Charakterisierung von Toponymverwendungen in Texten

Characterization of Toponym Usages in Texts

Masterarbeit

Master's Thesis

im Virtuellen Weiterbildungsstudiengang Wirtschaftsinformatik
an der
Otto-Friedrich-Universität Bamberg

vorgelegt von

Sebastian Johannes Wolf

angefertigt am

Lehrstuhl für Medieninformatik
Otto-Friedrich-Universität Bamberg

Prüfer: Prof. Dr. Andreas Henrich

Beginn der Arbeit: 11.11.2013
Abgabe der Arbeit: 05.05.2014

Kurzfassung

Viele Menschen nutzen für Informationsbedürfnisse - wie die Suche nach aktuellen Nachrichten, Kochrezepten, aber auch nach wissenschaftlichen Texten - nicht mehr die Tageszeitung, das Fernsehen oder Lexika, sondern das Internet als primäre Informationsquelle. Suchmaschinen wie Google oder Bing gehören deshalb seit vielen Jahren zum Alltag. Die entsprechenden Suchanfragen sind allerdings in vielen Fällen nur äußerst kurz und in natürlicher Sprache gehalten, was dazu führt, dass die Dienste den Hintergrund des Informationsbedürfnisses entsprechend interpretieren müssen, um die für den Nutzer relevanten Informationen ausgeben zu können.

Diverse Untersuchungen haben gezeigt, dass die Suchanfragen in vielen Fällen einen geographischen Bezug haben, etwa um einen Pizza-Lieferdienst in der näheren Umgebung zu finden, oder eben detaillierte Informationen über das nächste Urlaubsziel zu erhalten. Um die passenden Inhalte auswählen zu können, muss die Suchmaschine, oder allgemein das (geographische) Informationssystem, sowohl Ortsbezüge in der Anfrage als auch im eigenen Datenbestand korrekt interpretieren können. Ein Ortsbezug kann hierbei implizit, etwa über den Standort des Nutzers bzw. dem Entstehungsort eines Textes, oder explizit über die konkrete Nennung von Orten, sogenannter Toponyme, in der Anfrage oder in Texten hergestellt werden.

Verschiedene Herausforderungen ist hierbei nicht nur die bloße Erkennung von Toponymen, also der Unterscheidung von Ortsbezügen und anderen Worten, sondern auch bei deren Auflösung, also der Zuweisung von konkreten Koordinaten zu einem Toponym. Allerdings sind nicht nur diese beiden Teilgebiete, sondern auch die *Charakterisierung von Toponymverwendungen* Gegenstand verschiedener Untersuchungen. So können zum Beispiel die Verwendung als Referenz oder als Attribut, oder aber eine wörtliche Verwendung von einer im übertragenen Sinne unterschieden werden. Dies ist in vielen Fällen zur Ermittlung des Informationsbedürfnisses der Nutzer von zentraler Bedeutung.

Während sich andere Arbeiten bislang auf englische Texte konzentriert haben, beschäftigt sich die vorliegende Arbeit nun mit dieser Toponymcharakterisierung in Texten deutscher Sprache. Hierzu werden zunächst die Grundlagen des geographischen Information Retrievals kurz erläutert, sowie die unterschiedlichen sprachlichen Stilmittel voneinander abgegrenzt und entsprechend definiert. Es folgen Überblicke über existierende Werkzeuge zur Textanalyse im Bereich der Computerlinguistik sowie zu verwandten Arbeiten im Bereich der Toponymerkennung, -auflösung und -charakterisierung.

Im Hauptteil der Arbeit wird beschrieben, wie das Textanalyseframework GATE (General Architecture for Text Engineering) um weitere existierende Komponenten zur Satzanalyse und Toponymerkennung erweitert wird, welche dann zusammen die Grundlage für die Erkennung von Verwendungsarten und Stilmitteln bilden. Hierzu wurde das GATE-Plugin YGRIEGA (YGRIEGA Georeferenced Rhetorical figures Identifier, Extractor and Grammar Analyzer) entwickelt, welches entsprechende Fundstellen anhand von Annotationen markiert und dessen Architektur und Implementierung im Detail vorgestellt wird. Die mit den GATE-Erweiterungen und dem neuen Plugin erzielten Ergebnisse bei der Toponymcharakterisierung werden dann anhand eines selbst annotierten Referenzkorpus mit Texten aus verschiedenen Genres evaluiert und miteinander in Beziehung gesetzt. Den Abschluss bilden eine Zusammenfassung der erreichten Ergebnisse mit einem Ausblick auf mögliche weitere Arbeiten im Kontext dieses Themengebiets.

Abstract

Many people do no longer use newspapers, television or encyclopedias as their primary source in order to fulfill their information needs. Instead, they surf the internet searching for news, cooking recipes or scientific papers. Search engines like Google or Bing have therefore become a vital part of people's lives over the last years. However, search requests often consist of just a few words in natural language. The problem hereby is that the used services need to interpret the background of the information need in order to return relevant information to the user.

Several studies showed that these search requests often contain a geographical reference, for example when users want to find the nearest pizza delivery service or to get detailed information about the next vacation destination. In order to return the right content, the search engine, or more in general the geographic information system, must be able to correctly interpret geographical references both in the search request and in its own database. Such a reference can be stated implicitly, e.g. by the user's current location or the one where the text was written, and explicitly, by using concrete location names, so-called toponyms, both in the request and in respective texts.

Different challenges arise not only in the task of pure toponym detection, i.e. to differentiate between location references and other words. Also the resolution of toponyms, that is the allocation of concrete coordinates to toponyms, is a topic of research. However, not only those two areas constitute a central part of several studies, but also the *characterization of toponym usages*. For example you can distinguish between toponym usage as a reference and as an attribute. Moreover, a location reference can be meant literally or figuratively. The detection of such characteristics is in many examples crucial in order to detect the user's information needs.

This thesis discusses that toponym characterization in texts written in German, whereas other authors so far have concentrated on English texts. At the beginning the thesis outlines the fundamentals of geographic information retrieval, which is followed by a detailed description of the different rhetorical figures that are relevant in this context. Afterwards, this thesis will give an overview over already existing tools for text analyses in the area of computational linguistics and related work in the areas of toponym detection, resolution and characterization.

The main part of the thesis consists of a description how the text engineering framework GATE (General Architecture for Text Engineering) was enhanced by several already existing components for sentence analysis and toponym detection. These tools then constitute the foundation for the detection of toponym usage kinds and rhetorical figures. The newly developed GATE plugin YGRIEGA (YGRIEGA Georeferenced Rhetorical figures Identifier, Extractor and Grammar Analyzer) marks respective findings with annotations. The architecture and implementation of this plugin will be outlined in detail. The results, that have been achieved with both the GATE enhancements and the new plugin, are evaluated against a manually annotated reference corpus that consists of texts from different genres. Finally, the conclusion summarizes the accomplishments of this work as well as an outlook on possible future enhancements and new developments in the context of this topic.

Inhaltsverzeichnis

Abbildungsverzeichnis iv

Tabellenverzeichnis vi

1 Einführung 1
1.1 Was sucht der Nutzer einer Suchmaschine? . . . 1
1.2 Geographisches Information Retrieval . . . 3
1.2.1 Begriffserklärungen . . . 3
1.2.2 Besondere Herausforderungen . . . 4
1.3 Gegenstand und Aufbau der Arbeit . . . 6

2 Grundlagen und Kontext 7
2.1 Sprachliche Stilmittel . . . 7
2.1.1 Verwendungsarten . . . 7
2.1.2 Figuren . . . 9
2.1.3 Tropen . . . 11
2.2 Textanalyse . . . 13
2.2.1 Einführung . . . 13
2.2.2 Analysewerkzeuge . . . 14
2.3 Verwandte Arbeiten . . . 20
2.3.1 Allgemeines zum Geographischen Information Retrieval . . . 20
2.3.2 Toponymerkennung und -auflösung . . . 21
2.3.3 Toponymcharakterisierung . . . 22

3 Toponymerkennung und Satzanalyse 24
3.1 Technische Infrastruktur . . . 24
3.1.1 Grundlagen . . . 24
3.1.2 Integration abhängiger Komponenten . . . 24
3.1.3 Vor- und Nachteile der bestehenden Komponenten . . . 27
3.2 Weiterentwicklung der Infrastruktur . . . 28
3.2.1 Integration von ParZu . . . 28
3.2.2 Integration des Stanford NER Taggers . . . 31
3.2.3 Erweiterung des GATE Gazetteers . . . 33
3.2.4 Toponymerkennung mit Geonames.org . . . 38

4 Charakterisierung von Toponymverwendungen 43
4.1 Einführung - Integration in GATE . . . 43

4.2 Architektur . . . 45
4.2.1 Grundlagen . . . 45
4.2.2 Implementierung . . . 46
4.3 Umsetzung . . . 51
4.3.1 Verwendungsarten . . . 51
4.3.2 Figuren . . . 56
4.3.3 Tropen . . . 62

5 Evaluation **70**
5.1 Vorgehensweise . . . 70
5.2 Ergebnisse . . . 72
5.2.1 Toponymerkennung . . . 72
5.2.2 Toponymcharakterisierung . . . 76

6 Zusammenfassung und Ausblick **79**
6.1 Ergebnisse der Arbeit . . . 79
6.2 Ausblick . . . 81

Literaturverzeichnis **84**

Anhang **90**
Texte des Referenzkorpus . . . 90

Eidesstattliche Erklärung **92**

Abbildungsverzeichnis

1.1 Suchanfrage bei google.de nach „Essen“ 2

2.1 TreeTagger-Ausgabe für den Beispielsatz 15
2.2 ParZu-Ausgabe für den Beispielsatz 16
2.3 Grafische Ausgabe von ParZu für den Beispielsatz 16
2.4 Ausgabe von GernEdiT für *beginnen* 17
2.5 Ausgabe des Stanford NER mit dem deWaC-Klassifizierer 18
2.6 Ausgabe von GATE Developer für den Beispielsatz 19

3.1 Annotationen des MuNPEx-Plugins für Nominalphrasen des Beispielsatzes 26
3.2 Grafische Ausgabe des Stanford NLP für den Beispielsatz 28
3.3 Quelltext des Shellskripts `parzu-gate` 30
3.4 Konfigurierte GenericTagger Processing Resource für ParZu 30
3.5 Features einer Token-Annotation nach erfolgreicher ParZu-Integration 31
3.6 Quelltext des Shellskripts `stanford_ner-gate` 32
3.7 Konfigurierte GenericTagger Processing Resource für Stanford NER 33
3.8 Features einer Location-Annotation nach erfolgreicher Stanford NER-Integration 33
3.9 Sequenzdiagramm des Programms `OpenGeoDBExtractor` 36
3.10 Quelltext des JAPE-Regel für OpenGeoDB-Einträge 37
3.11 `Location`-Annotation für einen mittels OpenGeoDB erkannten Ort 37
3.12 Pseudocode der Ermittlung der GeoNames-Suchanfrage 39
3.13 Sequenzdiagramm der Processing Resource Geonames.org Toponym Resolver . . 40
3.14 Inhalt der `creole.xml`-Datei für die Geonames.org Toponym Resolver PR 41
3.15 Integration der Geonames.org Toponym Resolver PR in GATE 42
3.16 `Location`-Annotation für einen mittels geonames.org erkannten Ort 42

4.1 Inhalt der `creole.xml`-Datei für beide YGRIEGA Processing Resources 44
4.2 Der Toponym Analyzer in einer GATE Anwendungspipeline 44
4.3 Blackboard-Architektur des Toponym Analyzers 46
4.4 Klassendiagramm der Toponym Analyzer-Basiselemente 48
4.5 Klassendiagramm der Blackboard-Architektur 49
4.6 Ausführungsreihenfolge der Wissensquellen im Toponym Analyzer 50
4.7 Pseudocode des Algorithmus der Verwendungsart *Referenz* mit Prä-/Zirkumposition 52
4.8 Erfolgreiches Auffinden der Verwendungsart *Referenz* im Beispielsatz 53
4.9 Beispiele für die Verwendungsart *Attribut* mit Zwischenworten 54
4.10 Pseudocode des Algorithmus der Verwendungsart *Attribut* mit Zwischenworten . 54
4.11 Ausgabe von ParZu für ein Toponym mit modifizierendem Attribut 55
4.12 Ausgabe von GATE für ein Toponym mit modifizierendem Attribut 56

4.13 Pseudocode des Algorithmus zur Erkennung einer Alliteration 57
4.14 Pseudocode des Algorithmus zur Erkennung einer Geminatio 58
4.15 Pseudocode des Algorithmus zur Erkennung einer Anapher 60
4.16 Pseudocode des Algorithmus zur Erkennung einer Epipher 61
4.17 Pseudocode des Algorithmus zur Erkennung einer Anadiplosis 63
4.18 Ausgabe von ParZu für einen Vergleich 64
4.19 Pseudocode des Algorithmus zur Erkennung eines Vergleichs 64
4.20 Pseudocode des Algorithmus zur Erkennung einer Synekdoché für Gesetzestexte 64
4.21 Pseudocode des Algorithmus zur Erkennung einer Synekdoché auf Basis des Prädikats 66
4.22 Quelltext der JAPE-Regel für Antonomasien 67
4.23 Quelltext der JAPE-Regel für Allegorien 68
4.24 Quelltext der JAPE-Regel für Metaphern 69

5.1 Toponymerkennungsraten des Textes *putin* 74
5.2 Toponymerkennungsraten des Textes *uncle-sam* 74
5.3 Toponymerkennungsraten des Textes *island* 75
5.4 Toponymerkennungsraten des gesamten Evaluationskorpus 76

Tabellenverzeichnis

5.1 Anzahl der Texte mit höchstem F-Measure je Anwendungspipeline 73
5.2 Ergebnisse der Toponymerkennung des gesamten Evaluationskorpus 76
5.3 Ergebnisse der Toponymcharakterisierung bei den Verwendungsarten 77
5.4 Ergebnisse der Toponymcharakterisierung bei den Stilmitteln 77

1 Einführung

1.1 Was sucht der Nutzer einer Suchmaschine?

Suchmaschinen wie Google[1], Bing[2] oder DuckDuckGo[3] sind schon seit vielen Jahren nicht mehr aus dem Leben vieler Menschen wegzudenken. Internet-Nutzer steuern diese Dienste für viele Informationsbedürfnisse an. Sie finden so unter anderem aktuelle Nachrichten, Kochrezepte, Reiseberichte, aber auch wissenschaftliche Texte. Das einer Anfrage zugrundeliegende Informationsbedürfnis ist allerdings nur in wenigen Fällen offensichtlich und zweifelsfrei interpretierbar. Denn im Gegensatz zu einer Anfrage an ein Datenbanksystem, für die man zur Ermittlung des Datensatzes eines Kunden konkrete Informationen wie die Kundennummer oder eine Adresse benötigt, beinhalten die textuellen Anfragen an Suchmaschinen oft wenige Informationen darüber, was der Nutzer wirklich wissen möchte. So könnte die Suchanfrage „Essen“ einerseits einem akuten Hungergefühl des Nutzers entspringen, weswegen er auf der Suche nach einem Schnellimbiss in der Umgebung ist. Andererseits könnte er auch auf der Suche nach Informationen über die Stadt Essen sein, etwa nach Reiseberichten oder Stadtführern. Solche Einwortanfragen sind trotz ihres Mangels an Genauigkeit in Suchanfragen weit verbreitet [CMS10].

Diese kurzen Suchanfragen müssen allerdings nicht unbedingt der Bequemlichkeit der Nutzer geschuldet sein. Die Ursache liegt in vielen Fällen darin, dass es für Menschen schwierig ist, ein Informationsbedürfnis zu formulieren, gerade weil das Gesuchte ihre Wissenslücke darstellt [CMS10]. Diesen Zustand haben Belkin et al. bereits 1982 als *Anomalous State of Knowledge* bezeichnet, den eine Suchmaschine beheben soll. Hierbei sei es für einen Dienst zum Zweck der Bearbeitung einer Suchanfrage geeigneter, den Versuch zu unternehmen, dieses Wissensdefizit zu beschreiben, als den Nutzer darum zu bitten, die Anfrage zu konkretisieren [BOB82].

Das Informationsbedürfnis hinter einer entsprechenden Suchanfrage hat in vielen Fällen einen geographischen Hintergrund. Dies gilt ebenfalls für die bereits angesprochenen Beispiele, sei es, weil der Nutzer im ersten Fall kaum an Essensmöglichkeiten am anderen Ende der Welt interessiert sein dürfte, oder weil im zweiten Fall Informationen zu einer Stadt einen konkreten Ortsbezug haben. Suchmaschinen wie Google adressieren dieses geographische Informationsbedürfnis in vielen Fällen, indem sie ortsbezogene Informationen in den Ergebnissen besonders hervorheben (vgl. Abbildung 1.1 auf der nächsten Seite). Diese Fokussierung kommt hierbei nicht von ungefähr. Verschiedene Untersuchungen haben ergeben, dass sich der Anteil der Suchanfragen mit geographischem Bezug mindestens im zweistelligen Bereich bewegt. Jones et al. sprechen von 12,7% der Anfragen mit „Geo-Spezifikation“ [JZR08], Sanderson und Kohler von 18,6% der Anfragen mit einem „geographischen Term“ [SK04]. Weitere Forscher haben zudem her-

[1] `https://www.google.de/`
[2] `http://www.bing.com/`
[3] `https://duckduckgo.com/`

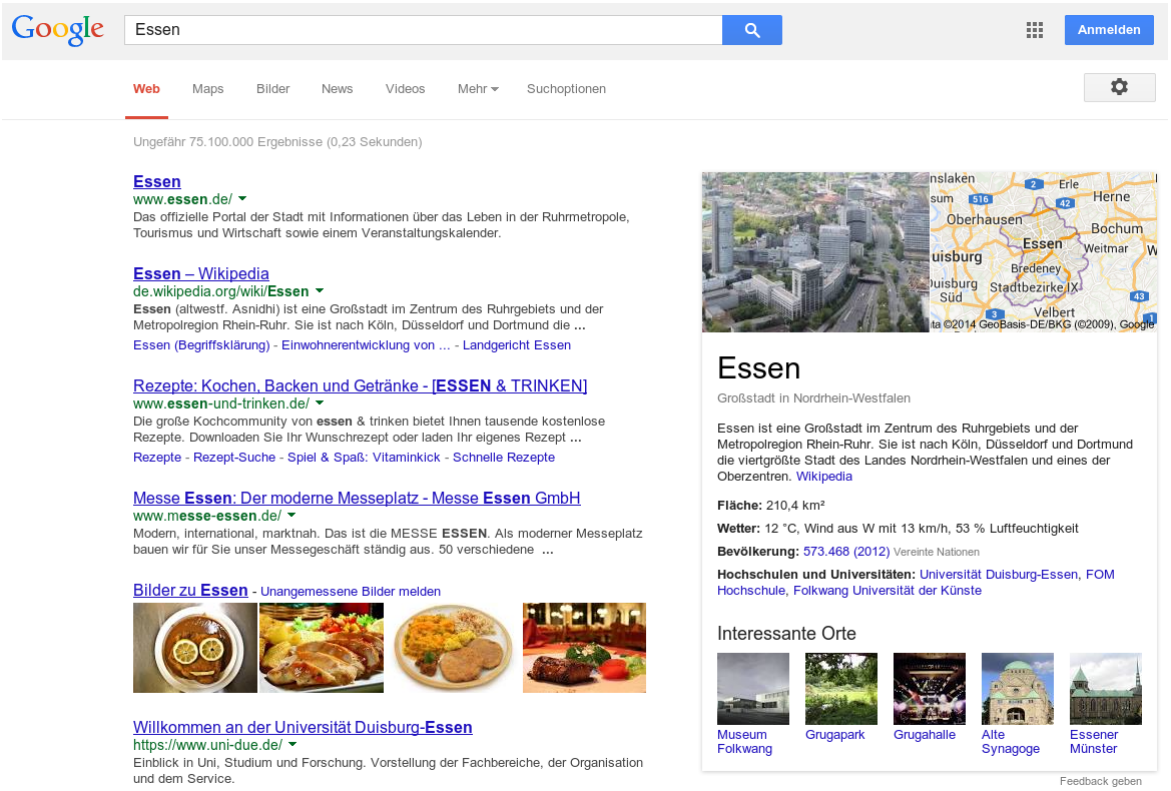

Abbildung 1.1 — Suchanfrage bei google.de nach „Essen“

ausgefunden, dass Nutzer nur bei weniger als der Hälfte der Suchanfragen mit geographischem Bezug auch explizite geographische Suchworte verwenden [WC08]. Die Erkennung dieser geographischen Bezüge dient dabei nicht nur dem Nutzer, weil er so bessere Suchergebnisse erhält, sondern auch den Anbietern von Suchmaschinen, weil diese so zielgerichteter Werbung platzieren können [YRL09].

Allerdings verbergen sich nicht nur bei der Interpretation der Suchanfragen mit Hinblick auf geographische Referenzen besondere Herausforderungen. Auch die Einordnung und Georeferenzierung der zu durchsuchenden Inhalte ist in vielen Fällen schwierig. Wurde von der Suchmaschine beispielsweise festgestellt, dass nach Reiseberichten zu einer bestimmten Region gesucht wurde, sind etwa Informationen über die wirtschaftliche Entwicklung dieser Region eher weniger relevant. Das *Genre* eines Textes oder die Textsorte (vgl. [Mic01]) ist somit neben dem Ortsbezug eine weitere Dimension, die es zu beachten gilt. Wenn der Text konkrete Hinweise wie Überschriften mit „Reiseführer“, „Reisebericht“ o.ä. enthält, ist die Textsorte noch verhältnismäßig einfach zu identifizieren. In anderen Fällen, etwa bei Texten, in denen touristisch relevante Informationen eher beiläufig erwähnt werden, oder bei Posts in sozialen Netzwerken, die weniger einem klassischen Aufbau eines journalistischen Berichts folgen, ist diese Identifikation schon deutlich anspruchsvoller.

Um in solchen Fällen die Textsorte zu ermitteln, können mehrere Quellen herangezogen werden. So ist etwa ein Gedicht allein durch die Satzstruktur schnell von Prosatexten zu unterscheiden.

Auch die Verwendung von bestimmten Zeitformen kann auf ein bestimmtes Genre hindeuten, weil z.B. in Reiseberichten in der Regel eher in der Vergangenheit gesprochen wird, aber in den Wahlprogrammen politischer Parteien das Augenmerk auch auf zukünftige Entwicklungen gelenkt wird. Weitere zentrale Elemente eines Textes sind die verwendeten Stilmittel, die sich auf verschiedene Strukturebenen, wie etwa auf die Textgliederung, den Satzbau oder die Kommunikationsform erstrecken. Sie können zwar nicht alleine als charakteristische Merkmale zur Bestimmung der Textsorte dienen, also etwa um künstlerische von nicht-künstlerischen Texten voneinander zu unterscheiden, allerdings bieten sie starke Anhaltspunkte hierfür [Mic01].

Eine Erkennung der in Texten verwendeten Stilmittel im Zusammenhang mit konkreten Ortsbezügen kann so dazu beitragen, die Informationsbedürfnisse der Nutzer besser zu befriedigen. Die vorliegende Masterarbeit wird sich im Folgenden damit befassen, die angesprochenen Eigenschaften von Texten miteinander in Beziehung zu setzen und konkrete Schritte zur *Charakterisierung von Ortsbezügen in Texten* zu entwickeln.

1.2 Geographisches Information Retrieval

1.2.1 Begriffserklärungen

Information Retrieval

Bei der thematischen Erläuterung war bisher ausschließlich von Suchmaschinen die Rede. Diese stellen jedoch nur die praktische Anwendung von Techniken aus der Disziplin des *Information Retrieval* mit Hinblick auf große Textsammlungen dar [CMS10]. Beim Information Retrieval handelt es sich nach Gerard Salton um das gesamte Fachgebiet, welches sich „mit der Struktur, der Analyse, Organisation, Speicherung, der Suche und dem Auffinden von Information“ beschäftigt [Sal68]. Im weiteren Verlauf dieser Arbeit wird deshalb in der Regel allgemein auf Konzepte und Hintergründe des Information Retrieval - meistens mit geographischem Bezug - eingegangen und nicht notwendigerweise auf ihre konkrete Umsetzung in der Anwendung *Suchmaschine*.

Toponyme

Wie bereits erläutert, kann sich ein Ortsbezug einer Suchanfrage oder eines Textes auch durch einen impliziten Kontext ergeben. Wenn jedoch explizit ein Ort genannt oder referenziert wird, so wird dieser im wissenschaftlichen Diskurs in der Regel als *Toponym* bezeichnet. Ausgehend von der Definition des Arbeitskreises Terminologie der Sachverständigengruppe der Vereinten Nationen für geographische Namen UNGEGN versteht man unter einem Toponym einen „Eigennamen für ein topographisches Objekt“. Es ist dabei ein „Oberbegriff für geographische [...] und extraterrestrische Namen“. Ein topographisches Objekt kann dabei ein „Teil der Oberfläche der Erde oder jedes anderen Planeten oder Satelliten, der eine erkennbare Identität besitzt“, darstellen. Mit geographischen bzw. extraterrestrischen Namen werden dabei die entsprechenden Objekte auf der Erdoberfläche bzw. auf anderen Planeten oder Satelliten bezeichnet [Arb10].

Toponymerkennung und -auflösung

Mit Hinblick auf die Erkennung und Auflösung von Toponymen orientiert sich diese Masterarbeit an der Dissertation von Volker Lüdecke (vgl. [LÜ10]). Er bezeichnet die Toponymerkennung als „Vorgang, aus einem Text oder einer gegebenen Menge von Wörtern diejenigen Wörter oder Wortgruppen zu identifizieren, die zu der Klasse der Toponyme gehören“. Ein weiterer Arbeitsschritt stellt dann die Toponymauflösung dar, worunter der Vorgang fällt, „für ein Toponym, das im Rahmen einer Toponymerkennung identifiziert werden konnte, aus einer Menge von topographischen Objekten, die mit demselben Begriff bezeichnet werden, dasjenige Objekt zu ermitteln, das vom Verfasser des Textes bzw. der Menge von Wörtern zum Ausdruck gebracht werden soll“. Technisch gesehen verbirgt sich hinter der Toponymauflösung also das Zuordnen von geographischen Koordinaten zu den gefundenen Toponymen.

Toponymcharakterisierung

Wie in der Motivation zu dieser Arbeit erläutert, besteht die besondere Herausforderung nun darin, die Ortsbezüge in Texten zu charakterisieren. Dieser Vorgang wird analog zu den anderen Begriffen als Toponymcharakterisierung bezeichnet und umfasst den Prozess, die Verwendungsart eines Toponyms, das im Rahmen der Toponymerkennung identifiziert werden konnte, sowie eventuelle Stilmittel in dessen Kontext zu bestimmen. Somit kann hier auch von der *Charakterisierung von Toponymverwendungen* gesprochen werden. Toponymauflösung und -charakterisierung schließen somit beide an die Toponymerkennung an, können allerdings vollkommen unabhängig voneinander betrachtet und durchgeführt werden.

1.2.2 Besondere Herausforderungen

Zwar folgt natürliche Sprache weitgehend gewissen Regeln und Normen, allerdings ist sie etwa im Gegensatz zu Programmiersprachen auch geprägt von Ausnahmen und Uneindeutigkeiten - sowohl in den verwendeten Wörtern, der Syntax, als auch in der Grammatik und Semantik. Da die Erkennung, Auflösung und Charakterisierung von Toponymen in Texten erfolgt, die in natürlicher Sprache verfasst sind, müssen in den einzelnen Analyseschritten besondere Herausforderungen bewältigt werden.

Bei der *Toponymerkennung* besteht die zentrale Problematik darin, echte Ortsbezüge zu erkennen, wenn das fragliche Wort sowohl ein Toponym als auch etwas anderes bezeichnen kann. Man spricht hier von einer „Geo/Non-Geo-Ambiguität“ [AHSS04]. Diese sind zum Teil höchst sprachspezifisch. So kann die Verwendung des Worts *Reading* in einem englischen Text eine Anwendung durchaus vor größere Probleme stellen, v.a. wenn es am Anfang eines Satzes steht und so wegen der Großschreibung sowohl das *Lesen* als auch eine Stadt im Vereinigten Königreich bezeichnen kann. In einem deutschen Text fällt zudem die Unterscheidung zwischen Eigennamen und anderen Substantiven mit Hinblick auf ihre Groß-/Kleinschreibung weg. Hier stellen etwa die Verwendungen der Wörter *Norden* (Himmelsrichtung und Stadt in Niedersachsen) oder *Acht* (Zahl und Ort in Rheinland-Pfalz) an jeder Stelle in einem Satz eine solche Ambiguität dar. Weiterhin haben bestimmte Orte oft unterschiedliche Namen in verschiedenen Sprachen, diese

werden dann als *Endonyme* oder *Exonyme* bezeichnet [LÏ0]. Ein prominentes Beispiel hierfür ist die Stadt *Aachen*, welche auf Französisch *Aix-la-Chapelle*, auf Niederländisch *Aken* und auf Spanisch *Aquisgrán* genannt wird.

Wenn ein bereits erkanntes Toponym lokalisiert werden soll, können zudem weitere Probleme auftreten. Bei dieser *Toponymauflösung* sind beispielsweise sogenannte „Geo/Geo-Ambiguitäten“ zu beachten [AHSS04]. Speziell die Namen von größeren Städten in Europa treten oft in ähnlicher oder gleicher Form auch bei Siedlungen in Amerika auf, so etwa Berlin, London oder Paris. Lüdecke hat in seiner Dissertation beide Arten von Ambiguitäten zu der Kategorie der *Homonyme* zusammengefasst und hier auch in einer Untersuchung mit einer deutschen Vokabelliste herausgefunden, dass je nach Zählweise 2,1% oder 4,4% der Einträge im verwendeten Wörterbuch sowohl einen Ort als auch etwas anderes bezeichnen können [LÏ0].

Lüdecke nennt zudem weitere Problemklassen, die in den verschiedenen Bereichen eine Rolle spielen können. So ist etwa bei der Toponymauflösung auf Mehrwortgruppen zu achten, die zudem korrekt zu bilden sind. Bei der Lokalisierung des Ortes *Frankfurt* sind zum Beispiel oft die folgenden Worte im Text von Bedeutung, um zweifelsfrei festzustellen, ob sich der Autor auf die Stadt am Main oder an der Oder bezieht. Allerdings kann diese Unterscheidung durch die vielen möglichen Schreibweisen wie *Frankfurt/Main* oder *Frankfurt am Main* erschwert werden [LÏ0]. In bestimmten Fällen können diese Mehrwortgruppen zudem auch eine Rolle bei der Toponymerkennung spielen. Das Wort *Singen* kann für sich alleine beispielsweise sowohl die Tätigkeit als auch verschiedene Orte darstellen. Erst wenn es etwa mit *Singen (Hohentwiel)* konkretisiert wird, ist sowohl die Erkennung als auch die Auflösung leichter möglich.

Weiterhin sind laut Lüdecke in bestimmten Fällen auch Abwandlungen der offiziellen Schreibweise von Orten zu berücksichtigen. So seien in Wörterbüchern nicht zwangsweise die verschiedenen Flexionsformen gelistet, zudem können speziell in nicht-redaktionellen Texten wie bei Inhalten aus dem Internet Schreibfehler eine wichtige Rolle bei der Erkennung spielen. Als letzte Kategorie betrachtet Lüdecke mögliche Veränderungen von Ortsnamen über die Zeit. Hier seien vor allem Eingliederungen und Zusammenschlüsse sowie die Änderungen der offiziellen Schreibweise zu beachten [LÏ0]. Beispiele für solche Änderungen sind etwa der Zusammenschluss der Städte Elberfeld und Barmen zu Wuppertal oder der Namenswechsel der Stadt Chemnitz, die von 1953 bis 1990 Karl-Marx-Stadt hieß.

Andere Autoren betonen auch noch weitere Herausforderungen, speziell bei der Toponymauflösung. So verwenden viele Autoren vage Ortsangaben wie „nördlich von Paris“ oder „in der Nähe von Bamberg“. Die Bedeutung solcher Angaben hängen sehr stark vom jeweiligen Kontext und der Wahrnehmung von Distanz seitens des Autors ab [BGP12] und sind so nur schwierig einzuordnen.

Die Verwendung von Ortsbezügen im Rahmen von Stilmitteln spielt vor allem bei der *Toponymcharakterisierung* eine zentrale Rolle. Beispiele für solche Stilmittel sind Allegorien wie *Uncle Sam* für die USA oder Metaphern wie die *Wiege der Menschheit* für Ostafrika. Vergangene Arbeiten etwa von Katja Markert und anderen haben sich in diesem Bereich speziell mit der Metonymie beschäftigt (vgl. u.a. [MH02]). Hierbei wird in der Regel ein Begriff bezeichnet, der durch ein Wort ersetzt wird, welches mit dem Begriff z.B. räumlich oder logisch in Zusammenhang steht [KRS02]. Da sich diese Masterarbeit im Besonderen mit diesem Themenbereich befasst, werden Stilmittel im Grundlagenkapitel 2.1 nochmals ausführlich behandelt. Diese Erläuterungen stel-

len dann die Basis für die Einordnung von verwandten Arbeiten und die konkrete Ausarbeitung der *Charakterisierung von Toponymverwendungen in Texten* in den folgenden Kapiteln dieser Arbeit dar.

1.3 Gegenstand und Aufbau der Arbeit

Im Abschnitt 1.1 wurde bereits dargelegt, dass diese Arbeit dazu beitragen soll, die Informationsbedürfnisse der Nutzer besser zu befriedigen, indem Toponymverwendungen in Texten charakterisiert werden. Diese Charakterisierung setzt, wie in Abschnitt 1.2.1 erläutert, auf der Toponymerkennung auf, ist allerdings unabhängig von der Toponymauflösung. Deshalb werden im Folgenden verstärkt die Themenbereiche Erkennung und Charakterisierung, jedoch nicht die Auflösung von Toponymen betrachtet. Bei der Erkennung wird hierbei so weit wie möglich auf existierende Grundlagen und Techniken aufgesetzt und diese nur dann durch eigene Beiträge ergänzt, wenn dies zur Schaffung einer besseren Grundlage für die Charakterisierung von Toponymverwendungen geboten ist. Der Hauptfokus liegt auf der Definition, Abgrenzung und Erkennung der verschiedenen Charakteristika von Toponymverwendungen. Die Erkennung erfolgt mittels einer passenden Software, welche im Detail erläutert wird. Zum Ende der Arbeit werden die Funktion dieses Programms auf Basis von Referenztexten analysiert und die Ergebnisse dargestellt. Die Arbeit beschränkt sich hierbei ausschließlich auf die deutsche Sprache.

In Kapitel 2 werden zunächst die Grundlagen zu sprachlichen Stilmitteln beschrieben. Im Anschluss folgt ein Überblick über verfügbare Analysewerkzeuge für natürliche Sprache, die im Kontext dieser Arbeit von Bedeutung sind. Darauf aufbauend folgen Informationen über verwandte Vorgängerarbeiten. Kapitel 3 widmet sich der Toponymerkennung und der Satzanalyse, welche die Grundlagen einer korrekten Toponymcharakterisierung bilden. Die technische Infrastruktur für die entsprechende Implementierung wird hier näher beleuchtet. Hinzu kommen Details zur Integration abhängiger Komponenten und die Verbesserung dieser Infrastruktur. Die Umsetzung der Charakterisierung von Toponymverwendungen folgt dann in Kapitel 4. Hierbei wird die Architektur des Analyseprogramms beschrieben, sowie die Details zur Erkennung der einzelnen Stilmittel erläutert. In Kapitel 5 werden die Ergebnisse dieser Charakterisierung evaluiert. Hierzu wird eine kleine Kollektion aus Texten verwendet, deren Inhalt zuvor manuell auf die verschiedenen Toponymcharakteristika untersucht wurde. Diese manuell erfasste Referenz wird mit denen vom Programm gefundenen Charakterisierungen verglichen und die Ergebnisse anhand verschiedener Metriken ausgewertet. Zum Abschluss werden in Kapitel 6 die Neuerungen und Ergebnisse nochmals zusammengefasst und ein Ausblick auf mögliche weitere Arbeiten auf Basis dieser Masterarbeit gegeben.

2 Grundlagen und Kontext

2.1 Sprachliche Stilmittel

2.1.1 Verwendungsarten

Definition

Unabhängig von der Einordnung in Stilmittel wie der Metapher oder der Allegorie kann es für eine eingehende Charakterisierung von Toponymverwendungen sinnvoll sein, einige einfache Verwendungsarten zu unterscheiden, die sich aus der bloßen Satzstruktur erschließen. Diese können dann zum einen die Grundlage für die Erkennung der eigentlichen Stilfiguren bilden, oder für sich stehend bereits wertvolle Charakteristika des Textes offenbaren, die etwa für eine Genreerkennung oder ähnliche Zwecke herangezogen werden können. Bereits existierende Arbeiten (vgl. Kapitel 2.3) haben hier keine entsprechende Unterteilung vorgenommen. Aus diesem Grund werden im Folgenden drei neue Verwendungsarten definiert, welche auf einfachen grammatikalischen Regeln beruhen und so eine transparente Unterteilung ermöglichen.

Referenz

Toponyme werden oft zusammen mit *Adpositionen* verwendet, um einen Bezug zum entsprechenden Ort herzustellen. Zu den Adpositionen gehören Prä-, Post- und Zirkumpositionen. Folgende Beispiele veranschaulichen dies:

Präposition

In der **Ukraine** wächst die Angst vor Instabilität.

Postposition

Der **Elbe** ENTLANG existieren viele Wanderwege.

Zirkumposition

UM **Ulm** HERUM sagen sich Fuchs und Hase Gute Nacht.

Eine solche Verwendung von Toponymen mit Adpositionen wirkt sich in vielen Fällen charakteristisch auf den entsprechenden Satz oder Teilsatz aus, zudem ist sie grammatikalisch klar von anderen Verwendungen unterscheidbar. Aus diesen Gründen werden diese unter der Verwendungsart *Referenz* zusammengefasst.

Attribut

In weiteren Fällen stehen Toponyme nicht für sich alleine im Satzkontext, sondern werden als Beschreibung für andere Substantive verwendet. So kann das Toponym zum einen als Adjektiv, oder als eigenes Nomen im Genitiv verwendet werden, wie die folgenden Beispiele zeigen:

Toponym als Adjektiv

> **Nordkoreanische** MEDIEN sprechen von einem Affront.
> Die **Würzburger** RESIDENZ stammt aus dem Barock.

Genitiv

> Die SCHULDEN **Norwegens** wachsen ins Unermessliche.
> **Islands** POLIZEI wehrt sich gegen weitere Vorwürfe.

Eine Besonderheit ergibt sich aus der eher in der deutschen Umgangssprache verbreiteten Nutzung des Dativ als Ersatz für den Genitiv, der u.a. in der Buchreihe „Der Dativ ist dem Genitiv sein Tod“ von Bastian Sick aufgegriffen wurde. Da diese Verwendungsart stellvertretend für die Verwendung als Genitiv steht, fällt sie auch in die hier untersuchte Kategorie:

Dativ

> Der KAISER von **China** dankt ab.

Die Verwendung von Toponymen als Beschreibung eines anderen Substantivs unterscheidet sich deutlich von einer alleinstehenden Nutzung. Verwendungen als Adjektiv, Genitiv oder als Dativ im Rahmen einer Possesivkonstruktion werden deshalb als eigene Verwendungsart *Attribut* definiert.

Standard

Alle weiteren Verwendungen von Toponymen werden nicht mehr gesondert unterteilt, sondern im Folgenden unter einer Sammelkategorie zusammengefasst. Hierzu zählen vor allem die Verwendungen von Toponymen ohne weitere Attribute oder Toponyme, die durch Adjektive näher beschrieben werden, aber nicht selbst als Adjektiv eingesetzt werden. Folgende Beispiele illustrieren diese Verwendungsart:

Ohne Attribute

> **Bamberg** ist Sitz des gleichnamigen Erzbistums der römisch-katholischen Kirche.

Adjektiv zum Toponym

> Das SCHÖNE **Spanien** ist immer eine Reise wert.

Diese Sammelkategorie wird somit als Verwendungsart *Standard* definiert. Im weiteren Verlauf der Arbeit wird nochmal genauer erläutert, dass sich auch in den einzelnen Verwendungsarten, vor allem jedoch in dieser Sammelkategorie, durchaus noch Unterscheidungsmerkmale ergeben

können. Allerdings wurden die Verwendungsarten vor allem deshalb auf wenige, allgemeine Kategorien beschränkt, um die weitere, darauf aufbauende Analyse mit Hinblick auf die rhetorischen Stilmittel so weit wie möglich zu vereinfachen.

2.1.2 Figuren

Einführung

Kolmer und Rob-Santer betonen, dass die Rhetorik als eigene Disziplin auf eine sehr lange Geschichte bis ins antike Griechenland zurückblicken kann, wobei sie dabei eng mit den Anfängen der Demokratie verbunden war. Im Rahmen der politischen Auseinandersetzungen musste jede Partei die Volksversammlung selbst von ihren Ideen und Vorsätzen überzeugen, damit diese auch umgesetzt werden konnten. Hierbei wurde laut den Autoren schon früh die Idee von der *Rede als Waffe* geprägt, nach der eine gute Rede auch einer schwächeren Idee zum Sieg verhelfen konnte [KRS02]. Über die Jahrhunderte wurde die *Redekunst* so immer weiter verfeinert, besonders sticht hier die Rhetorik im Römischen Reich hervor. Ciceros Hauptwerk „De oratore“ („Über den Redner“) sei zum Beispiel bis heute ein viel beachtetes Werk über „die Bedeutung der Redekunst, die Integrität des Redners“, seine Allgemeinbildung und seine Aufgaben, wozu neben dem eigentlichen Inhalt vor allem die Stilarten und die Darbietung eine zentrale Rolle spielen [KRS02].

Die heutigen Definitionen der verschiedenen Stilmittel gehen in vielen Bereichen auf die damaligen Werke zurück. Hierbei sind jedoch die einzelnen Kategorien nicht immer trennscharf voneinander abgrenzbar und widersprechen sich in einigen Fällen sogar. Die folgenden Erläuterungen stützen sich deshalb im Wesentlichen auf die Unterteilung aus dem „Studienbuch Rhetorik“ von Kolmer und Rob-Santer (vgl. [KRS02]). Weiterhin werden nur diese Stilmittel erläutert, die für die weitere Analyse und die folgende Charakterisierung von Toponymverwendungen eine Rolle spielen. So können bestimmte Stilelemente in der deutschen Sprache allgemein durchaus relevant sein. Wenn diese allerdings im Kontext von Toponymen selten oder gar nicht vorkommen, werden diese in dieser Arbeit nicht weiter behandelt.

Die Stilmittel werden hierbei in zwei verschiedene Kategorien, die *Figuren* und die *Tropen* eingeteilt, wobei auch hier die Unterscheidung in einigen Bereichen schwierig oder umstritten ist. Nach Kolmer und Rob-Santer handelt es sich bei Figuren um die Elemente eines Textes, bei dem ein gewöhnlicher „Ausdruck durch einen weniger gewöhnlichen“ ersetzt wird [KRS02]. Der eigentliche Sinn des Wortes oder der Wortgruppe bleibt hierbei allerdings erhalten. Die Figuren werden in die vier verschiedene Kategorien Klang-, Positions-, Sinn- und Satzfiguren unterteilt, die im Folgenden anhand von Beispielen kurz vorgestellt werden.

Klangfiguren

Alliteration

Aus dem Bereich der Klangfiguren wird in dieser Arbeit ausschließlich die Alliteration, auf deutsch auch *Stabreim* genannt, betrachtet. Wenn in einem Satz oder einem Teilsatz zwei

aufeinanderfolgende oder benachbarte Wörter die gleichen Anfangslaute, besonders zwei gleiche Konsonanten, aufweisen, liegt eine Alliteration vor.

Beispiel: **B**amberg **b**angt um seinen Ruf.

Positionsfiguren

Geminatio

Bei einer Geminatio handelt es sich um die Verdoppelung eines Worts oder einer Wortgruppe, analog einer Aufzählung. Durch sie wird in der Regel eine große Verstärkung des Ausdrucks angestrebt.

Beispiel: **Dänemark**, **Dänemark** über alles...

Anapher

Eine Anapher ist wie die Geminatio eine Figur der Wortwiederholung. Hier wird ein Wort allerdings mehrmals zu Beginn aufeinanderfolgender Sätze oder Satzteile verwendet. Auch hiermit wird in der Regel eine starke Hervorhebung des Textinhalts angestrebt.

Beispiel: **Frankreich** wird wachsen, **Frankreich** wird wieder aus der Krise kommen.

Epipher

Die Epipher entspricht einer Anapher, wobei hier das Stilmittel der Wiederholung am Ende eines Satzes oder eines Satzteils eingesetzt wird. Nach Kolmer und Rob-Santer kommt diese wesentlich seltener vor, allerdings wird durch die Wiederholung am Satzende das entsprechende Wort besonders stark betont, weil es durch die natürliche Pause nach dem Ende eines Satzes länger nachwirke [KRS02].

Beispiel: Entscheidend ist **Russland**. Ganz Europa schaut auf **Russland**.

Anadiplosis

Mit einer Anadiplosis kombiniert der Autor im Prinzip die beiden Stilmittel Anapher und Epipher. Hierbei entspricht das letzte Wort eines Satzes oder eines Satzteils dem ersten Wort des jeweils nächsten Satzes oder des nächsten Satzteils.

Beispiel: So etwas passiert nur in **Österreich**. **Österreich** war in dieser Hinsicht schon immer besonders.

Sinnfiguren

Epitheton Ornans

Unter der Vielzahl an existierenden Sinnfiguren scheint ausschließlich das Epitheton Ornans, das *schmückende Beiwort*, für die Untersuchung im Rahmen von Toponymverwendungen relevant zu sein. Hierbei wird das Substantiv, bzw. das Toponym im konkreten Kontext dieser Arbeit, um ein anderes Substantiv, ein Adjektiv oder eine Umschreibung

ergänzt, obwohl das zum Verständnis des Textes nicht notwendig wäre. Nach Kolmer und Rob-Santer verleiht dieses Stilmittel dem Wort allerdings eine größere Anschaulichkeit [KRS02].

Beispiel: Das FRÄNKISCHE **Bamberg** ist Teil des UNESCO-Weltkulturerbes.

Satzfiguren

Similitudo

Auch bei den Satzfiguren gibt es nur wenige Beispiele, die sich explizit auf ihre Untersuchung im Rahmen von Toponymverwendungen eignen. Eine zentrale Ausnahme ist die Similitudo, bzw. der einfache Vergleich. Hierbei wird ein Toponym mit einem anderen Begriff in Bezug gesetzt, indem Ähnlichkeiten oder Unterschiede aufgezeigt werden. Dieses Stilmittel dient vor allem zur besseren Veranschaulichung einer Aussage.

Beispiel: Die verwüstete Fläche ist DREIMAL SO GROSS WIE das **Saarland**.

2.1.3 Tropen

Einführung

Bei einem Tropus tritt eine neue Bedeutung an die Stelle des eigentlichen Wortsinns, das verwendete Wort wird also im übertragenen Sinne gebraucht. Kolmer und Rob-Santer sprechen beim Einsatz von Tropen von einem höheren Grad an „Lebhaftigkeit“ des Textes, der aus mehreren Ursachen resultieren kann. Der im übertragenen Sinne gebrauchte Begriff kann zum einen eine höhere Anschaulichkeit als der ersetzte Begriff besitzen. Zum anderen kann der intellektuelle Reiz darin bestehen, den eigentlichen Sinn des Textes erst entschlüsseln zu müssen [KRS02].

Bei den Tropen werden die zwei Kategorien *Grenzverschiebungstropen* und *Sprungtropen* unterschieden. Bei den Grenzverschiebungstropen handelt es sich um Ausdrücke, die dem ersetzten Begriff inhaltlich gleich oder ähnlich sind. Bei Sprungtropen sind hingegen beide Begriffe inhaltlich nicht miteinander vergleichbar. Hier sind die Übergänge zwischen den einzelnen Stilmitteln besonders fließend, weswegen die einschlägige Literatur teils deutlich unterschiedliche Einordnungen vornimmt. Die folgenden Definitionen fußen wiederum auf dem „Studienbuch Rhetorik“ von Kolmer und Rob-Santer [KRS02].

Grenzverschiebungstropen

Synekdoché

Unter einer Synekdoché versteht man in der Regel die Ersetzung eines Begriffs durch einen anderen, der zum ursprünglichen im weiteren oder engeren Zusammenhang steht. Die hier zusammengefassten Stilmittel sind auch unter *Pars pro toto* (Teil für Ganzes) oder *Totum pro parte* (Ganzes für Teil) bekannt. Bei Toponymen finden diese häufig in Ausdrücken Anwendung, wenn von den Tätigkeiten der Regierung eines Landes, einer Region oder einer

Stadt geredet wird. Das Toponym steht hier dann z.B. stellvertretend für die Regierung, welche in der jeweiligen Stadt ansässig ist.

Beispiel: **Berlin** hat beschlossen, das Elterngeld zu erhöhen.

Antonomasie

Bei der Verwendung einer Antonomasie wird ein Begriff durch eine charakteristische Eigenschaft bzw. ein bezeichnendes Merkmal ersetzt. Alternativ ersetzt ein Begriff umgekehrt ein solches Merkmal. Im konkreten Fall von Toponymen bezieht sich diese in der Regel auf ein Charakteristikum des jeweiligen Ortes, etwa zur bildhaften Beschreibung. Da Bamberg zum Beispiel wie Rom auf sieben Hügeln erbaut wurde und in Franken liegt, wird oft durch Verwendung einer solchen Antonomasie auch vom „fränkischen Rom“ gesprochen.

Beispiel: **Das fränkische Rom** hat eine lange Geschichte.

Metonymie

Bei einer Metonymie geht die Grenzverschiebung sogar noch über den eigentlichen Begriffsinhalt hinaus. Hier steht der verwendete Begriff nur noch logisch, räumlich oder etwa zeitlich in Beziehung. Im Zusammenhang mit Toponymen werden beispielsweise die handelnden Akteure durch den Ort, in dem sie wohnen, ersetzt.

Beispiel: **Frankreich** leistete großen Widerstand.

Sprungtropen

Metapher

Dieses Stilmittel wird oft eingesetzt, um einen abstrakten oder nur schwer verständlicher Begriff besser zu veranschaulichen. Es handelt sich aber nur dann um eine Metapher, wenn der verwendete und der ersetzte Begriff in keinem inhaltlichen Zusammenhang zueinander stehen. Mit Hinblick auf dessen Verwendung mit Toponymen kann er beispielsweise eingesetzt werden, wenn ein Gebiet nicht exakt umrissen ist, man aber dennoch eine gewisse Vorstellung hat, wo es sich befindet und wie groß es ist.

Beispiel: Besuchen Sie die **Wiege der Menschheit**!

Allegorie

Eine Allegorie hängt eng mit dem Stilmittel der Metapher zusammen. Hierbei wird der ursprüngliche Begriff aber auf ein bestimmtes Bild und dessen Charakter projiziert, welcher dann im Text genutzt wird. So stand beispielsweise der Deutsche Michel mit seiner charakteristischen Zipfelmütze vor allem im 19. Jahrhundert stellvertretend für die Biederkeit vieler Deutscher und somit für Deutschland allgemein. Eine weitere Nationalallegorie wäre etwa die Figur des Uncle Sam für die USA.

Beispiel: **Uncle Sam** marschiert in Afghanistan ein.

2.2 Textanalyse

2.2.1 Einführung

Zur Analyse von Texten in natürlicher Sprache existiert eine Vielzahl unterschiedlicher Werkzeuge. Viele davon sind sprachspezifisch, funktionieren also beispielsweise ausschließlich für Texte in englischer oder deutscher Sprache. Andere wiederum lassen sich über bestimmte Erweiterungen auch für verschiedene Sprachen einsetzen.

Um die Struktur und den Inhalt eines Textes verstehen zu können, werden zunächst die sogenannten *Tokens* mittels eines *Parsers* identifiziert [CMS10]. Ein Token entspricht dabei in vielen Fällen einem einzelnen Wort bzw. einem Satzzeichen. Diese Tokens bilden dann die Grundlage für die weiteren Analyseschritte, beispielsweise um einzelne Sätze und Satzteile anhand der Satzzeichen zu identifizieren und voneinander abgrenzen zu können. Dieser erste Schritt in der Textanalyse ist dabei nicht trivial, da eine bloße Unterteilung eines Textes in Tokens anhand der Leerzeichen in vielen Fällen zu kurz greift. Weiterhin existieren bestimmte Sonderzeichen, wie etwa der Apostroph oder der Bindestrich, welche integraler Bestandteil eines Wortes sein können und bei denen entschieden werden muss, ob die Zeichen, welche vor und nach diesen Sonderzeichen auftreten, als ein Wort oder als zwei Wörter zu betrachten sind. Darüber hinaus kennt beispielsweise die chinesische Sprache überhaupt keinen Worttrenner wie das Leerzeichen [CMS10]. In anderen Sprachen wie dem Deutschen wiederum stellt sich die Problematik der zusammengesetzten Wörter (Komposita), bei denen entschieden werden muss, ob diese als ein Wort oder als mehrere Wörter zu betrachten sind. Weitere Aufgaben ergeben sich aus der Behandlung von Wörtern je nach Groß-/Kleinschreibung, von Zahlen, aber auch von Punkten, die neben ihrer Funktion als Satztrenner z.B. auch Bestandteil von Eigennamen sein können.

Sind die einzelnen Tokens identifiziert, besteht die nächste Herausforderung darin, die entsprechende Wortart (englisch *part-of-speech*) und seine Grundform herauszufinden. Die Ermittlung der Grundform, dem *Lemma*, ist nötig, um Tokens miteinander in Beziehung setzen zu können, selbst wenn sie in verschiedenen grammatikalischen Formen im Text auftreten. So können über diese *Lemmatisierung* beispielsweise die Formen *ritt* und *geritten* auf den Infinitiv *reiten* zurückgeführt werden. Im weiteren Verlauf der Textanalyse wird dann ggf. nur noch das Lemma eines Tokens verwendet. Bei der Bestimmung der Wortart werden hingegen die einzelnen Token in verschiedene Kategorien wie *Substantiv*, *Adjektiv* oder *Verb* eingeteilt.

Darauf aufbauend können sich dann weitere, zum Teil deutlich komplexere Analysen anschließen. Bestimmte Werkzeuge ermöglichen es, die Eigenschaften bestimmter Wörter zu identifizieren. Hierzu zählt beispielsweise die Einordnung von Eigennamen in Kategorien wie *Person*, *Beruf*, *Ort* oder *Organisation*. Diese *Named Entity Recognition* spielt im konkreten Bezug zu dieser Arbeit besonders bei der Toponymerkennung eine zentrale Rolle. Die Bedeutung eines Wortes lässt sich allerdings auch noch in einer weiteren Dimension betrachten. So bieten bestimmte Datenbanken die Möglichkeit, Wörter mit ähnlicher oder gleicher Semantik, sogenannter *Synonyme*, zu identifizieren. Weiterhin beinhalten diese Datenbanken oft auch Daten über eine mögliche Mehrfachbedeutung eines Wortes, einem sogenannten *Homonym*, oder entsprechender über- und untergeordneter Bedeutungen, also beispielsweise die Charakterisierung eines *Apfels* als *Obst*.

Diese Informationen können dann u.a. dazu verwendet werden, um Aussagen in verschiedenen Sätzen im Verlauf eines Textes miteinander in Beziehung zu setzen.

Zur detaillierten Analyse eines Satzes ist es in vielen Fällen auch notwendig, die einzelnen Abhängigkeiten der Satzteile und ihre Funktion genau zu kennen. Speziell in einer Sprache wie der deutschen ist es ausschließlich anhand der Reihenfolge der einzelnen Satzteile nur schwer zu erkennen, welche Wörter Subjekt, Prädikat und Objekt sowie welche Teile des Satzes zum Haupt- oder ggf. weiteren Nebensätzen gehören. Weiterhin ist oft auch eine Charakterisierung der Fälle eines Wortes, also die Unterscheidung zwischen Genitiv, Akkusativ oder Dativ, nötig, um komplexe Untersuchungen durchführen zu können. Hierzu wird eine sogenannte *morphologische Analyse* des Textes durchgeführt.

Im Folgenden werden einige konkrete Werkzeuge zu den genannten Teilgebieten kurz vorgestellt. Die Auswahl erfolgte anhand ihrer Eignung zur Analyse von Texten in deutscher Sprache sowie ihrem Einsatz in verwandten Arbeiten, die in Kapitel 2.3 vorgestellt werden. Eine detaillierte Beschreibung der einzelnen Funktionen, sowie ihrer Vor- und Nachteile, erfolgt dann in den Kapiteln 3 und 4, wenn sie konkret zur Erkennung und Charakterisierung von Toponymen eingesetzt werden.

2.2.2 Analysewerkzeuge

Wortartanalyse mit dem TreeTagger

Zur Wortartanalyse oder dem Part-of-Speech-Tagging von deutschen Texten wird häufig der eigentlich sprachunabhängige *TreeTagger* eingesetzt, der bereits im Jahr 1994 von Helmut Schmid vorgestellt wurde [Sch94] und im Jahr darauf weiter verbessert wurde [Sch95]. Auf der Grundlage von Entscheidungsbäumen annotiert er Wörter eines Textes mit den zugehörigen Wortarten und den entsprechenden Lemmata. Das eigentliche Part-of-Speech-Tagging für Deutsch erfolgt auf Basis des *Stuttgart-Tübingen-Tagsets* (STTS), vgl. hierzu [STT99]. Zur Verdeutlichung der Ergebnisse wird als Beispiel der Satz „Die Olympischen Winterspiele 2014 in Sotschi haben begonnen“ mit dem TreeTagger analysiert. In Abbildung 2.1 wurden zur besseren Visualisierung noch Überschriften hinzugefügt. Die eigentliche Ausgabe erfolgt ohne diese.

Wie das Ergebnis zeigt, analysierte TreeTagger den Satz Wort für Wort und konnte bis auf den Ort *Sotschi* zu jedem Token auch ein passendes Lemma identifizieren. Zur genaueren Analyse der Wortarten müssen die entsprechenden Abkürzungen der ermittelten Part-of-Speech-Tags (POS-Tags) in den Erläuterungen zum STTS nachgelesen werden. So wurden zum Beispiel die Worte *Winterspiele* als auch *Sotschi* korrekt als normale Nomina identifiziert, während *begonnen* als Vollverb im Partizip Perfekt erkannt werden konnte.

Abhängigkeitsanalyse mit ParZu

Der *Zurich Dependency Parser for German*, kurz ParZu, baut bei der Abhängigkeitsanalyse auf den Ergebnissen des TreeTaggers und einem von drei unterstützten Werkzeugen zur morphologischen Analyse eines Satzes auf. Bei der Analyse der linguistischen Struktur eines Satzes können

```
Token         POS-Tag  Lemma
==================================
Die           ART      die
Olympischen   ADJA     olympisch
Winterspiele  NN       Winterspiel
2014          CARD     @card@
in            APPR     in
Sotschi       NN       <unknown>
haben         VAFIN    haben
begonnen      VVPP     beginnen
.             $.       .
```

Abbildung 2.1 — TreeTagger-Ausgabe für den Beispielsatz

unter anderem das Subjekt, Prädikat und Objekt eines Satzes identifiziert werden[1]. Darüber hinaus werden die durch TreeTagger und die morphologische Analyse ermittelten Informationen, wie etwa die Kasus der einzelnen Satzteile, ebenfalls ausgegeben. Die Autoren beschreiben in der zugrundeliegenden Arbeit (vgl. [SSVW09]) die Besonderheiten der deutschen Sprache, besonders mit Hinblick auf die vergleichsweise frei wählbare Wortreihenfolge. Sie sprechen hier ebenfalls die Tatsache an, dass Analysewerkzeuge für Deutsch in der Regel schlechtere Ergebnisse liefern als ähnliche Programme für Englisch, wobei es umstritten zu sein scheint, ob es grundsätzlich an der Sprache liegt, oder an anderen Ursachen wie dem geringeren Fokus seitens der Forschung und Entwicklung.

ParZu bietet als Ergebnis der Abhängigkeitsanalyse sowohl eine textuelle Ausgabe zur maschinellen Weiterverarbeitung als auch eine grafische Ausgabe an. Jedes Token wird hierbei über seine Position im Satz mit einer entsprechenden ID (`Pos`) identifiziert. Ist ein Token von einem anderen abhängig, so referenziert das abhängige Token das andere mit dessen ID im Feld `HeadPos`. Für den Beispielsatz gibt ParZu das in Abbildung 2.2 gezeigte Ergebnis aus. Auch hier wurden zur besseren Darstellung wiederum Überschriften verwendet, die in der eigentlichen Ausgabe nicht vorhanden sind. Darüber hinaus werden nur die für das Verständnis wesentlichen Ergebnisfelder dargestellt.

Die Informationen, die über die TreeTagger-Ausgabe hinausgehen, finden sich in den Feldern `Morph`, `HeadPos` und `Class`. Das Feld `Morph` beinhaltet die morphologischen Informationen zum entsprechenden Token, wie zum Beispiel, ob es sich um ein Wort in der Einzahl oder Mehrzahl handelt, welcher Fall benutzt wird oder in welcher Person das Verb verwendet wird. Bei der Interpretation der ParZu-Ausgabe des Feldes `HeadPos` wird deutlich, dass sowohl der Punkt als Satzende und das Prädikat auf der Wurzelebene der Satzstruktur stehen, also mit `0` kein weiteres Elterntoken referenzieren. Das Subjekt *Winterspiele* und das Partizip *begonnen* referenzieren wiederum das Prädikat mit der ID `7` als `HeadPos`. Alle weiteren abhängigen Teile des Satzes folgen diesem Schema. Das Feld `Class` kategorisiert das Token anhand seiner Abhängigkeiten zu anderen Satzteilen. Die verwendeten Klassen folgen dem Schema, welches Kilian Foth in [Fot04] beschrieben hat.

[1] `https://github.com/rsennrich/ParZu` - abgerufen am 22. Februar 2014

Pos	Token	Lemma	POS-Tag	Morph	HeadPos	Class
1	Die	die	ART	Def|_|Nom|Pl	3	det
2	Olympischen	olympisch	ADJA	Pos|_|Nom|Pl|Wk|	3	attr
3	Winterspiele	Winterspiel	NN	_|Nom|Pl	7	subj
4	2014	2014	CARD	_	3	app
5	in	in	APPR	_	3	pp
6	Sotschi	Sotschi	NE	Neut|_|Sg	5	pn
7	haben	haben	VAFIN	3|Pl|Pres|_	0	root
8	begonnen	beginnen	VVPP	_	7	aux
9	.	.	$.	_	0	root

Abbildung 2.2 — ParZu-Ausgabe für den Beispielsatz

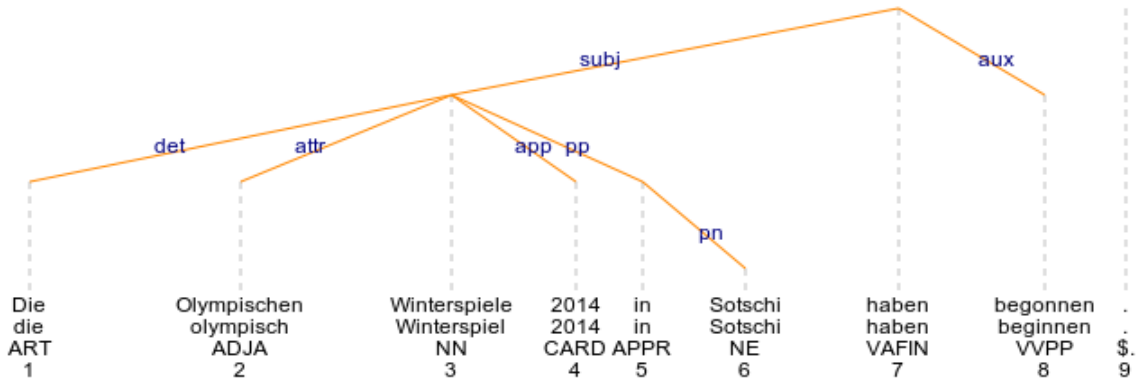

Abbildung 2.3 — Grafische Ausgabe von ParZu für den Beispielsatz

Abbildung 2.3 zeigt die Abhängigkeiten zwischen den Satzteilen besonders deutlich auf. Ausgehend vom Prädikat des Satzes wird eine Baumstruktur aufgespannt, in der die Satzteile die Knoten und die Kanten die Beziehungen zwischen ihnen darstellen. Die Wurzel des Baums besteht aus dem Prädikat. Die Blätter des Baumes sind die Satzteile, von denen keine weiteren Abhängigkeiten ausgehen.

Lexikalisch-semantische Analyse mit GermaNet

GermaNet ist eine Datenbank mit lexikalisch-semantischen Informationen über eine Vielzahl an Nomen, Verben und Adjektiven der deutschen Sprache, welche an der Universität Tübingen entwickelt wird. Hierbei werden die einzelnen Wörter anhand ihrer Bedeutung in sogenannten *Synsets* gruppiert und semantische Abhängigkeiten zwischen diesen Synsets modelliert. Es lehnt sich hierbei an dem für die englische Sprache verfügbaren *WordNet* an, welches an der Universität in Princeton entwickelt wird[2]. Laut Aussage der ursprünglichen Autoren von GermaNet ist es hierbei kompatibel zu WordNet, beinhaltet allerdings einige Anpassungen, sodass WordNet-

[2] `http://wordnet.princeton.edu/` - abgerufen am 09. März 2014

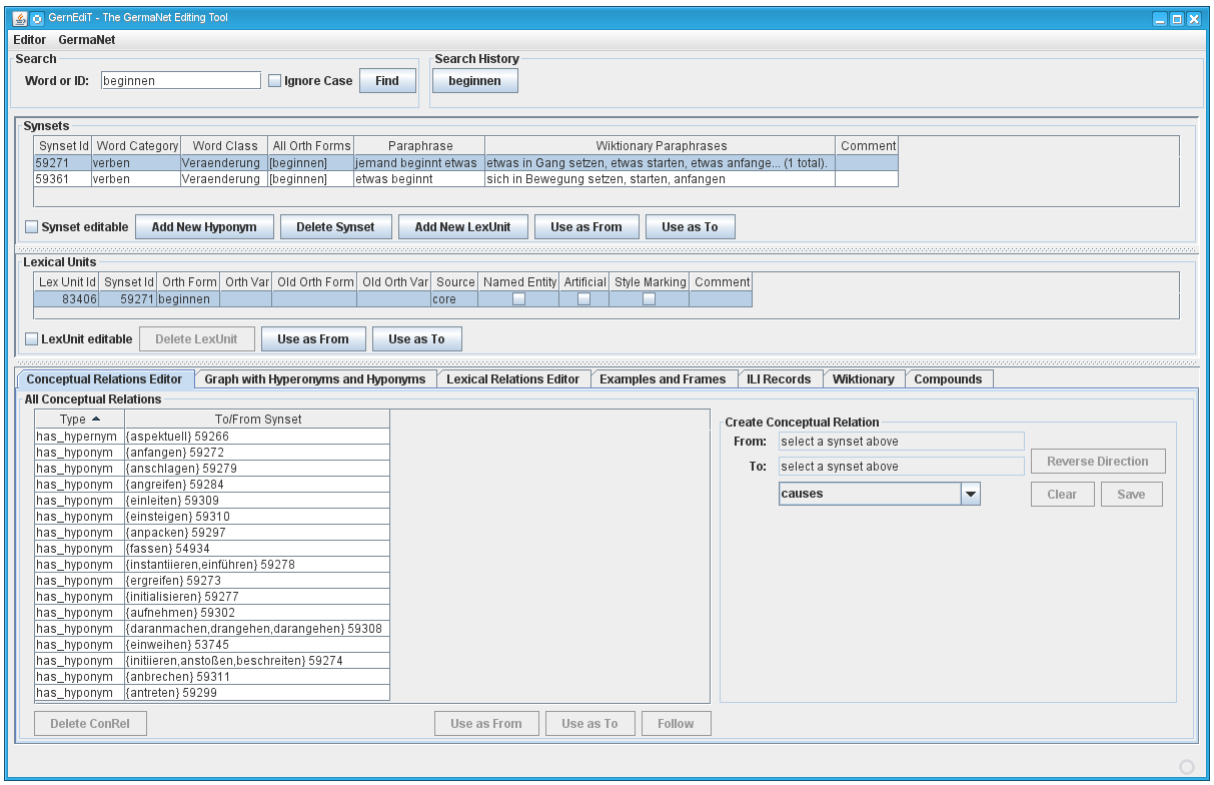

Abbildung 2.4 — Ausgabe von GernEdiT für *beginnen*

basierte Anwendungen mit nur geringen Anpassungen auch mit GermaNet lauffähig gemacht werden können [HF97].

GermaNet unterteilt Wörter hierbei in die drei verschiedenen Kategorien *Nomen*, *Adjektive* und *Verben*. Davon unabhängig werden sie auch in entsprechende Klassen eingeteilt. Beispiele für solche Klassen sind *Bewegung*, *Kommunikation* oder *Verhalten*. Neben diesen Klassen sind insbesondere die semantischen Beziehungen zwischen den Worten für die weitere Analyse von Bedeutung. Hierzu zählen die Beziehungstypen *Synonym* und *Antonym*, welche Worte gleicher oder entgegengesetzter Bedeutung bezeichnen. Neben weiteren Typen, die sowohl in [HF97] als auch mit allen bisherigen Erweiterungen auf [Ebe14] erläutert werden, sind hier vor allem *Hyponymie* und *Meronymie* von Bedeutung. Hyponymien bezeichnen *ist-ein*-Beziehungen, so etwa wie bei *Eiche* zu *Baum*. Meronymien sind dabei Teile-Ganzes-Beziehungen und sind ausschließlich für Nomen modelliert, ein Beispiel hierfür wäre das Verhältnis von *Lenkrad* zu *Auto*.

Abbildung 2.4 zeigt die Informationen aus GermaNet für das Verb *beginnen*. