der Webseite, aus der es gewonnen wurde, und dem eigentlichen Inhalt. Der Titel des Dokuments wird angezeigt, damit die Mitarbeiter auf den ersten Blick die Antwortvorschäge grob identifizieren können. Die URL wird gebraucht, falls die vorgeschlagene Antwort nicht ausreichend ist und weitere Informationen eingeholt werden müssen. Des Weiteren kann die Url genutzt werden, um dem Studierenden und auf seine E-Mail-Anfrage den Link zur Webseite mit der entsprechenden Antwort weiterzuleiten.

9 Antwortvorschläge in InSight

In diesem Kapitel wird das integrierte Frage-Antwort-System im Helpdesk `InSight` dargestellt und die einzelnen Schritte im Detail beschrieben. Im ersten Unterkapitel folgt eine kurze Einführung in Frage-Antwort-Systeme allgemein und im zweiten Unterkapitel wird die entwickelte Frage-Antwort-Verarbeitung in `Insight` im Detail präsentiert.

9.1 Frage-Antwort-Systeme

> *Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers).*
>
> *Manning et al. [27]*

Ein Frage-Antwort-System (QA) nutzt Techniken des Information Retrieval (IR). IR ist ein Forschungsgebiet der Informatik, welches sich mit der Speicherung von Daten und dem gezielten Finden von den gespeicherten Dokumenten beschäftigt. Das QA ist das Interface zwischen der Frage und den Daten (Storage), wie in Abbildung 9.1 visualisiert. Das QA liefert aus dem Storage Antwortvorschläge auf eine Frage.

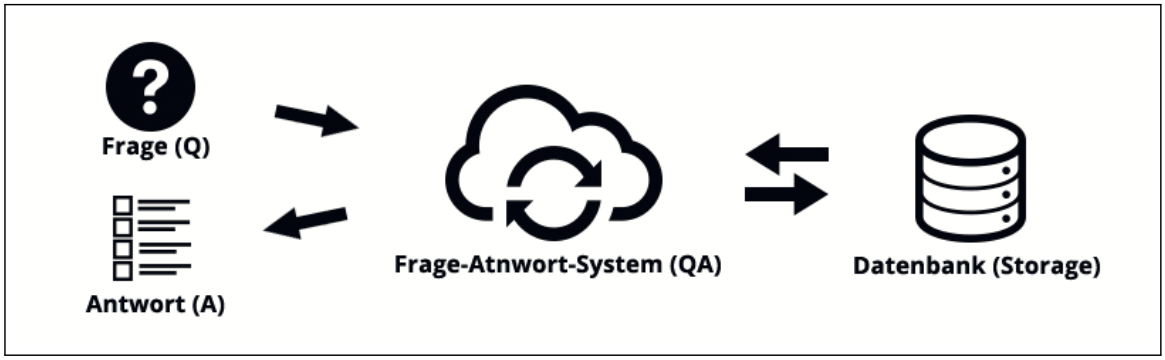

Abbildung 9.1: Frage-Antwort-System als Schnittstelle

Eine traditionelle QA-Pipeline besteht im Allgemeinen aus vier Schritten [2], die in Abbildung 9.2 visualisiert sind. Im ersten Schritt Fragenanalyse wird die Frage verarbeitet und wichtige Informationen werden extrahiert, die für die folgenden Schritte von Relevanz sind. Im zweiten Schritt wird anhand dieser extrahierten Informationen im Storage gesucht und ein relativ großer Pool an Antwortdokumenten wird zurück gegeben. Bei längeren Dokumenten muss die Antwort gegebenfalls aus dem Dokument extrahiert werden, welches im dritten Schritt stattfindet. Im letzten Schritt werden aus einem Pool an Antwortmöglichkeiten die Relevantesten selektiert und diese Entscheidung der Selektion validiert.

Ein erfolgreiches Frage-Antwort-System ist das IBM Watson [28], welches mit Erfolg an der amerikanischen Quiz-Show Jeopardy![1] teilgenommen hat. Das Augenmerkmal dieser Architektur liegt im Selektions- und Validierungsschritt. Um im *Information Retrieval* bessere Ergebnisse zu liefern, sollte sehr viel Wert auf die Selektion und Validierung von Antworten gelegt werden [2].

[1] http://www.jeopardy.com (30.10.2014)

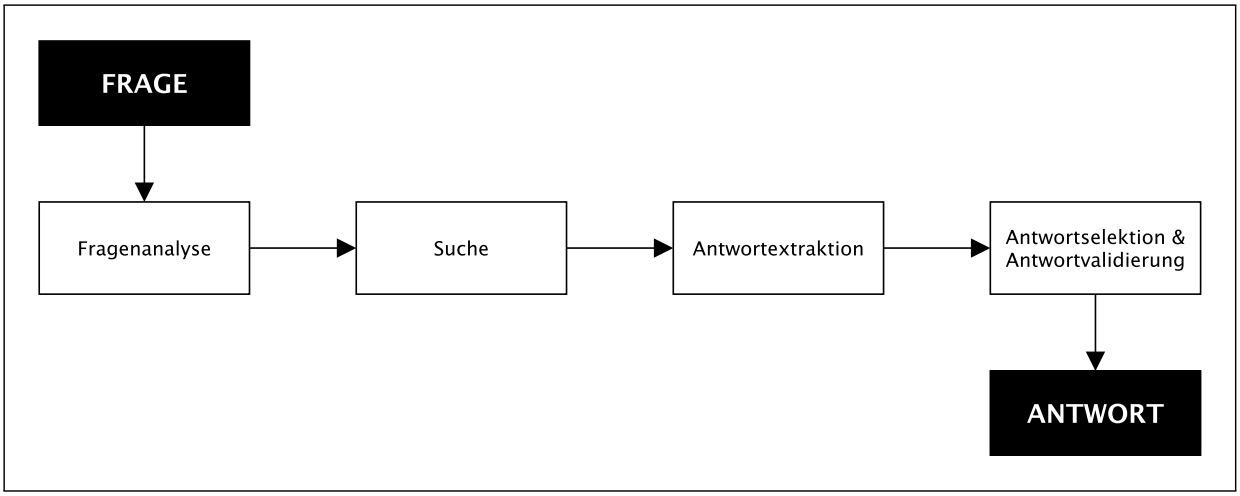

Abbildung 9.2: Traditionelle Frage-Anwort-Pipeline [2]

Apache Lucene

Viele Frage-Antwort-Systeme verwenden für die Suche nach relevanten Dokumenten in einem Korpus zu einer formulierten Frage die Volltextsuche Apache Lucene[2], welche innerhalb Appache Solr[3] als eine Serverapplikation angeboten wird.

TF-IDF

Die Apache Lucene Search Engine basiert auf dem TF-IDF-Maß [29], welches in Gleichung 9.1 definiert ist. Das TF-IDF-Maß berechnet die Relevanz eines Wortes t zu einem Dokument d, wobei gleichzeitig alle Dokumente D in Betracht gezogen werden. t ist umso relevanter für d, desto öfter t in d vorkommt ($tf(t,d)$) und je weniger t in allen anderen Dokumenten D vorkommt ($idf(t,D)$).

$$tf\text{-}idf(t,d,D) = tf(t,d) \cdot idf(t,D) \tag{9.1}$$

Die tf-Funktion ist in Gleichung 9.2 definiert. $f(t,d)$ zählt die absolute Häufigkeit von t in d. $max\{f(w,d) : w \in d\}$ ist die absolute Häufigkeit eines Wortes w, welches am häufigsten in d vorkommt.

$$tf(t,d) = 0.5 \cdot \frac{0.5 \cdot f(t)}{max\{f(w,d) : w \in d\}} \tag{9.2}$$

Die idf-Funktion ist in Gleichung 9.3 definiert. Es wird der Logarithmus, der die Division der Anzahl von Dokumenten D durch die Anzahl der der Dokumente in denen t vorkommt, berechnet.

$$idf(t,D) = log \frac{|D|}{|max\{d \in D : t \in d\}|} \tag{9.3}$$

[2] http://lucene.apache.org/core/ (05.11.2014)
[3] http://lucene.apache.org/solr/ (05.11.2014)

Lucene Score

Der Score (Relevanz) in der Apache Lucene Search Engine für eine Frage q, bestehend aus Wörtern t, zu einem Dokument d in einem Korpus wird mit der Funktion[4] in 9.4 berechnet.

$$score(q,d) = coord(q,d) \cdot queryNorm(q) \cdot \sum_{t \text{ in } q} (tf(t \text{ in } d) \cdot idf(t)^2 \cdot t.getBoost() \cdot norm(t,d)) \quad (9.4)$$

In der Funktion 9.4 ist die tf-Funktion wie in Gleichung 9.5 definiert.

$$tf(t) = frequency^{\frac{1}{2}} \quad (9.5)$$

Die idf-Funktion (9.6) ist mit einem Smooth-Faktor addiert, damit ein nicht Vorkommen eines Fragewortes t im Dokument d den Score aus 9.4 nicht zu sehr negativ beeinflusst.

$$idf(t) = 1 + log \frac{|D|}{|max\{d \in D : t \in d\}| + 1} \quad (9.6)$$

$coord(q,d)$ ist eine Funktion, die basierend auf der Häufigkeit der Fragewörter in einem Dokument einen Score berechnet. Je mehr Fragewörter in einem Dokument d vorkommen, desto höher ist der Score der Funktion.

