Konzept und Architektur eines Software-Werkzeuges zur automatisierten Identifikation und Analyse von Argumentationsstrukturen

Constantin Houy, Tim Niesen, Jesús Calvillo, Peter Fettke, Peter Loos

Institut für Wirtschaftsinformatik (IWi)
Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI)
und Universität des Saarlandes
Campus, Geb. D 3_2
66123 Saarbrücken
{constantin.houy, tim.niesen, jesus.calvillo, peter.fettke, peter.loos}@iwi.dfki.de

Abstract: Die Entwicklung überzeugender Argumentation ist – ebenso wie die Analyse gegebener Argumentationsstrukturen – eine wichtige Aufgabe der Rechtswissenschaft. Die Formulierung rechtswissenschaftlicher Argumentation stellt eine anspruchsvolle intellektuelle Aufgabe dar, die sich auf möglichst viele relevante Hintergrundinformationen stützen sollte. Einer ständig wachsenden Anzahl verfügbarer Gerichtsentscheidungen steht dabei die beschränkte menschliche Informationsverarbeitungskapazität gegenüber. Um diesen Problemen zu begegnen, wird im Rahmen des vom BMBF-geförderten Konsortialprojektes ARGUMENTUM ein Software-Werkzeug entwickelt, das eine automatische Identifikation und Analyse von Argumentationsstrukturen in den elektronisch verfügbaren Entscheidungen des Bundesverfassungsgerichts unterstützen soll. Im vorliegenden Beitrag werden das Konzept sowie die Architektur des ARGUMENTUM Software-Werkzeuges präsentiert und erste Einblicke in die aktuelle Entwicklung des Prototyps gegeben.

1 Einleitung

Das Argumentieren und die Analyse von Argumentationsstrukturen sind anspruchsvolle intellektuelle Aktivitäten und darüber hinaus bedeutsame Aufgaben der Rechtswissenschaft sowie der juristischen Praxis. In Argumentationen werden Rechtfertigungen oder Widerlegungen von Behauptungen entwickelt, um Personen von der Korrektheit oder Falschheit einer bestimmten Aussage zu überzeugen. Neben der theoretischen Auseinandersetzung mit Argumentation, die von der Rechtswissenschaft gepflegt wird, ist im Kontext der juristischen Praxis vor allem die Identifikation relevanter Argumente zu einer Rechtsfrage ausgesprochen wichtig. Im Gegensatz zum Fallrecht (engl. *Case-Law*), das insbesondere im anglo-amerikanischen Rechtskreis Anwendung findet, schreibt das deutsche Rechtssystem zwar nicht zwingend eine Berücksichtigung bereits ergangener Entscheidungen vor, allerdings orientiert sich die Rechtsprechung in Deutschland häufig auch an thematisch ähnlichen Fällen. Juristen, die sich in unterschiedlichen Rollen z.B. als Richter oder Anwalt mit einer Rechtsfrage beschäftigen, müssen sich daher einen Überblick über Argumente zu einem Rechtsproblem und deren Darlegung in Gerichtsur-

teilen verschaffen, um diese bei der Erstellung ihrer eigenen Argumentation zu dieser Rechtsfrage berücksichtigen zu können.

Typischerweise sind Juristen nicht sämtliche potentiell relevanten Argumente zu einer Rechtsfrage *ad-hoc* präsent. Bei einer ständig zunehmenden Anzahl von Gerichtsentscheidungen erscheint es mit Blick auf die natürlichen Limitationen der menschlichen Informationsverarbeitungskapazität zunehmend schwierig bis unmöglich, alle Argumente in ergangenen Urteilen zu einem bestimmten Thema zu kennen. Vor dem Hintergrund einer ständig wachsenden Verfügbarkeit elektronischer Entscheidungscorpora bedienen sich deshalb zunehmend mehr Juristen der Möglichkeiten einer rechnergestützten Suche nach Argumenten. Allerdings verfügen aktuelle Suchmaschinen i. d. R. nur über die Möglichkeit der Volltextsuche in Gerichtsurteilen. Eine gezielte Suche nach bestimmten Typen von Argumentationsstrukturen oder nach *Pro-* oder *Contra-*Argumenten zu einer bestimmten Rechtsfrage ist mithilfe aktueller Suchmaschinen bisher nicht möglich.

Um verschiedene Unzulänglichkeiten gegenwärtiger Suchmethoden überwinden zu können, werden im Rahmen des aktuellen, vom BMBF geförderten Konsortialforschungsprojektes ARGUMENTUM die Potenziale und Grenzen einer automatisierten Identifikation und Analyse von Argumentationsstrukturen in Gerichtsurteilen mithilfe von Methoden der *Künstlichen Intelligenz* untersucht. Ein Ziel des Projektes liegt in der Entwicklung eines Software-Werkzeuges, das ein gezieltes Suchen und Auffinden von Argumenten zu einer bestimmten Rechtsfrage ermöglichen soll [Ho12]. Anwender, die nach Argumenten in Gerichtsentscheidungen zu einem bestimmten Thema suchen, sollen mit diesem Werkzeug schneller und einfacher relevante Argumente identifizieren können. Bei der Prototypentwicklung wird das elektronisch verfügbare und frei zugängliche Corpus der Entscheidungen des Bundesverfassungsgerichts (*BVerfG*) zugrunde gelegt.[1]

Das *Ziel* dieses Beitrags ist es, sowohl das im Rahmen des Projektes entwickelte *Konzept* für die Aufbereitung des BVerfG-Entscheidungscorpus zur Unterstützung einer automatisierten Identifikation und Analyse von Argumentationsstrukturen als auch die *Architektur* des entsprechenden *Software-Werkzeuges* vorzustellen. Außerdem sollen erste Ergebnisse der aktuell laufenden Implementierung des Prototyps vorgestellt werden.

Hinsichtlich seiner *Forschungsmethode* basiert der vorliegende Beitrag auf einem *gestaltungsorientierten Ansatz* [He04]. Neben einer Analyse der Struktur des BVerfG-Entscheidungscorpus wurde eine Anforderungserhebung aus unterschiedlichen Nutzerperspektiven durchgeführt. Auf dieser Basis wurde dann mit der iterativen Gestaltung des Software-Werkzeuges unter Berücksichtigung dreier zentraler Phasen der Informationssystementwicklung (analog zum *ARIS-Phasenkonzept* nach Scheer [Sc02]) fortgefahren, nämlich mit der Erstellung: 1. eines *Fachkonzeptes*, 2. eines technischen *Datenverarbeitungs(DV)-Konzeptes* und 3. der *Implementierung*. Insbesondere die Ergebnisse dieser drei Phasen werden im Rahmen des vorliegenden Beitrags ausführlicher beschrieben. Da aufgrund der Laufzeit des Projektes ARGUMENTUM (*Juni 2012 - Mai 2015*) die Implementierung des Prototyps aktuell noch nicht abgeschlossen ist, wird im Rahmen dieses Beitrags nicht der finale Prototyp präsentiert, sondern ein aktueller Stand, der bereits zahlreiche Funktionalitäten anbieten kann.

[1] http://www.bundesverfassungsgericht.de/entscheidungen.html

Der vorliegende Beitrag ist wie folgt *strukturiert*: nach dieser Einleitung wird im folgenden Kapitel der aktuelle Stand der Forschung im Kontext der automatisierten Identifikation und Analyse von Argumentationsstrukturen beschrieben. Kapitel 3 präsentiert strukturelle Besonderheiten des verwendeten Entscheidungscorpus im Hinblick auf die anschließende Konzeptentwicklung. In Kapitel 4 wird das Konzept (*Fachkonzept*) für eine automatisierte Erschließung des Corpus eingeführt, bevor Kapitel 5 den Entwurf der Architektur des Software-Werkzeuges zur automatischen Identifikation und Analyse von Argumentationsstrukturen (*DV-Konzept*) präsentiert. Im Anschluss wird in Kapitel 6 der aktuelle Stand der Implementierung des Prototyps gezeigt. Kapitel 7 schließt den Beitrag mit einem Resümee und einem Ausblick.

2 Stand der Forschung

Zum Thema automatisierte Identifikation und Analyse von Argumentationsstrukturen existieren einige Vorarbeiten. Diese sind vor allem in den Bereichen *Computer-Supported Argumentation* [Sc10, Lu97] sowie im Themenfeld *Argumentation Mining* [MM11] angesiedelt. Es existieren im Bereich *Computer-Supported Argumentation* zahlreiche Systeme und Prototypen, die eine Untersuchung und Aufbereitung von Argumentationsstrukturen unterstützen. Der Beitrag von Scheuer et al. [Sc10] gibt zum aktuellen Stand einen umfassenden Überblick. Ein Großteil der dort aufgeführten Software-Werkzeuge unterstützt insbesondere das Erlernen logisch korrekter Argumentation. Einige dieser Systeme ermöglichen auch eine semi-automatische Aufbereitung von Argumentationen in Internetforen. Eine eingehende automatisierte Analyse von Argumentationsstrukturen wird von den meisten Systemen bisher nicht unterstützt.

Hinsichtlich einer automatisierten Identifikation und Analyse von Argumentationsstrukturen in Texten ist infolgedessen eine intensivere Betrachtung des Forschungsgebietes *Argumentation Mining* notwendig. Unter dem Begriff *Argumentation Mining* werden Ansätze zur Identifikation von Argumenten in elektronischen Texten verstanden, die auf Text-Mining-Ansätzen basieren. Die Forschung zum Argumentation Mining hat in den vergangenen Jahren an Bedeutung gewonnen und wurde auch im Kontext der Rechtswissenschaft untersucht, da Texte aus diesem Bereich häufig zu einem gewissen Grad formalisiert sind [Mo07, Wy10]. Es existieren Text-Mining-Ansätze zur Identifikation und Analyse argumentativer Strukturen, die sich verschiedener Ansätze des *Maschinellen Lernens* (ML) bedienen [MM11]. Diese Ansätze wurden bisher allerdings ausschließlich für die Analyse englischsprachiger Texte entwickelt. Argumentation-Mining-Ansätze und entsprechende Werkzeuge, die in der Lage sind, deutschsprachige Texte zu analysieren, sind den Autoren dieses Beitrags nicht bekannt.

Es existieren inzwischen einige Auszeichnungssprachen zur Annotation von Argumentationsstrukturen, wie z. B. die XML-basierte *Argument Markup Language* (AML), die das Training von ML-Methoden unterstützen kann, als Teil des sogenannten *Araucaria*-Systems [RR04]. Die theoretische Grundlage für solche Annotationssysteme bilden im Bereich der philosophischen Argumentationslehre bekannte Argumentationstheorien wie z. B. das *Toulmin*-Schema [To75] oder die Argumentationsschemata nach *Walton* [Wa96]. Argumentationstheorien bzw. -schemata sind sprachunabhängig einsetzbar und

könnten auch im Rahmen der Entwicklung eines Software-Werkzeuges zur Identifikation und Analyse von Argumentationsstrukturen in der deutschen Sprache angewendet werden. Somit können auch diese Techniken einen Beitrag zur Behebung des oben beschriebenen Defizits leisten. Im folgenden Kapitel werden das BVerfG-Entscheidungscorpus sowie seine strukturellen Charakteristika etwas ausführlicher vorgestellt und hinsichtlich der angestrebten automatisierten Verarbeitung diskutiert.

3 Das Entscheidungscorpus des Bundesverfassungsgerichts

Im Rahmen des Projektes ARGUMENTUM stellt die Aufbereitung des elektronisch verfügbaren Entscheidungscorpus des BVerfG – ein frei zugängliches Textcorpus mit inzwischen mehr als 5.000 Einzelentscheidungen aus den Jahren 1998 bis heute – für eine automatische Identifikation und Analyse von Argumentationsstrukturen ein zentrales Ziel dar. Typischerweise gliedern sich die meisten Entscheidungen des BVerfG, die umfassende Begründungen enthalten, in folgende fünf Bereiche:

1. *Leitsätze*: Zusammenfassung der Kernaussagen der Urteilsentscheidung, die allerdings keine Begründungen für die Entscheidung enthält,

2. *Rubrum*: Vorstellung der am Verfahren beteiligten Parteien und des Gerichts sowie Erläuterung des Verfahrensgegenstandes,

3. *Tenor*: Entscheidungssatz des Verfahrens, der teilweise Aussagen über die vorläufige Vollstreckbarkeit und Verfahrenskosten enthält,

4. *Tatbestand*: Überblick über den unstreitigen Sachverhalt, den streitigen Parteivortrag und eine eventuelle Verfahrenshistorie sowie

5. *Entscheidungsgründe*: Begründung des Entscheidungssatzes im Tenor. Dieser Abschnitt enthält die eigentliche Argumentation zur Entscheidung.

Während die Abschnitte *eins* bis *vier* typischerweise aus Freitextpassagen bestehen, enthält der fünfte Abschnitt eine etwas „strenger organisierte“ Unterstruktur: Sätze, die entweder inhaltlich oder argumentativ in Zusammenhang stehen, sind in fortlaufend nummerierten Abschnitten zusammengefasst. Ergänzend zu diesen Nummerierungen existieren verschiedene Stufen von Subnummerierungen. Auf Grundlage einer Voruntersuchung von BVerfG-Entscheidungen ist momentan davon auszugehen, dass die Nummerierungen relevante Anhaltspunkte für argumentative Zusammenhänge zwischen Bestandteilen der Entscheidungsgründe darstellen können.

Weitere bedeutende Merkmale sind neben der Struktur einer BVerfG-Entscheidung ihre *Zulässigkeit* sowie ihre *Begründetheit*. Während die Zulässigkeit formale Kriterien wie die Zuständigkeit des Gerichts und die fristgerechte Antragsstellung betrifft, drückt die Begründetheit die Beurteilung des Gerichts hinsichtlich der Frage aus, ob eine Verfassungsklage berechtigt war. Nach unserer Voruntersuchung ist davon auszugehen, dass auch die in den Entscheidungen zur *Zulässigkeit* und *Begründetheit* von Urteilen ver-

wendeten Formulierungen hinreichend konsistent sind, um regelbasierte Text-Mining-Ansätze für deren automatisierte Analyse einzusetzen. Im folgenden Kapitel wird nun das entwickelte Konzept für das angestrebte Software-Werkzeug präsentiert.

4 Konzept zur Aufbereitung des BVerfG-Entscheidungscorpus

Entwickelt wurde das im Folgenden präsentierte Konzept unter Berücksichtigung des aktuellen Stands der Forschung zum Argumentation Mining, der erwähnten Voruntersuchung zur Struktur des BVerfG-Entscheidungscorpus sowie auf Basis einer Anforderungsanalyse, die mit den Partnern des Konsortialprojektes mithilfe verschiedener Nutzungsszenarien erarbeitet wurde. Es wurden dabei grundsätzlich zwei Nutzerperspektiven unterschieden: 1.) die Perspektive des juristischen Praktikers, dem Unterstützung bei der Recherche in einer Fachdatenbank angeboten werden soll, z. B. einem Anwalt, der gezielt nach Argumenten sucht, die bei einer bestimmten Entscheidung erfolgreich waren und 2.) die Perspektive des Rechtswissenschaftlers, der durch die gezielte Untersuchung argumentativer Zusammenhänge neue Erkenntnisse generieren möchte oder Antworten auf argumentationstheoretische Fragestellungen sucht.

Vor dem Hintergrund der oben beschriebenen Vorarbeiten im Bereich Argumentation Mining, wurden bestehende Ideen und Argumentation-Mining-Ansätze adaptiert und an die Besonderheiten des BVerfG-Corpus angepasst. Abb. 1 zeigt eine Übersicht über das entwickelte Phasenkonzept, welches als Rahmenwerk für die Implementierung des ARGUMENTUM Software-Werkzeuges zur Identifikation und Analyse von Argumentationsstrukturen dient und in [Ho13] ausführlich präsentiert wurde.

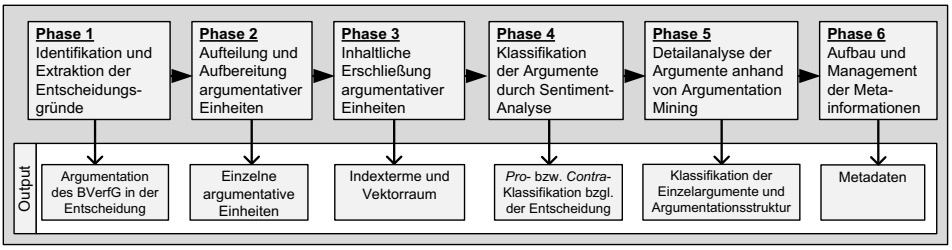

Abb. 1: Übersicht über das Konzept zur Aufbereitung des BVerfG-Corpus

Phase 1: Im ersten Schritt ist es notwendig, alle argumentativen Textpassagen einer Entscheidung zu identifizieren und für eine weitere Analyse zu extrahieren. Aufgrund der ausgeprägten Standardisierung und Strukturiertheit des Corpus kann dies vorwiegend mit regelbasierten Methoden geschehen.

Phase 2: Im Anschluss werden die zusammenhängenden Textpassagen in kleinere argumentative Einheiten zerlegt, um sie einer detaillierteren Analyse (*Phase 5*) zugänglich zu machen. Da Entscheidungen eine interne Struktur haben, wird aktuell eine strukturelle Trennung auf Ebene der Abschnitte vorgenommen.

Phase 3: Um eine automatisierte Identifikation von Argumenten bzgl. ihrer Inhalte zu ermöglichen, werden alle argumentativen Einheiten zunächst inhaltlich erschlossen. Bedeutungstragende Worte (sog. *Indexterme*) werden extrahiert und auf deren Basis die Textinhalte in eine Vektordarstellung (engl. *Vector Space Model*) transformiert [JM09, S. 802ff.]. Dies ermöglicht das Auffinden und Vergleichen der Inhalte argumentativer Einheiten anhand mathematischer Verfahren.

Phase 4: Bevor eine detaillierte Analyse der Argumentation in einzelnen Abschnitten durchgeführt werden kann, muss zunächst bestimmt werden, ob die Gesamtargumentation eines Urteils *für* oder *gegen* eine vorgebrachte Klage ausfällt. Dies soll mit Ansätzen und Verfahren aus dem Bereich *Sentiment-Analyse* bewerkstelligt werden [Ho10].

Phase 5: Aufbauend auf den Ergebnissen der vorangegangenen Phasen kann nun eine Feinanalyse einzelner Abschnitte und Sätze durchgeführt werden. Dazu sollen Ansätze des Argumentation Mining aus [MM11] verwendet werden, die auf Charakteristika der deutschen Sprache anzupassen sind. Solche Analysen können u. a. die Klassifikation von Aussagen, z. B. als *Prämisse* oder *Konklusion* nach dem *Toulmin*-Schema, oder eine Beschreibung der Beziehungen zwischen argumentativen Teilaussagen ermöglichen.

Phase 6: Abschließend werden die im Rahmen der Analyse gewonnenen Erkenntnisse und Metadaten zusammengestellt und gespeichert. Eine strukturierte Ablage dieser Informationen ist entscheidend, da sie erheblichen Einfluss auf die Performanz des zukünftigen Software-Prototyps haben wird. Im folgenden Kapitel wird nun aufgrund dieses fachlichen Konzeptes eine Architektur für das Werkzeug (*DV-Konzept*) entwickelt.

5 Architektur des Software-Werkzeuges

Das bereits vorgestellte Konzept bildet die Grundlage für die Entwicklung der Architektur des ARGUMENTUM Software-Werkzeuges. Abb. 2 auf der folgenden Seite gibt einen Überblick über diese Architektur. Die einzelnen Klassen und Software-Module werden von links nach rechts gemäß dem dargestellten fachlichen Konzept angeordnet, was eine Zuordnung der Klassen und Module zu den einzelnen Phasen ermöglicht.

Das Modul zur Phase 1 (*ArgumentationExtractor*) übernimmt dabei das Einlesen aller Entscheidungen des BVerfG-Corpus. Aus jeder Entscheidung werden dann die Entscheidungsgründe extrahiert, welche die Argumentation des BVerfG enthalten, bevor in Phase 2 durch den *ParagraphTokenizer* eine Trennung dieser Argumentation in „argumentative Einheiten“ vorgenommen wird. In Phase 3 werden durch das Modul *IndexTermExtractor*, das aus sieben Submodulen besteht, für jede argumentative Einheit Daten generiert, die die Erzeugung eines Indexterm-Vektorraums und somit das Auffinden von Argumenten zu einem bestimmten Thema unterstützen. Das Software-Modul zu Phase 4 (*SentimentAnalyzer*) soll eine regelbasierte Klassifikation von Argumenten (*Pro* vs. *Contra*) ermöglichen, während der *ArgumentAnalyzer* eine weiterführende Detailanalyse einzelner Argumente in Phase 5 erzielen soll. Der *MetadataManager* unterstützt eine strukturierte Verwaltung der entwickelten Analysedaten. Folgende Abschnitte stel-

len die einzelnen Softwaremodule der Architektur detaillierter vor und erläutern grundlegende Annahmen im Kontext der Prototypimplementierung.

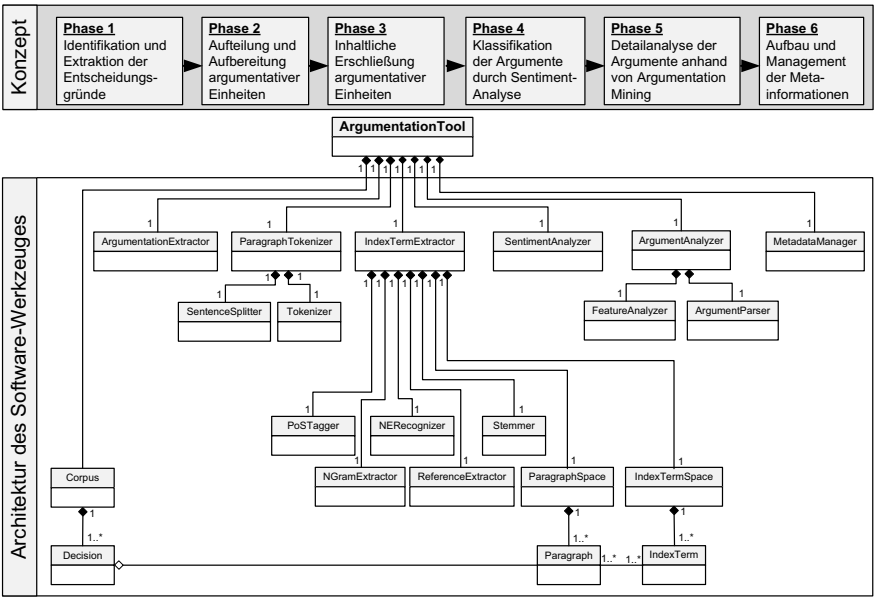

Abb. 2: Architekturentwurf des ARGUMENTUM Software-Werkzeuges

ArgumentationExtractor: Als Eingabe für dieses Modul dienen die ursprünglichen Entscheidungstexte des BVerfG im HTML-Format, die zunächst in eine reine Textrepräsentation überführt werden (*plain text*). Anschließend werden strukturelle Eigenschaften des Corpus genutzt, um die Entscheidungsgründe vom übrigen nicht argumentativen Text eines Dokumentes zu trennen. So weist z.B. die Überschrift mit der Zeichenfolge „Gründe:“ verlässlich darauf hin, dass sich die Entscheidungsgründe in der nachfolgenden Textpassage befinden.

ParagraphTokenizer: Die extrahierte Argumentation mit der Entscheidungsbegründung wird anschließend in einzelne argumentative Einheiten zerlegt. Eine solche Zerlegung kann hinsichtlich verschiedener Kriterien vorgenommen werden und stellt als Segmentierungsproblem eine Designentscheidung bei der Prototypentwicklung dar. Nach aktuellem Kenntnisstand kann davon ausgegangen werden, dass sowohl die Abschnittsnummerierungen als auch weitergehende Informationen aus der inhaltlichen Erschließung argumentativer Einheiten in Phase 3 für die Segmentierung genutzt werden können. Dies ist im Rahmen der Implementierung weiter zu verifizieren und zu verfeinern.

SentenceSplitter und *Tokenizer*: Als Voraussetzung für die Anwendbarkeit der nachfolgenden Methoden müssen die Inhalte der zu analysierenden argumentativen Einheiten als Aneinanderreihung kleinstmöglicher Bestandteile (sog. *Tokens*, i.d.R. einzelne Wörter) vorliegen. Argumentative Einheiten werden dazu zunächst in einzelne Sätze und anschließend in Tokens zerlegt. Im Ergebnis entsteht so eine Token-Darstellung jeder argumentativen Einheit, die als Eingabe für die folgenden Module dient.

IndexTermExtractor: Dieses übergeordnete Modul besteht aus sieben Einzelmodulen, die für die Extraktion bedeutungstragender Wörter zur Beschreibung einer argumentativen Einheit (Indexterme) sowie für die Generierung des Vektorraums zuständig sind und im Folgenden genauer beschrieben werden.

PoSTagger: Für jedes Token einer argumentativen Einheit können durch diese Methode die zugehörige Wortart und weitere relevante Informationen bestimmt werden, wie z. B. das Tempus eines Verbs. Diese Information dient u. a. als Eingabe für später ausgeführte Funktionen, wie die *N-Gram-Identifikation*, und wird insbesondere als Merkmal für die spätere Detailanalyse argumentativer Einheiten gespeichert.

NERecognizer: Auf Basis der Token-Darstellungen der argumentativen Einheiten werden in diesem Modul Eigennamen und Informationsobjekte wie Personen, Orte oder Organisationen (*Named Entities*, NE) identifiziert [JM09, S. 761ff.]. Diese NE sind als charakteristische, bedeutungstragende Wörter in einer argumentativen Einheit zu betrachten. Alle identifizierten NE dienen als Indexterme und werden beim Aufbau des Vektorraums berücksichtigt. Zu jeder NE wird die Häufigkeit ihres Auftretens in der jeweiligen argumentativen Einheit als Indikator ihrer inhaltlichen Bedeutung berechnet.

Stemmer: Das Stemming-Modul reduziert alle Wörter einer argumentativen Einheit auf ihre Grundform und abstrahiert somit z. B. von Flexionen und Pluralformen [Po75]. Beispielsweise werden die Begriffe „Anwältin“, „kommt“ und „Werbungen“ zu den Formen „anwalt“, „komm“ und „werbung“ reduziert. Dieses Vorgehen erweitert den Ergebnisraum für Suchanfragen, da es die konkrete Repräsentation eines Begriffes generalisiert und somit hilft, semantisch gleiche oder ähnliche Tokens zu identifizieren.

NGramExtractor: Auf Basis der grundformreduzierten Tokens nach dem Stemming extrahiert dieses Modul Mehrwortgruppen (sog. *N-Gramme*) und speichert sie inkl. der Häufigkeit ihres Auftretens pro argumentative Einheit. Zur Erkennung von *Unigrammen* ($n = 1$) und *Bigrammen* ($n = 2$) werden *Stop*-Wörter von der Betrachtung ausgeschlossen und nur Kombinationen der Wortarten *Substantiv*, *Verb*, *Adjektiv* und *Adverb* untersucht. Bei der Erkennung von *Trigrammen* ($n = 3$) kann das dritte Wort auch einer abweichenden Wortklasse angehören, beispielsweise soll die Wortgruppe „Zulässigkeit des Antrags“ identifiziert werden können, jedoch nicht die Kombination „die Zulässigkeit des“.

ReferenceExtractor: Die Entscheidungstexte des BVerfG-Corpus enthalten häufig Verweise auf Gesetzestexte oder Gerichtsentscheidungen, die ebenso als aussagekräftige Indexterme im Vektorraummodell verwendet werden können. Vor diesem Hintergrund extrahiert dieses Modul alle auftretenden Verweise in einer argumentativen Einheit und erzeugt es mögliche Variationen dieser Verweise durch Abstraktion von der konkreten Gliederungsstruktur, z. B. werden zusätzlich zu einem identifizierten Verweis wie „§ 40 Abs. 1 VwGO“ auch die Variationen „§ 40 VwGO“ und „VwGO“ erzeugt. Zwar können unterschiedliche Absätze innerhalb eines Gesetztesparagraphs inhaltlich voneinander abweichen, allerdings ist trotzdem davon auszugehen, dass dieses Vorgehen hilfreiche Informationen zur Bestimmung ähnlicher argumentativer Einheiten anhand ihres *gemeinsamen Bezuges* zu ausgewiesenen Gesetzesstellen oder Gerichtsentscheidungen ermöglicht.

IndexTermSpace und *ParagraphSpace*: Die Ergebnisse der vorherigen Analysemethoden, d. h. *Named Entities*, *N-Gramme* sowie Referenzen auf Gesetze und Entscheidungen, dienen zusammen mit der zugehörigen Auftretenshäufigkeit als Eingabe für diese beiden Module. Sie regeln für die interne Informationsverarbeitung das Verhältnis zwischen argumentativen Einheiten und Indextermen im Corpus. Dies geschieht durch eine eindeutige Zuordnung von Indextermen zu argumentativen Einheiten und vice versa. Schließlich wird für jeden Indexterm in jeder argumentativen Einheit die sogenannte *tf-idf*-Häufigkeit (*term frequency/inverse document frequency*) berechnet. Aus diesen Häufigkeiten der Indexterme pro argumentative Einheit wird ein Vektor aufgebaut, der eine semantische Repräsentation dieser Einheit darstellt. Zwischen den Vektoren verschiedener Einheiten können nun Ähnlichkeitsvergleiche mithilfe von Vektordistanzmaßen, z. B. der Cosinus-Ähnlichkeit, vorgenommen werden. Die Gesamtheit aller Vektoren bildet schließlich den Vektorraum über dem Corpus.

SentimentAnalyzer: Unter Verwendung einer regelbasierten Klassifikation wird innerhalb dieses Moduls zunächst bestimmt, ob eine argumentative Einheit *für* oder *gegen* einen Sachverhalt, d. h. eine vorgebrachte Klage, argumentiert. Anhand dieser Funktionalität soll anschließend auch eine Klassifikation größerer Argumentationsstränge, die sich aus mehreren argumentativen Einheiten zusammensetzen können, möglich sein.

ArgumentAnalyzer: Dieses Modul besteht aus zwei Submodulen (*FeatureAnalyzer* und *ArgumentParser*), welche die automatische Identifikation und Detailanalyse von Argumentationsstrukturen anhand der in den vorangegangenen Phasen erhobenen Informationen sowie anhand weiterer textimmanenter Merkmale ermöglichen sollen. Auf der Basis von Methoden des *Maschinellen Lernens* sollen Argumentationsschemata in den Entscheidungstexten identifiziert und kenntlich gemacht werden können.

MetadataManager: Im letzten Schritt des Phasenmodells werden die im Laufe der vorangegangenen Analyseschritte gewonnenen Metainformationen in strukturierter Form abgelegt und verwaltet, z. B. in einer XML-Datenstruktur, die einen schnellen Zugriff auf die Daten und somit eine gute Performanz des Software-Werkzeuges sicherstellen soll. Dies wird durch das Modul *MetadataManager* realisiert.

6 Stand der Implementierung des Software-Werkzeuges

Im Kontext des *Natural Language Processing* (kurz: NLP) existieren verschiedene Softwaresysteme zur automatisierten Verarbeitung natürlichsprachlicher Texte, die für eine Vielzahl von Sprachen vorgefertigte Analysemethoden und entsprechende Software-Module anbieten. Die Verwendung einer solchen Umgebung wurde während der Entwicklung des vorgestellten Software-Werkzeuges aus folgenden Gründen angestrebt: 1.) die dort vorimplementierten Methoden sind weitgehend etabliert und ausgereift, 2.) die aufwendige Annotation großer Textmengen zum Trainieren eigener Modelle für die verschiedenen Analysemodule kann entfallen und 3.) einige Umgebungen (z.B. *OpenNLP*) enthalten bereits Programme zur Evaluation der mitgelieferten Methoden. Um einen Überblick über bestehende NLP-Systeme zu erhalten und potentiell in Frage kommende Software zu identifizieren, wurde eine systematische Untersuchung durchgeführt.

Insgesamt konnten auf diese Weise 29 NLP-Systeme ausgemacht werden, die dann hinsichtlich der Kriterien Aktualität, Umfang der Dokumentation, Lizenz, verwendete Programmiersprache und Verfügbarkeit des Quellcodes bewertet wurden.

Nach Evaluation der identifizierten Softwaresysteme wurde die Verwendung des *OpenNLP*-Projektes der *Apache Software Foundation* als Basis für grundlegende NLP-Funktionalitäten entschieden.[2] OpenNLP bietet vorgefertigte Methoden für zahlreiche Grundfunktionalitäten der maschinellen Sprachverarbeitung. Für den Großteil dieser Methoden sind Modelle verfügbar, die mit deutschen Corpora trainiert wurden. Tabelle 1 zeigt die im aktuellen Stand der Implementierung verwendeten NLP-Softwarekomponenten getrennt nach den in Abb. 2 vorgestellten Modulen des Architekturentwurfs. Abb. 3 zeigt einen Screenshot des Prototyps.

Tabelle 1: Aktueller Stand der Implementierung der ARGUMENTUM Softwaremodule

Softwaremodul im Architekturentwurf	**Verwendete NLP-Softwarekomponenten**
ArgumentationExtractor	Jericho HTML Parser,[3] eigene Implementierung
ParagraphTokenizer	OpenNLP Sentence Detector, OpenNLP Tokenizer
PoSTagger	OpenNLP Part-of-Speech Tagger
NERecognizer	Stanford Named Entity Recognizer for German[4]
Stemmer	Snowball German Stemmer[5]
NGramExtractor / ReferenceExtractor / ParagraphSpace /IndexTermSpace	eigene Implementierung

Abb. 3: Aktuelle Bedienoberfläche des Softwareprototyps mit priorisierter Ergebnisliste

[2] http://opennlp.apache.org
[3] http://jerichohtml.sourceforge.net
[4] http://nlp.stanford.edu/software/CRF-NER.shtml
[5] http://snowball.tartarus.org/algorithms/german/stemmer.html

Zu den aktuell unterstützten Funktionen gehört die Formulierung von Suchanfragen über die grafische Bedienoberfläche und das anschließende Retrieval argumentativer Einheiten im Corpus, die semantisch ähnlich zu den eingegebenen Suchanfragen sind. Die Trefferliste im unteren Bereich der Anzeige ist nach Relevanz bezüglich der formulierten Suchanfrage geordnet, d. h. Urteile mit starker Ähnlichkeit stehen höher in der Liste als solche mit geringer Ähnlichkeit. Intern wird diese Relevanzbestimmung durch die Berechnung von Abständen zwischen den Indextermvektoren realisiert.

7 Resümee und Ausblick

Im vorliegenden Beitrag wurde die Konzeption und Entwicklung des ARGUMENTUM Software-Werkzeuges zur automatischen Identifikation und Analyse von Argumentationsstrukturen in BVerfG-Entscheidungen dargelegt. Hierzu wurde zunächst ein Phasenkonzept zur Aufbereitung des Entscheidungscorpus präsentiert, das anschließend in einem Architekturentwurf für ein Software-Werkzeug mündete. Auf Basis dieser Architektur wurde schließlich im Rahmen der Implementierung ein innovativer Softwareprototyp geschaffen, dessen aktuell verfügbarer Funktionsumfang vorgestellt wurde.

Die aktuelle Implementierung des Software-Werkzeuges stellt noch eine vergleichsweise frühe Entwicklungsstufe im Rahmen des iterativen Entwicklungsprozesses dar, die noch nicht über den vollen Funktionsumfang verfügt. Gleichwohl zeigt sie die konkrete Umsetzbarkeit der ersten drei Phasen des präsentierten Phasenkonzeptes zur Aufbereitung des BVerfG-Corpus und ermöglicht bereits ein Retrieval von inhaltlich ähnlichen Urteilsabschnitten zu einer beliebigen Suchanfrage. Im Rahmen der weiteren Arbeiten wird nicht nur die Umsetzung der übrigen Konzeptphasen angestrebt, sondern auch die iterative Weiterentwicklung und Verbesserung aller Funktionalitäten des Prototyps vorgenommen. Das als Grundlage verwendete OpenNLP-System verfügt über geeignete Softwarekomponenten für eine Umsetzung der vorgestellten Ideen zu den Konzeptphasen 4 und 5 bezüglich des Maschinellen Lernens, die in Zukunft umgesetzt werden sollen.

Im Rahmen der gewählten iterativen Vorgehensweise soll auch das Feedback der juristischen Anwendungspartner, die sich der Anwendung des Werkzeuges in mehreren Testphasen widmen werden, in die Weiterentwicklung des Prototyps einfließen. Darüber hinaus wird angestrebt, den getesteten Prototyp anhand geeigneter Methoden zu evaluieren. In diesem Zusammenhang bietet es sich an, Laborexperimente mit juristischen Anwendern des Prototyps durchzuführen und zu erheben, ob und inwiefern sich die Verwendung des Prototyps auf die Performanz des Auffindens passender Urteile und Argumentationsstrukturen im Hinblick auf Effektivität und Effizienz auswirkt. Zusätzlich ist unter dem Stichwort der *Generalisierung der Ergebnisse* zu eruieren, inwiefern sich die eingesetzten Techniken und die erreichte Retrieval-Leistung des Prototyps auch auf argumentative Texte anderer Domänen übertragen lassen. In jedem Fall existiert weiteres Forschungspotential.

Danksagung: Die vorgestellten Forschungsergebnisse wurden im Rahmen des Projektes „*Analyse und Synthese von Argumentationsstrukturen durch rechnergestützte Methoden am Beispiel der Rechtswissenschaft* (ARGUMENTUM)" erarbeitet, das durch das

Bundesministerium für Bildung und Forschung (BMBF) unter FKZ 01UG1237C gefördert wird. Die Autoren bedanken sich außerdem bei den ARGUMENTUM Projektpartnern für zahlreiche Hinweise sowie bei den drei anonymen Gutachtern für die konstruktiven Anmerkungen, die zur Verbesserung dieses Artikels beigetragen haben.

Literaturverzeichnis

[He04] Hevner, A. R.; March, S. T.; Park, J.; Ram, S.: Design Science in Information Systems Research. In: MIS Quarterly 28 (2004) 1, S. 75-105.

[Ho10] Hogenboom, A.; Hogenboom, F.; Kaymak, U.; Wouters, P.; de Jong, F.: Mining Economic Sentiment Using Argumentation Structures. In: J. Trujillo; G. Dobbie; H. Kangassalo; S. Hartmann; M. Kirchberg; M. Rossi; I. Reinhartz-Berger; E. Zimányi; F. Frasincar (Hrsg.): Advances in Conceptual Modeling – Applications and Challenges, LNCS 6413. Springer, Berlin, 2010, S. 200-209.

[Ho12] Houy, C., Fettke, P., Loos, P., Speiser, I., Herberger, M., Gass, A., Nortmann, U.: ARGUMENTUM - Towards computer-supported analysis, retrieval and synthesis of argumentation structures in humanities using the example of jurisprudence. In: Wölfl, S. (Hrsg.) KI-2012: Poster and Demo Track of the 35th German Conference on Artificial Intelligence, pp. 30-33. DFKI, Saarbrücken, Germany (2012).

[Ho13] Houy, C., Niesen, T., Fettke, P., Loos, P.: Towards Automated Identification and Analysis of Argumentation Structures in the Decision Corpus of the German Federal Constitutional Court. Proceedings of the 7th IEEE International Conference on Digital Ecosystems and Technologies (IEEE-DEST). Menlo Park, California, USA (2013)

[JM09] Jurafsky, D.; Martin, J. H.: Speech and Language Processing: an introduction to natural language processing, computational linguistics, and speech recognition. 2. Aufl., Pearson, Upper Saddle River, NJ 2009.

[Lu97] Ludwig, B.: Computerunterstützung der Argumentation in Gruppen. Wiesbaden 1997.

[MM11] Mochales, R.; Moens, M.-F.: Argumentation mining. In: Artificial Intelligence and Law 19 (2011) 1, S. 1-22.

[Mo07] Moens, M.-F.; Boiy, E.; Mochales-Palau, R.; Reed, C.: Automatic Detection of Arguments in Legal Texts. ICAIL '07 Proceedings of the 11th International Conference on Artificial intelligence and Law: 225 - 230. Stanford, California, USA 2007.

[Po75] Porter, M. F.: An Algorithm for suffix stripping. In: Program 14 (1980) 3, S. 130-137.

[RR04] Reed, C.; Rowe, G.: Araucaria: Software for argument analysis, diagramming and representation. In: International Journal on Artificial Intelligence Tools 13 (2004) 4, S. 961-979.

[Sc02] Scheer, A.-W.: ARIS – Vom Geschäftsprozeß zum Anwendungssystem. 4. Aufl., Springer, Berlin 2002.

[Sc10] Scheuer, O.; Loll, F.; Pinkwart, N.; McLaren, B. M.: Computer-Supported Argumentation: A Review of the State of the Art. In: International Journal of Computer-Supported Collaborative Learning 5 (2010) 1, S. 43-102.

[To75] Toulmin, S.: Der Gebrauch von Argumenten. Kronberg 1975.

[Wa96] Walton, D. N.: Argumentation Schemes for Presumptive Reasoning. Mahwah, NJ 1996.

[Wy10] Wyner, A.; Mochales-Palau, R.; Moens, M.-F.; Milward, D.: Approaches to Text Mining Arguments from Legal Cases. In: E. Francesconi; S. Montemagni; W. Peters; D. Tiscornia (Hrsg.): Semantic Processing of Legal Texts, LNCS 6036. Springer, Berlin 2010, S. 60-79.