$queryNorm(q)$ ist ein Normalisierungsfaktor, um die berechneten Scores für die verschiedenen Dokumente im Korpus vergleichbar zu machen.

$t.getBoost()$ ist das Gewicht für ein Wort t aus der Anfrage q. Damit lassen sich die Wörter in der Frage nach ihrer Relevanz zur ganzen Frage gewichten, um damit bessere Ergebnisse mit der Apache Lucene Search Engine zu erzielen.

$norm(t,d)$ ist ein Normalisierungsfaktor innerhalb der Berechnung des Score eines Wortes t zu einem Dokument d, um die Relevanz der Wörter miteinander zu d vergleichbar zu machen.

9.2 Frage-Antwort-Prozess in InSight

Dieses Unterkapitel behandelt das Thema der Antwortgenerierung auf eine neue eingehende Frage im System. In Abblidung 9.3 ist ein Überblick über den ganzen Prozess von der eingehenden Fragenanalyse bis zur Antwortselektion und Präsentation der Antwortvorschläge dargestellt. In den folgenden Abschnitten werden die einzelnen Schritte des Prozesses im Detail beschrieben.

Eingabe

In der Regel kommt die Studentenfrage per E-Mail im E-Mail-Client der Studienberatung an. Diese kann eins zu eins in das Fragenfeld in der Webapplikation `InSight` kopiert werden, welches in Abbildung 8.7 zu sehen ist. Das System verarbeitet auch manuell eingetippte Fragen und gibt Antwortvorschläge dazu.

Segmentierung

Der erste Schritt ist den wesentlichen Inhalt im Eingabetext zu identifizieren, dies trifft bei identisch kopierten E-Mails aus dem E-Mail-Client zu. Der Segmentierungsprozess ist in Kapitel 6.2 genau be-

[4] https://lucene.apache.org/core/4_3_0/core/org/apache/lucene/search/similarities/TFIDFSimilarity.html (05.11.2014)

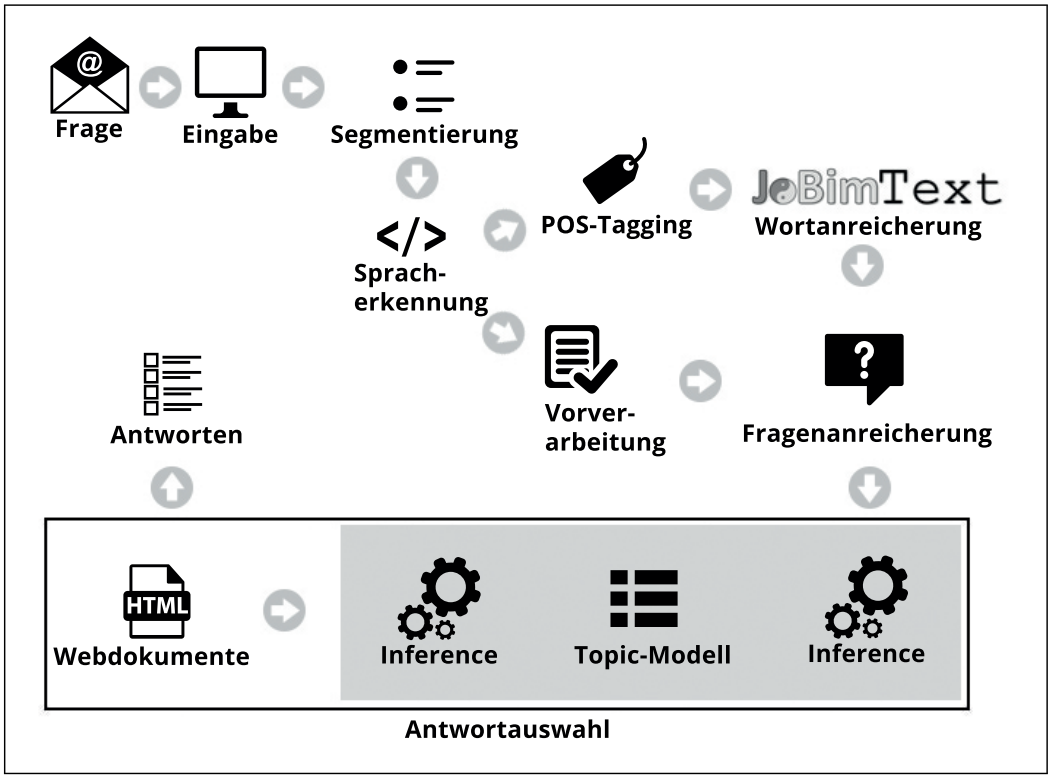

Abbildung 9.3: Frage-Antwort-Prozess in `InSight`

schrieben. Die E-Mail-Anfrage wird in Abschnitte unterteilt, wie zum Beispiel Anrede, Grußformel und Disclaimer. Die nicht relevanten Abschnitte werden durch vordefinierte Schüsselbegriffe identifiziert.

Spracherkennung

Die Sprache des Fragetexts wird anhand der relevanten Segmente bestimmt, die Spracherkennung ist in Kapitel 6.2 beschrieben. Jedem Segment wird eine Sprache zugeordnet. Die Fragesprache wird durch die Mehrheit der zugeordneten Segmentsprachen bestimmt.

Part of Speech (POS) Tagging

Für die Weiterverarbeitung werden nur die relevanten Abschnitte herangezogen und zusammengefügt. Die relevanten Abschnitte durchlaufen im Folgenden zwei voneinander unabhängige Prozessschritte. Der erste Prozessschritt beinhaltet das Part Of Speech (POS) Tagging [23], welches eine Zuordnung von Worten und Satzzeichen zu Wortarten unter Berücksichtigung des Kontexts herstellt. Mit dem POS Tagger werden die Nomen in den Sätzen idenfiziert. Im weiteren Verlauf wird die Frage durch semantisch in Beziehung stehende Wörter der Nomen angereichert.

Wortanreicherung

Für die identifizierten Nomen im Fragetext werden weitere semantisch in Beziehung stehende Wörter durch das JoBimText Framework[5] basierend auf [30] gesucht, mit denen die Frage angereichert wird. In [30] wurde die Idee des zweidimensionalen Textes präsentiert. Für ein Wort liefert dieser Ansatz die ähnlichsten Wörter zu diesem. In Abbildung 9.4 sind die acht ähnlichsten Wörter zu *cold* aufgelistet. Dieser Ansatz ist im JoBimText umgesetzt. Somit werden semantisch in Beziehung stehende Wörter zum identifizierten Nomen aus dem Fragetext gefunden. Als Ausgabe liefert dieser Schritt das Nomen aus dem Fragetext mit den dazu gefundenen semantisch in Beziehung stehenden Wörtern.

[5] http://maggie.lt.informatik.tu-darmstadt.de/jobimtext/ (30.10.2014)

Abbildung 9.4: Die acht ähnlichsten Wörter zu *cold* [3]

Vorverarbeitung

Bevor die Frage mit den semantisch in Beziehung stehenden Wörtern angereicht wird, wird der relevante Fragetext vorverarbeitet. Dabei werden folgende Schritte durchgeführt:

1. Entfernen von Zeilenumbrüchen.
2. Named Entity markieren. Der Prozess ist in Kapitel 4 beschrieben.
3. Satzzeichen aus dem Text entfernen.

Fragenanreicherung

Der Schritt der Fragenanreicherung erhält aus den Schritten davor zum einen den bereinigten Text und zum anderen die Nomen aus dem originalen Text mit den dazu in Beziehung stehenden semantisch ähnlichen Wörtern, die durch den Schritt der Wortanreicherung generiert wurden. Die Auswahl, welche Wörter aus der Wortanreicherung in die Frage aufgenommen werden, wird durch die Methode in Abbildung 9.5 umgesetzt. Die Methode (Abbildung 9.5) wurde iterativ mit der Testmenge A aus der Evaluation in Kapitel 10 optimiert, bis das Verfahren mit der Fragenanreicherung bessere Ergebnisse lieferte als ohne die Fragenanreicherung.

```
private static boolean isMember(String noun, String candidate) {

    if(noun.contains(candidate))
        return false;

    for(int i = 0; i < noun.length(); i++) {
        if(i + 6 < noun.length()) {
            String sixGram = noun.substring(i, i + 6);
            if(candidate.contains(sixGram))
                return true;
        }
    }

    return false;
}
```

Abbildung 9.5: Implementierte Methode zur Auswahl von Wörtern zur Fragenanreicherung

Liefert die Methode *isMember true* zurück, wird der *candidate* aus dem JoBimText Framework zum Fragetext hinzugenommen. Die Methode wird für jedes Paar von einem Nomen aus der Frage und den entsprechend erzeugten Kandidaten für das jeweilige Nomen durch den JoBimText angewendet. Die erste If-Anweisung verhindert Vorschläge in die Frage aufzunehmen, die durch eine Trennung von Konjunktionswörtern entstehen würde. Somit wird die Generalisierung des Nomen aus der Frage unterbunden. Dem entgegengesetzt werden Spezialisierungen der Nomen in andere semantische Richtungen gefördert und in die Frage aufgenommen. Dabei wird ein Sliding-Window mit der Länge von sechs Zeichen über den Text geschoben. Ist ein Fragment des Nomens im Kandidaten enthalten, wird die Frage durch den Kandidaten angereichert.

Antwortauswahl

In Abbildung 9.6 ist das Konzept der Antwortselektion auf eine neue Frage visuell dargestellt. Der Prozess zur Antwortauswahl auf eine neue Frage beinhaltet mehrere Schritte, die im Folgenden beschreiben werden:

- **Topic-Modell-Generierung**
  Die Antwortauswahl und das Matching zur eingehenden Frage erfolgt über das erzeugte probabilistische Mix-Topic-Modell. Das Mix-Topic-Modell wird sowohl aus den Webdokumenten als auch aus den vorhanden E-Mail-Dokumenten generiert. Die theorethischen Grundlagen zu probabilistischen Modellen sind in Kapitel 2.1 aufgeführt und die technologische Umsetzung in Kapitel 4.2 zu finden. Die Topic-Anzahl wurde mit n gleich 200 festgelegt, die restlichen Parameter des JGibbLDA wurden mit ihrer Standardparameterisierung zur Generierung des Modells übernommen.

- **Antwortvorverarbeitung**
  Das generierte Modell wird einmal erzeugt und kann dann benutzt werden, um neue Dokumente zu den generierten *Topics* zuzuordnen (Inference). In diesem Schritt werden die Webdokumente, die möglichen Antworten auf Fragen, mittels Inference den *Topics* aus dem Mix-Topic-Modell zugeordnet. Für die Inference wird das Gibbs Sampling, welches in Kapitel 2 beschrieben ist, benutzt sowie die Idee des Ansatzes aus [26] aufgegriffen. Die Inference bestimmt in jedem Inference-Schritt für jedes Wort in einem Dokument die Topic-Zuweisung. Der Topic-Tracker verfolgt und speichert die einzelnen Topiczuweisungen in jedem Schritt und bestimmt die Dokukment-Topic-Verteilung nach der Berechnung durch die beobachteten Topiczuweisungen. Die einzelnen Schritte zur Berechnung der Dokument-Topic-Verteilung sind im Folgenden aufgelistet und basieren auf der *mode method* aus [26]:

  1. Führe 200 Iterationen durch:
     a) Starte die Inference und führe 40 Inference-Schritte durch.
     b) Speichere in jedem Inference-Schritt die *Topics* und die absolute Häufigkeit von jedem *Topic* in der Topic-Wort-Zuweisung.
  2. Berechne die Wahrscheinlichkeit des Dokuments zu allen beobachteten *Topics* aus Schritt 1b.
     a) Für jedes beobachtete *Topic* i wird die relative Wahrscheinlichkeit berechnet:

$$\frac{\text{absolute Häufigkeit von } \textit{Topic } i}{\sum_{j=1}^{\text{alle beobachteten } \textit{Topics} T} \text{absolute Häufigkeit von } \textit{Topic } j} \tag{9.7}$$

  Die berechneten Dokument-Topic-Verteilungen der Webdokumente werden für die Antwortselektion einer Frage gebraucht.

- **Antwortselektion**
  Die Frage wird mit dem gleichen angepassten Verfahren zur Inference zu den *Topics* aus dem Mix-Topic-Modell zugeordnet, wie die Webdokumente im vorherigen Schritt. Die Fragen-Topic-Verteilung als auch die Webdokument-Topic-Verteilung ist bekannt. Die Antwortselektion ist in den folgenden Schritten dargestellt:

  1. Wähle die drei wahrscheinlichsten *Topics* T_3 für die Frage f.
  2. Wähle für jedes *Topic* t_i aus T_3 die zwei wahrscheinlichsten Webdokumente d_{ij} aus.
  3. Berechne für jedes Paar von Frage f und Webdokument d_{ij} den Score:

$$Score(d_{ij}) = probability(f, T_i) + probability(d_{ij}, T_i) \tag{9.8}$$

4. Ordne die Antworten absteigend nach dem Score.

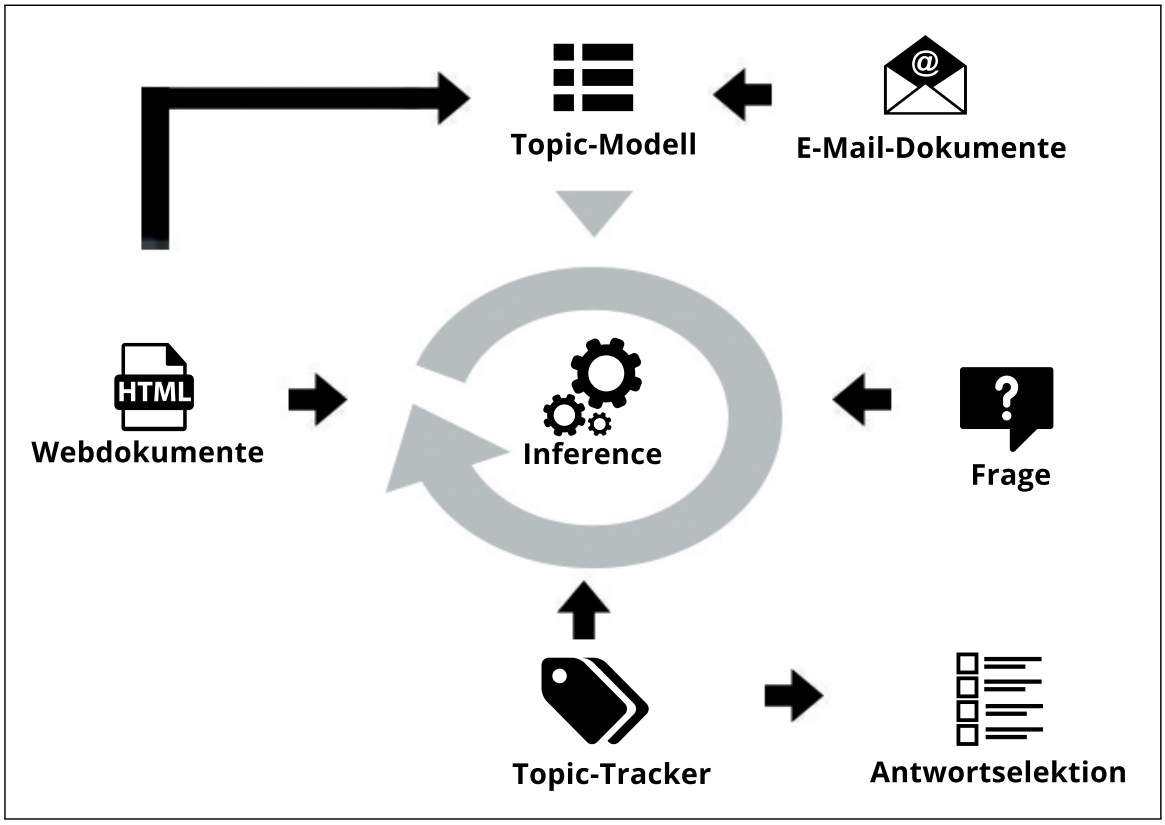

Abbildung 9.6: Konzept der Antwortauswahl

10 Evaluierung

In diesem Kapitel wird der entwickelte Topic-Modell-Ansatz für das Frage-Antwort-System aus Kapitel 9.2 mit Apache Solr verglichen, welches die Apache Lucene Volltextsuche verwendet, die in Kapitel 9.3 beschrieben wurde. Die zwei Ansätze werden anhand des Normalized Discounted Comulative Gain (NDCG) Maßes miteinandeinder verglichen. Das NDCG Maß ist im ersten Unterkapitel 10.1 beschrieben. Der experimentelle Aufbau ist in Kapitel 10.2 definiert. Die Evaluierung der Verfahren ist in zwei Schritten durchgeführt worden. Durch den ersten Schritt der Vorevaluation wurden mehrere Parameter für den entwickelten Ansatz bestimmt. Die Vorevaluation ist im Unterkapitel 10.3 dargestellt. In der Hauptevaluation in Unterkapitel 10.4 wird der Topic-Modell-Ansatz mit einer Volltextsuche verglichen.

10.1 Evaluierungsmaß

Das Normalized Discounted Comulative Gain (NDCG) Maß [31] dient zur Bewertung von vorgeschlagenen Antwortmöglichkeiten in einem Frage-Antwort-System. Dabei wird die Relevanz der Antwort und ihr Rang in der Antwortliste betrachtet. Die Annahme beim NDCG Maß ist, dass relevantere Dokumente zu einer Anfrage in der Antwortliste von Antworten vor unrelevanten Antworten aufgelistet sein sollten. Ein Frage-Antwort-System ist demnach besser, wenn es mehr relevante Antworten zu einer Frage ganz am Anfang in der Liste von Antwortvorschlägen präsentiert. Der einfachste Vorgänger des NDCG Maßes ist der Comulative Gain (CG) in Gleichung 10.1 formuliert. Beim CG Maß wird der Relevanz-Score rel_i der ersten p Antworten aufsummiert.

$$CG_p = \sum_{i=1}^{p} rel_i \tag{10.1}$$

Im CG Maß wird die Position der Antworten nicht betrachtet. Um die Position der Antwort bei der Berechnung der Antwortqualität eines Frage-Antwort-Systems mitzubetrachten, wird das Discounted Comulative Gain (DCG) Maß verwendet, welches in Gleichung 10.2 definiert ist. Das DCG Maß summiert den Relevanz-Score der Antworten i bis einschließlich Antwort p auf. Alle Relevanz-Scores ab der zweiten Antwort werden in der Aufsummierung durch $log_2(i)$ dividiert, damit werden Antworten, die im Ranking höher sind, größer gewichtet.

$$DCG_p = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{log_2(i)} \tag{10.2}$$

Die Anzahl der Antwortvorschläge variiert bei verschiedenen Anfragen. Um unterschiedliche Verfahren oder Frage-Antwort-Systeme miteinander zu vergleichen, muss das DCG Maß normailisiert werden, indem der DCG Score durch den Ideal Discounted Comulative Gain (IDCG) dividiert wird. Beim IDCG Maß werden zuerst die p Antworten nach ihrem Relevanz-Score rel_i absteigend sortiert und dann der DCG Score berechnet. In Abbildung 10.3 ist die Gleichung für den Normalized Discounted Comulative Gain (NDCG) formuliert.

$$NDCG_p = \frac{DCG_p}{IDCG_p} \tag{10.3}$$

Der Topic-Model-Ansatz und die Volltextsuche werden mit dem NDCG Score miteinander verglichen, wobei der Parameter p mit 1 und 3 gesetzt wurde. Die Schreibweise für die zwei Evaluierungsmaße sind

NDCG@1 und *NDCG@3*. Somit werden die zwei Verfahren anhand der relevantesten Antwort miteinander verglichen als auch durch die drei bestmöglichen Antworten der jeweiligen Verfahren.

10.2 Experimenteller Aufbau

Als Datengrundlage dienten sowohl die deutsprachigen Konversationen der E-Mail-Anfragen an die Studienberatung aus den Jahren von 2010 bis 2013 als auch die aktuellen deutschprachigen Webdokumente, die aus den Webseiten der Studienberatung generiert wurden. Insgesamt gab es 4148 E-Mail-Dokumente und 243 Webdokumente. Der Trainingskorpus zum Erzeugen des probabilistischen Topic-Modells aus Kapitel 9.2 bestand aus den E-Mail-Dokumenten aus den Jahren 2010 bis 2012 und den Webdokumenten. Der Testkorpus waren die E-Mail-Anfragen aus dem Jahr 2013. Der Trainingskoprus bestand aus 3231 E-Mail-Dokumenten und der Testkorpus aus 917 E-Mail-Dokumenten. Alle Datensätze sind in Tabelle 10.1 aufgelistet.

Korpus	**Datenmenge**	**Datengröße**
Insgesamt	E-Mail-Dokumente (2010 - 2013)	4148
	Webdokumente	243
Trainingskorpus	E-Mail-Dokumente (2010 - 2012)	3231
	Webdokumente	243
Testkorpus	E-Mail-Dokumente (2013)	917

Tabelle 10.1: Datensätze im Überblick

Der Datensatz `A` bestand aus 100 E-Mail-Dokumenten aus dem Testkorpus. Mit dem Datensatz `A` wurden in der Vorevaluation die Parameter für den Topic-Model-Ansatz bestimmt. Der Datensatz `B` besteht aus 40 extrahierten Dokumenten aus dem Testkorpus, die nicht im Datensatz `A` enthalten sind. Mit dem Datensatz `B` wurde der Topic-Modell-Ansatz mit der Volltextsuche verglichen. Die Antworten wurden durch zwei Personen bewertet, die im Studienbüro arbeiten und somit Experten in diesem Gebiet sind. Die erste Person `A` hat mit dem Datensatz `A` die Vorevaluation übernommen. Die zweite Person `B` bewertete in der Hauptevaluation mit dem Datensatz B die zwei Verfahren gegeneinander. Die Evaluationsgliederung ist in Abbildung 10.1 dargestellt.
Für die Volltextsuche wurde die Apache Solr mit der Apache Lucene Search Engine verwendet, die in Kapitel 9.1 beschrieben wurde. Es wurden die bereinigten Texte der Webdokumente indexiert.

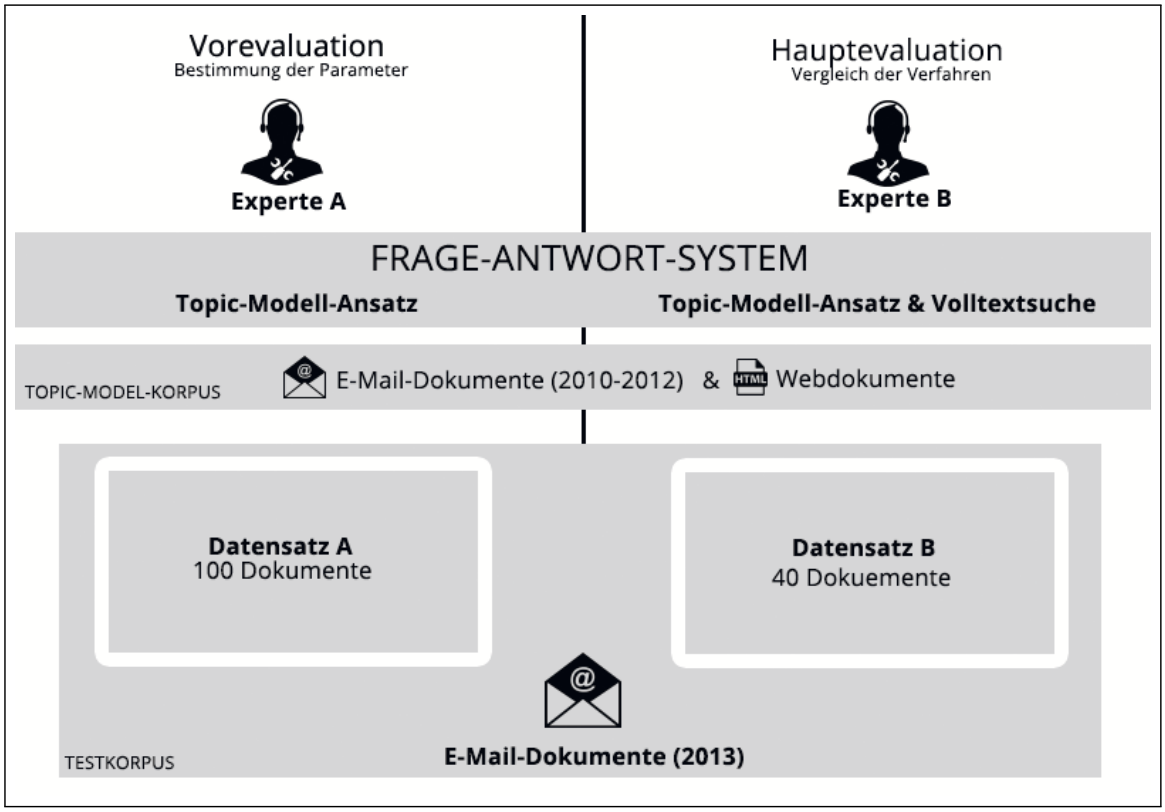

Abbildung 10.1: Visuelle Darstellung der Evaluationsgliederung

10.3 Bestimmung der Parameter

Die Vorevaluation zur Bestimmung der Parameter wurde durch den ersten Experten A vorgenommen, einem Mitarbeiter der Studienberatung. In Abblidung 10.2 ist die Benutzerschnittstelle dargestellt, die zur Bewertung von den Antworten zu einer Frage von dem Experten A benutzt wurde. Die Evaluationswebseite ist in zwei Hälften geteilt. Auf der rechten Seite sind Antworten zu einer E-Mail-Anfrage aufgelistet. In der Vorevaluation konnte jede passende Antwort zu einer Frage mit *good* bewertet werden. Falls eine Frage nicht als *good* markiert wurde, galt diese Antwort als nicht paasend. Somit war eine Antwort entweder zutreffend oder nicht. Die Antworten sind passende Dokumente zur E-Mail-Frage aus dem Webdokumentkorpus. Für jede Antwort wird der Titel des Webdokuments, die URL der Webseite, aus der das Webdokument generiert wurde, und der Antworttext angezeigt.

Datensatz A

Auf der linken Seite der Evaluationswebseite in Abbildung 10.2 ist die Frage visualisert. Falls die E-Mail nicht zur Evaluation geeignet war, wurde diese herausgefiltert. Entweder wurde diese E-Mail mit dem Button `skip` übersprungen oder mit dem Button `mark` vermerkt. Eine E-Mail-Frage kann zudem mit folgenden Labels gelabelt werden: *multiple questions*, *too specific*, *not on website*, *not a question* oder *spam*. Zudem kann auch zu jeder Frage eine eigene Notiz hinterlegt werden. Der verwendete Datensatz A in diesem Schritt enthielt nur Fragen von E-Mails, die durch die Expertenmeinung zur Evaluation herangezogen werden konnten, also E-Mails, die nicht vermerkt oder übersprungen wurden. Somit wurde der Datensatz A durch die Experten generiert, die entschieden haben, ob die Frage zur Evaluation herangezogen werden konnte. Insgesamt wurden 225 Fragen im Testkorpus in der Vorevaluation angeschaut, davon wurden 100 Fragen zur Bewertung des Frage-Antwort-System ausgewählt.

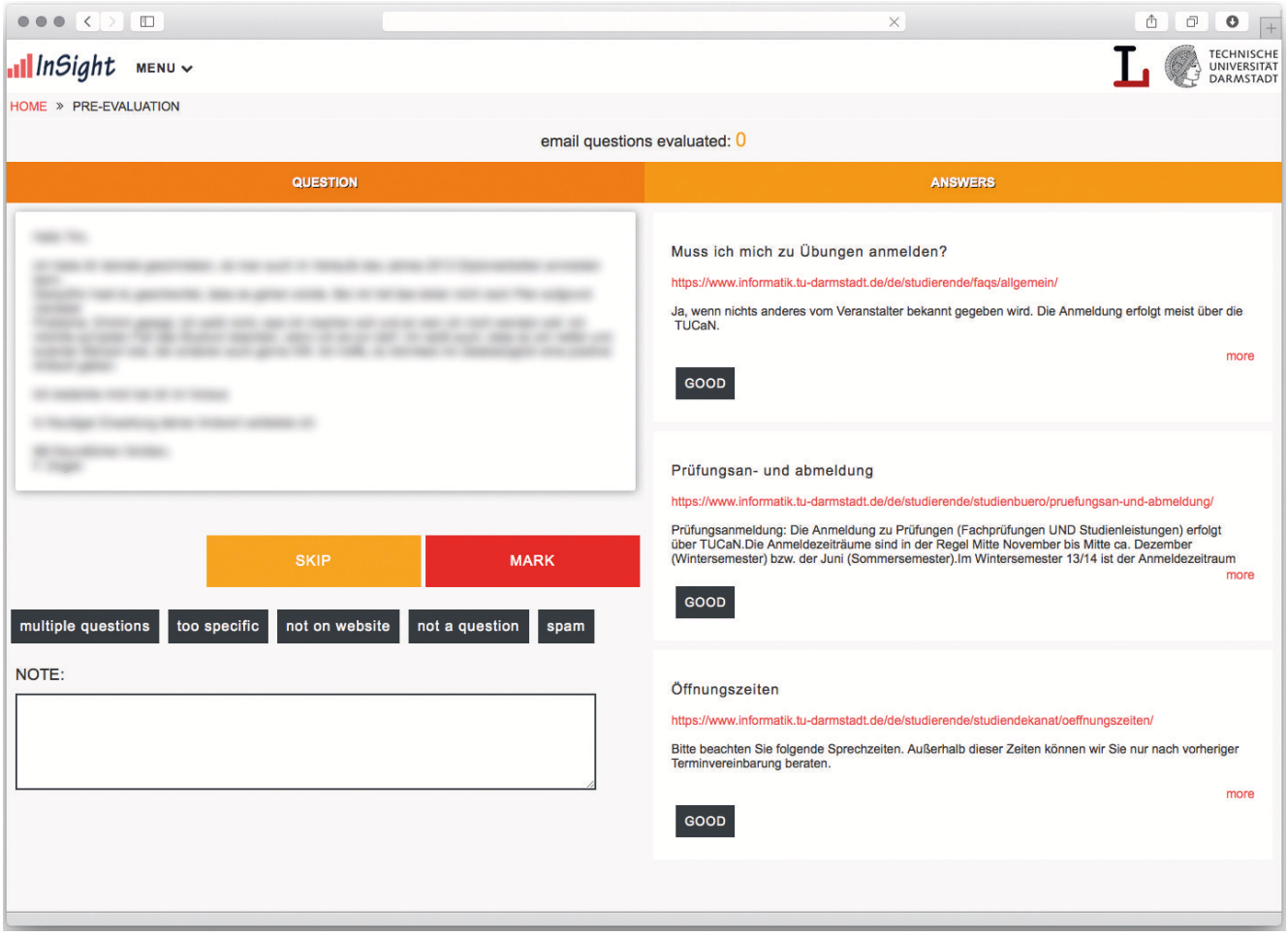

Abbildung 10.2: Benutzerschnittstelle der Vorevaluation zur Bestimmung der Parameter

Inference

Zum Matching von Fragen zu Antworten über das generierte Topic-Model wurde das angepasste Verfahren aus dem Kapitel 9.2 verwendet, welches auf der *mode method* aus [26] basiert. In der Vorevaluation wurde nur eine Iteration mit 40 Inference-Schritten durchgeführt.

Bestimmung des probabilistischen Modells

Durch die erste Evaluationsrunde wurde das Topic-Modell für den Topic-Modell-Ansatz bestimmt. Zur Auswahl standen drei probabilistische Topic-Modelle:

- E-Mail-Topic-Modell (nur aus E-Mail-Dokumenten)
- Web-Topic-Modell (nur aus Weddokumenten)
- Mix-Topic-Modell (aus E-Mail- und Webdokumenten)

Alle drei Topic-Modelle wurden mit der Topicanzahl n gleich 100 generiert. Insgesamt gab es sechs Antwortmöglichkeiten. Aus jedem Topic-Modell gab es je zwei Antwortvorschläge. Die Auswertung der probabilstischen Topic-Modelle ist in Tabelle 10.2 zu sehen.

Topic-Modell	**Anzahl an passenden Antworten (*"good"*)**
E-Mail-Topic-Modell	60
Web-Topic-Modell	40
Mix-Topic-Modell	78

Tabelle 10.2: Auswertung in Bezug zur Topic-Modell-Wahl

Die Antworten aus dem Mix-Topic-Modell wurden 78 mal als passend (mit *good*) markiert. Ausgehend von den Ergebnissen hat das Mix-Topic-Modell die meisten passendsten Antworten geliefert.

Bestimmung der Topic-Anzahl

Ein weiterer wichtiger Schritt ist die Auswahl der Topic-Anzahl n für das im vorherigen Schritt ausgewählte Mix-Topic-Modell. Getestet wurde das System mit einer Topic-Anzahl von 100, 200 und 500. Dabei wurden die Fragen aus dem Datensatz A verwendet. Neue Antworten, die nicht im ersten Schritt der Topic-Modell-Auswahl bewertet wurden, sind durch den Experten A nachbewertet worden. Dabei wurden die ersten drei Antwortenvorschläge pro Konfiguration bewertet, um den NDCG@1 und NDCG@3 auszurechnen. Die Ergebnisse sind in Tabelle 10.3 aufgelistet.

Das beste Ergebnis hat das Mix-Topic-Modell mit einer Topic-Anzahl von n gleich 200 erziehlt.

10.4 Vergleich der Verfahren

Die Hauptevaluation zum Vergleich vom entwickelten Topic-Model-Anssatz gegenüber einer Volltextsuche mit Lucene wurde durch den zweiten Experten B vorgenommen, einem weiteren Mitarbeiter der Studienberatung. In Abblidung 10.2 ist die Benutzerschnittstelle dargestellt, die zur Bewertung der Frage-Antwort-Paare in der Vorevaluation verwendet wurde. In der Hauptevaluation wurde die Bewertung einer Antwort erweitert. Eine Antwort kann jetzt durch eine dreistufige Skala bewertert werden, wie in Abbildung 10.3 dargestellt. Ein Frage-Antwort-Paar kann mit *good*, *partly correct* oder *bad* bewertet werden.

Topic-Anzahl n	NDCG@N	Ergebnis
100	NDCG@3	0.282
	NDCG@1	0.16
200	NDCG@3	0.434
	NDCG@1	0.23
500	NDCG@3	0.332
	NDCG@1	0.12

Tabelle 10.3: Ergebnisse des Mix-Topic-Modells für unterschiedliche n

Abbildung 10.3: Bewertungsskala der Frage-Antwort-Paare in der Hauptevaluation

Datensatz B

In der Hauptevaluation konnte eine E-Mail wie in der Vorevaluation vermerkt oder übersprungen werden. Der verwendete Datensatz B in dieser Bewertung enthielt nur Fragen von E-Mails, die durch die Expertmeinung als nutzbar definiert wurden. Insgesamt gab es 40 E-Mail-Dokumente von 120 betrachteten E-Mail-Anfragen, die durch den Experten B aus dem Testkorpus extrahiert und zur Evaluationsbewertung herangezogen wurden. Die betrachteten 225 Fragen aus der Vorevaluation überschneiden sich nicht mit den 120 betrachteten Fragen aus der Hauptevaluation.

Inference

Zum Matching von Fragen zu Antworten über das generierte Topic-Modell wurde das angepasste Verfahren aus dem Kapitel 9.2 verwendet, welches auf der Idee des Verfahrens *mode method* aus [26] basiert. Im Gegensatz zur Vorevaluation wurden in der Hauptevaluation 200 Iterationen mit jeweils 40 Inference-Schritten durchgeführt. Durch die Erhöhung der Iterationen im Verfahren steigt die Stabilität der Dokument-Topic-Verteilung in der Berechnung zu den drei wahrscheinlichsten Topics für das jeweilige Dokument. Zum Vergleich der Stabilität des Verfahrens wurde für jede Frage aus dem Datensatz A 10 mal die Inference durchgeführt. Dabei wurde die relative Häufigkeit pro Frage betrachtet: Wie oft taucht das gleiche *Topic* jeweils maximal auf Platz 1, 2 oder 3 auf? Wobei der Platz 1 das wahrscheinlichste *Topic*

nach der Berechnung darstellt. Die Ergebnisse in Tabelle 10.4 stellen die durchschnittlichen Ergebnisse für unterschiedliche Anzahlen an Iterationen mit dem Verfahren dar.

Iterationen	Platz	Wahrscheinlichkeit
1	1	0.886
	2	0.707
	3	0.557
50	1	0.971
	2	0.933
	3	0.894
100	1	0.978
	2	0.948
	3	0.923
200	1	0.979
	2	0.957
	3	0.943

Tabelle 10.4: Ergebnisse der Stabilität der Inference bei unterscheidlicher Anzahl an Iterationen

Topic-Modell-Ansatz vs. Volltextsuche

Beim Vergleich des Topic-Modell-Ansatzes und der Volltextsuche wurden insgesamt sechs Antworten präsentiert: Die drei passendsten Antworten aus dem Topic-Modell-Ansatz und die drei passendsten Antworten aus der Volltextsuche mit Lucene. In Tabelle 10.5 sind die Ergebnisse des Vergleichs aufgelistet. Der entwickelte Topic-Modell-Ansatz hat deutlich bessere Ergebnisse geliefert als die Volltextsuche.
Für die 40 Fragen aus dem Datensatz B hat der Topic-Modell-Ansatz 18 mal eine passende Antwort geliefert, mindestens eine Frage war teilweise passend *partly correct* für die jeweilige gestellte Frage. Die Volltextsuche lieferte nur 8 mal eine passende Antwort zu den gestellten E-Mail-Anfragen aus dem Datensatz B. Im Durchschnitt gab es pro Frage eine passende Antwort aus dem Topic-Modell-Ansatz oder aus der Volltextsuche, falls es eine passende Antwort gab. Die Verteilung der bewerteten Antworten bezogen auf ihren Rang in der Antwortliste ist in Tabelle 10.6 aufgelistet.
Bei kurzen E-Mail-Anfragen, die maximal einen Satz lang waren, lieferte die Volltextsuche bessere Antworten als der Topic-Modell-Ansatz. Im Allgemeinen beinhalten E-Mail-Anfragen indirekte Fragen mit eine zusätzlichen Beschreibung, in diesen Fällen war der Topic-Modell-Ansatz deutlich besser als die Volltextsuche. Die richtigen Antworten vom Topic-Modell-Ansatz wurde geliefert, wenn das Thema der E-Mail-Anfrage zu den frequenten Topics im E-Mail-Korpus gehörte, wie zum Beispiel E-Mail-Anfragen zu möglichen Sprechstunden, Beratungsterminen, Prüfungsfragen und Problemen mit TUCaN. Diese Themen können in Abbildung 8.2, in der die relevanten *Topics* aus dem E-Mail-Korpus abgebildet sind, wiedergefunden werden.

Verfahren	NDCG@N	Ergebnis
Topic-Modell-Ansatz	NDCG@3	0.425
	NDCG@1	0.325
Volltextsuche	NDCG@3	0.215
	NDCG@1	0.1

Tabelle 10.5: Ergebnisse der Hauptevaluation

		Anzahl der Antworten		
Verfahren	**Rang**	**good**	**partly correct**	**bad**
Topic-Modell-Ansatz	1	10	3	27
	2	5	1	34
	3	3	2	35
Volltextsuche	1	1	3	36
	2	2	3	35
	3	1	1	38

Tabelle 10.6: Bewertungsmatrix zum Rang und dessen Antwortmenge bezogen auf die Relevanz der Antwort in der Hauptevaluation

11 Ausblicke und Verbesserungen

Verschiedene Weiterentwicklungs- und Verbesserungsmöglichkeiten ergeben sich für das entwickelte Helpddesk `InSight`. Das Frage-Antwort-System kann in vieler Hinsicht weiterentwickelt werden, um bessere Antworten auf Fragen vorzuschlagen. Die Benutzerfreundlichkeit, Visualisierung und Interkationsmöglichkeiten für die Datenanalyse sind im aktuellen entwickelten System nicht ausgeschöpft und bieten viel Freiraum für weitere Ideen. Der ganze Prozess der Verarbeitung von Daten und weiteren Integrationsmöglichkeiten bieten viele weitere Optimierungsmöglichkeiten. Einige dieser Aspekte werden im Folgenden aufgegriffen.

Integration

Aktuell muss der Mitarbeiter im Studienbüro die E-Mail aus dem E-Mail-Client kopieren und in das Frageneingabefeld in der Webapplikation eingeben, um einen Antwortvorschlag zu bekommen. Die Integration in bestehende genutzte Systeme wie E-Mail-Clients oder Ticketsystem als auch andere Helpdseks ohne dieses Feature steigert die Performance der Mitarbeiter. Der Mitarbeiter bekommt zur gleichen Zeit im bestehenden System die Frage und die vorgeschlagenen Antworten visualisiert. Eine mögliche technologische Architektur in der Anbindung an einen E-Mail-Client und den entsprechenden E-Mail-Server ist in [32] präsentiert. Durch die Integration können die Topic Models automatisch erzeugt werden und es muss keine PST Datei zur Erzeugung des E-Mail-Korpus erstellt werden.

Ausreißer Detektion

Durch eine direkte Anbindung an die bestehenden genutzten Systeme im Support kann der Verlauf vom Eingang der Frage bis zur Bantwortung dieser durch einen Mitarbeiter mitverfolgt werden. Das entwickelte System weiß damit, inwieweit die Antwort zur Frage aus den Antwortvorschlägen übernommen sowie überarbeitet wurde oder selber erstellt wurde. Durch diese Informationen lässt sich über die Zeit bestimmen, ob das Topic-Model neu gelernt werden sollte, weil es keine sinnvollen Antworten auf bestimmte Fragen kennt. Ein einfacher naiver Ansatz wäre, wenn eine gewisse Anzahl an Ausreißern in einem gewissen Zeitabstand gesammelt wurde, sollte das Topic-Model neu generiert werden. Wie aufkommende Topics über die Zeit erkannt werden können, ist in [33] dargestellt.

Topic Model Matching

Die Topic-Modelle werden in gewissen Abständen immer wieder neu generiert, weil neue Themen auftauchen und die Topic-Modelle an die neuen Themen durch eine Neugenerierung angepasst werden müssen. Wenn eine Menge von unterschiedlichen Topic-Modellen über die Zeit vorhanden ist, bietet es sich an, diese miteinadner zu vergleichen. Ansätze zur visuellen Analyse zweier probabilistischer Topics sind in [34] präsentiert.

Spam Topic Tagger

Ein weiteres Feature zur Weiterentwicklung ist, dem Benutzer die Möglichkeit zur Markierung bestimmter Topics zu geben, die bei der Inference nicht berücksichtigt werden sollen. Damit kann die Genauigkeit bei der Inference erhöht werden, die Wörter bekommen in den Inferenceschritten keine Topics zugeordnet, die nicht für den Anwendungsfall relevant sind.

Question Answer Type

Eine mögliche Optimierung des Frage-Antwort-Systems ist den Type der Antwort zur gestellten Frage herauszufinden. Damit lassen sich die Frage als auch die Antworten kategorisieren. Eine Antwort könnte zum Beispiel die Uhrzeit oder der Ort eines bestimmten Termins sein. Durch diese Information lässt sich die Antwortselektion und Antwortvalidierung optimieren. Der Antworttext würde sich auch auf die

wesentlichen Informationen begrenzen können. Ein Ansatz zur Analyse des Question Answer Types ist in [35] zu finden.

Webseitenverarbeitung

Aktuell werden die Webseiten mit einem relativ naiven Ansatz in Webdokumente gegliedert. Viele Informationen überschneiden sich auf den Webseiten der Studienberatung, es gibt sehr viele quasi gleiche Inhalte auf den Webseiten. Im aktuellen Prozess werden die Webdokumente erzeugt und nur genau gleiche Webdokumente entfernt. Viele Webdokumente sind zu lang oder beinhalten keine relevanten semantischen Informationen. Hier muss die Verarbeitung der Webseiten optimiert werden. Des Weiteren werden keine PDF's gecrawlt. Durch das nicht Verarbeiten von PDF Dokumenten gehen viele relevante Informationen zur Beantwortung von E-Mail-Anfragen verloren. Wie Dokumente satzweise miteinander verglichen als auch segmentiert werden können, ist in [1] beschrieben. Evenutell könnten auch die in dieser Arbeit vorgestellten Key Phrases beim Analysieren der Webdokumente helfen.

E-Mail-Verarbeitung

In der E-Mail-Verarbeitung finden sich auch einige Punkte, die optimiert werden können. Die Segmentierung und Erkennung der relevanten Inhalte erfolgt über definierte Regeln. In zukünftigen Arbeiten kann ein Klassifizierer antrainiert werden, der die relevanten Inhalte eventuell besser identifiziert und nicht so stark fehleranfällig ist auf unvorhergesehene E-Mail-Strukturen, wie die jetztigen implementierten Regeln. In dieser Arbeit wurde dem Betreff kein besonderes Augenmerk geschenkt. Wahrscheinlich lässt sich viel mehr relevante semantische Informationen aus dem Betreff einer E-Mail extrahieren, als es bis jetzt in der vorliegenden Arbeit der Fall ist. Eine satzweise Analyse der E-Mail-Anfrage würde den relevanten Inhalt in der E-Mail besser eingrenzen, um so bessere Antwortvorschläge zu geben. Des Weiteren könnten auf diese Weise E-Mails ausgeschlossen werden, die keine Fragen beinhalten. Viele E-Mail-Konversationen gehen über einen längeren Zeitraum, somit auch über mehrere Themen, die voneinander ganz unbhängig sind. Eventuell sollten E-Mail-Konversationen durch diesen Aspekt getrennt werden. E-Mails fallen eher in die Kategorie der kürzeren Dokumente. Wie zum Beispiel mit kurzen Dokumenten und probabilistischen Modellen umgegangen wird, ist in [36] und in [37] beschrieben. Weiter Ideen, wie E-Mails und die Struktur der E-Mails im Verlauf durch probabilstische Modelle segmentiert werden können, ist in [38] als auch in [26] beschrieben.

Antwortkorpus

Im aktuellen System werden die Antworten nur aus den Webdokumente generiert. Durch eine bessere E-Mail-Verarbeitung könnten die Antworten in den E-Mails identifiziert werden und in den Antwortkorpus mitaufgenommen werden. Mit extrahierten Frage-Antwort-Paaren könnte das aktuelle Frage-Antwort-System optimiert werden.

Topic Naming

Die Topics im LDA sind durch ihre Verteilung über ein fixes Vokabular definiert. Der Name eines Topics bildet sich in der Regel aus den Top k Wörtern, die für das jeweilge Topic am wahrscheinlichsten sind. Die Darstellung der Wörter als Bag-Of-Words liefert in vielen Fällen nicht auf den ersten Blick die gewünschte Erkenntnis. Ein Ansatz zur besseren Bestimmung von Topic Namen ist in [39] definiert.

Self-Service

Um die E-Mail-Anfragen zu reduzieren kann das Frage-Antwort-System als webbasierter Self-Service auf der Webseite der Studienberatung angeboten werden, wie zum Beispiel iHelp aus [1]. In diesem Zusammenhang muss das Frage-Antwort-System für kürzere Fragetexte optimiert werden.

Key Phrases

Der naive Ansatz der Key Phrases ist noch nicht voll ausgeschöpft. Viele Phrasen sind identisch, wenn die Reihenfolge außer Acht gelassen wird. Die Key Phrases sollten im Falle der E-Mails auf E-Mail-Ebene und nicht auf einer E-Mail-Konversation erzeugt werden. Eventuell sollten Patterns eingefügt werden, um bestimmte Positionen in den Phrasen nur durch bestimmte Wortarten zu belegen.

Datenanalyse

Die Visualisierung der Topic-Modelle bietet viel Freiraum für Weiterentwicklungen. Ziel der Datenanalyse ist es, Unstimmigkeiten zwischen zwei unterschiedlichen Korpora durch die generierten probabilsitischen Modelle aufzudecken. Im akutellen Zustand wird für ein E-Mail-Topic der Verlauf der passenden Webdokumente im Liniendiagramm dargestellt (Abbildung 8.2 und 8.3). In diesem Fall fehlt die Information, aus welchen Webseiten die Dokumente extrahiert wurden, welches wichtig in diesem Zusammenhang wäre. Es fehlt die Informationsdarstellung der relevantesten Unterschiede zwischen den Topic-Modellen auf einen Blick. Hier sollten in zukünftigen Arbeiten Algorithmen entwickelt werden, mit denen zwei Topic-Modelle anhand ihrer Topic-Wort-Verteilung und Dokument-Topic-Verteilung als auch mittels der Inference der Dokumente aus dem jeweils anderen Korpus miteinander verglichen werden können. Hierzu gibt es Ansätze in [40] und [41], wie Dokumente mittels Topic-Modellen miteinander verglichen werden können

Fragenanreicherung

Im aktuellen System werden für die Nomen aus einer Frage mittels des JoBimText Framework ähnliche Wörter gefunden, welche Kandidaten für die Anreicherung der Frage sind. Die Methode in Abbildung 9.5 entscheidet, ob aus einem Kandidaten eine wirkliche Fragenanreicherung wird. Dies ist ein sehr naiver Ansatz, der noch ausbaufähig ist. Einerseits werden durch die Methode 9.5 zu viele Kandidaten abgeschnitten und andererseits zu viele in anderer Richtung zugelassen. Hier müssen bessere Algorithmen entwickelt werden, welche Wortkandidaten eine Verbesserung der Antwortfindung als auch Selektion bewirken.

Summarization

Das System iHelp [1] bietet als Antworten automatisch erstellte Zusammenfassungen, welches eine weitere Entwicklungsrichtung für das in dieser Masterarbeit entwickelte Helpdesk wäre.

FAQ

Im akutellen System gibt es nur einen manuell zu pflegenden Merkzettel für eine FAQ Liste. Durch eine Weiterentwicklung der Software bis zur Identifizierung von Fragen und Antworten in den E-Mails könnten Frage-Antwort-Paare erstellt werden. Mit einer direkten Anbindung an den E-Mail-Server und der dadurch möglichen Beobachtung des E-Mail-Verkehrs wären alle Notwendigkeiten gegeben, um automatisch themenspezifische akutelle FAQ Listen zu erstellen.

12 Konklusion

Die vorliegende Arbeit erforschte zwei Themen. Zum einem wurde überprüft, inwieweit automatisch generierte probabilistische Topic-Modelle für Frage-Antwort-Systeme in Betracht kommen, zum anderem wurde untersucht, inwiefern sich zwei Korpora miteinander durch ihre probabilistischen Modelle vergleichen lassen. Das Ziel anhand eines probabilistischen Modells auf Fragen die passende Antwort zu finden, wurde im Rahmen dieser Arbeit zufriedenstellend erreicht und regt für weitere Arbeiten in diesem Themengebiet an. Das Thema wurde in Zusammenarbeit mit der Studienberatung des Fachbereichs Informatik an der Technischen Universität Darmstadt erforscht. Als Grundlage dienten die E-Mail-Kommunikationen zwischen den Studierenden mit ihren Fragen und den Mitarbeitern der Studienberatung sowie die Webseiten der Studienberatung. Im Rahmen dieser Arbeit wurde das Helpdesk `InSight` entwickelt, welches ein Frage-Antwort-System beinhaltet und durch die Visualisierung der probabilistischen Modelle den Webseteinkorpus sowie den E-Mail-Korpus analysiert und dadurch die zwei Korpora miteinander verglichen werden können.

Frage-Antwort-System

Das integrierte Frage-Antwort-System im Helpdesk `InSight` wurde durch die Mitarbeiter der Studienberatung evaluiert. Der entwickelte Topic-Modell-Ansatz im Frage-Antwort-System wurde gegen die Volltextsuche Lucene[1] getestet und erreichte deutlich bessere Ergebnisse, die in Tabelle 10.5 dargestellt sind. Die erzielten Ergebnisse sind vielversprechend. Die aufgeführten Punkte im vorherigen Kapitel für zukünftige Arbeiten an diesem Thema sollten in Betracht gezogen werden und für weitere Entwicklungs- und Verbesserungsmöglichkeiten anreizen. Das entwickelte Baseline-System bietet eine gute Grundlage für zukünftige Arbeiten.

Datenanalyse

Das Helpdesk `InSight` bietet nebem dem Frage-Antwort-System auch die Möglichkeit, die verschiedenen Datensätze miteinander zu vergleichen, wie in Kapitel 8 präsentiert. Die Mitarbeiter der Studienberatung können die einzelnen Themen der E-Mail-Anfragen und die Themen auf den Webseiten analysieren und erkennen, welche Themen wie gut auf den Webseiten abgedeckt sind und anhand welcher Themen die Webseiten reorganisiert werden sollten. Die Datenanalyse bietet eine Menge von Weiterentwicklungsmöglichkeiten, wie im vorherigen Kapitel aufgeführt.

FAQ

Aktuell bietet das Helpdesk `InSight` nur einen Merkzettel, der in Kapitel 8 dargestellt ist. Die aufgeführten Punkte aus dem vorherigen Kapitel zur besseren Analyse der E-Mail-Anfragen und Separierung von Webseiten sowie deren Zuordnung und Dokumentenbildung werden langfristig zu der Möglichkeit führen, FAQs automatisch zu generieren.

[1] http://lucene.apache.org/core/ (02.11.2014)

Durch die Datenanalyse bekommen die Mitarbeiter der Studienberatung zum ersten Mal eine Gliederung und Zusammenfassung des Inhalts der E-Mail-Anfragen von Studierenden und den Webseiten der Studienberatung. Das ist ein erster Schritt zur Analyse und zum Vergleich der zwei Korpora. Die Datenanalyse ist noch nicht voll ausgeschöpft und bietet noch keinen detailierten Einblick, wo genau welche Themen auf den Webseiten nicht abgedeckt sind und auf welche spezifischen Fragen die Webseiten keine Antwort liefern.
Das Frage-Antwort-System liefert vielversprechende Ergebnisse zur Weiterentwicklung. Es funktioniert auf E-Mail-Fragen, die die Frage auch umschreiben und nicht nur einen Fragesatz enthalten, relativ gut. Zum Einsatz als webbasierter Self-Service muss die Antwortgenerierung auf Fragen mit nur einem Fragesatz optimiert werden. Weitere Weiterentwicklungsmöglichkeiten wurden im Kapitel 11 aufgelistet.

Literaturverzeichnis

[1] Dingding Wang, Tao Li, Shenghuo Zhu, and Yihong Gong. iHelp: An Intelligent Online Helpdesk System. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, pages 173–182, 2011.

[2] Anselmo Peñas, Eduard H. Hovy, Pamela Forner, Álvaro Rodrigo, Richard F. E. Sutcliffe, Corina Forascu, and Caroline Sporleder. Overview of QA4MRE at CLEF 2011: Question Answering for Machine Reading Evaluation. In *CLEF (Notebook Papers/Labs/Workshop)*, 2011.

[3] Chris Biemann and Martin Riedl. From Global to Local Similarities: A Graph-Based Contextualization Method using Distributional Thesauri. *Proceedings of the 8th Workshop on TextGraphs in conjunction with EMNLP*, October 2013.

[4] David M. Blei. Introduction to Probabilistic Topic Models. *Commun. ACM*, 55(4):77–84, April 2012.

[5] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet Allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[6] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal ot the American Sciety for Information Science*, 41(6):391–407, 1990.

[7] Thomas Hofmann. Probabilistic Latent Semantic Indexing. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, pages 50–57, New York, NY, USA, 1999. ACM.

[8] Mark Steyvers and Tom Griffiths. *Probabilistic Topic Models*. Lawrence Erlbaum Associates, 2007.

[9] Yee W. Teh, David Newman, and Max Welling. A Collapsed Variational Bayesian Inference Algorithm for Latent Dirichlet Allocation. In B. Schölkopf, J.C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 1353–1360. MIT Press, 2007.

[10] Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic Variational Inference. *J. Mach. Learn. Res.*, 14(1):1303–1347, May 2013.

[11] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101(Suppl. 1):5228–5235, April 2004.

[12] W. R. Gilks. *Markov Chain Monte Carlo In Practice*. Chapman and Hall/CRC, 1999.

[13] D.J. Arnold, Lorna Balkan, Siety Meijer, R.Lee Humphreys, and Louisa Sadler. *Machine Translation: an Introductory Guide*. Blackwells-NCC, London, 1993.

[14] Ronan Collobert and Jason Weston. Fast Semantic Extraction Using a Novel Neural Network Architecture. *in Proc. ACL*, pages 560–567, 2007.

[15] Martha Palmer, Daniel Gildea, and Paul Kingsbury. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Comput. Linguist.*, 31(1):71–106, 2005.

[16] Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.

[17] Yi Zhang, Jamie Callan, and Thomas Minka. Novelty and Redundancy Detection in Adaptive Filtering. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '02, pages 81–88, New York, NY, USA, 2002. ACM.

[18] Dingding Wang, Tao Li, Shenghuo Zhu, and Chris Ding. Multi-document Summarization via Sentence-level Semantic Analysis and Symmetric Matrix Factorization. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 307–314, New York, NY, USA, 2008. ACM.

[19] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Scoiety, Series B*, 39(1):1–38, 1977.

[20] Xin Liu, Yihong Gong, Wei Xu, and Shenghuo Zhu. Document Clustering with Cluster Refinement and Model Selection Capabilities. In *In Proceedings of the 25th Annual International ACM Sigir Conference on Research and Development in Information Retrieval*, pages 191–198, New York, NY, USA, 2002. ACM Press.

[21] Dragomir R. Radev, Eduard Hovy, and Kathleen McKeown. Introduction to the Special Issue on Summarization. *Comput. Linguist.*, 28(4):399–408, 2002.

[22] David Guthrie, Ben Allison, W. Liu, Louise Guthrie, and Yorick Wilks. A Closer Look at Skip-gram Modelling. *In Proceedings of the Fifth international Conference on Language Resources and Evaluation (LREC)*, pages 1222–1225, 2006.

[23] Kristina Toutanova and Christopher D. Manning. Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-speech Tagger. In *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora: Held in Conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13*, pages 63–70, 2000.

[24] Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA, 1999.

[25] A. chaney and D. Blei. Visualizing Topic Models. *International AAAI Conference on Social Media and Weblogs*, 2012.

[26] Martin Riedl and Chris Bieman. Text Segmentation with Topic Models. *JLCL*, 27(1):47–69, 2012.

[27] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[28] David A. Ferrucci, Eric W. Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John M. Prager, Nico Schlaefer, and Christopher A. Welty. Building Watson: An Overview of the DeepQA Project. *AI Magazine*, 31(3):59–79, 2010.

[29] Ho Chung Wu, Robert Wing Pong Luk, Kam Fai Wong, and Kui Lam Kwok. Interpreting TF-IDF Term Weights As Making Relevance Decisions. *ACM Trans. Inf. Syst.*, 26(3):13:1–13:37, June 2008.

[30] Chris Bieman and Martin Riedl. Text: Now in 2D! A Framework for Lexical Expansion with Contextual Similarity. *Journal of Language Modelling*, pages 55–95, 2013.

[31] Kalervo Järvelin and Jaana Kekäläinen. Cumulated Gain-based Evaluation of IR Techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, October 2002.

[32] Frank Wagner, Kathleen Krebs, Cataldo Mega, Bernhard Mitschang, and Norbert Ritter. Towards the Design of a Scalable Email Archiving and Discovery Solution. In Paolo Atzeni, Albertas Caplinskas, and Hannu Jaakkola, editors, *ADBIS*, volume 5207 of *Lecture Notes in Computer Science*, pages 305–320. Springer, 2008.

[33] Ankan Saha and Vikas Sindhwani. Learning Evolving and Emerging Topics in Social Media: A Dynamic Nmf Approach with Temporal Regularization. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, WSDM '12, pages 693–702, 2012.

[34] Patricia Crossno, Andrew T. Wilson, Timothy M. Shead, and Daniel M. Dunlavy. TopicView: Visually Comparing Topic Models of Text Collections. In *ICTAI*, pages 936–943. IEEE, 2011.

[35] Partha Pakray, Pinaki Bhaskar, Somnath Banerjee, Bidhan Chandra Pal, Sivaji Bandyopadhyay, and Alexander F. Gelbukh. A Hybrid Question Answering System based on Information Retrieval and Answer Validation. In Vivien Petras, Pamela Forner, and Paul D. Clough, editors, *CLEF (Notebook Papers/Labs/Workshop)*, 2011.

[36] Rishabh Mehrotra, Scott Sanner, Wray Buntine, and Lexing Xie. Improving LDA Topic Models for Microblogs via Tweet Pooling and Automatic Labeling. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 889–892, New York, NY, USA, 2013. ACM.

[37] Daniel Ramage, Susan T. Dumais, and Daniel J. Liebling. Characterizing Microblogs with Topic Models. In William W. Cohen and Samuel Gosling, editors, *ICWSM*. The AAAI Press, 2010.

[38] Shafiq Joty, Giuseppe Carenini, Gabriel Murray, and Raymond T. Ng. Exploiting Conversation Structure in Unsupervised Topic Segmentation for Emails. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 388–398, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[39] Abram Hindle, Neil A. Ernst, Michael W. Godfrey, and John Mylopoulos. Automated Topic Naming to Support Cross-project Analysis of Software Maintenance Activities. In *Proceedings of the 8th Working Conference on Mining Software Repositories*, MSR '11, pages 163–172, New York, NY, USA, 2011. ACM.

[40] Wayne Xin Zhao, Jing Jiang, Jianshu Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li. Comparing Twitter and Traditional Media Using Topic Models. In *Proceedings of the 33rd European Conference on Advances in Information Retrieval*, ECIR'11, pages 338–349, Berlin, Heidelberg, 2011. Springer-Verlag.

[41] Xiaojun Quan, Gang Liu, Zhi Lu, Xingliang Ni, and Liu Wenyin. Short text similarity based on probabilistic topics. *Knowl. Inf. Syst.*, 25(3):473–491, December 2010.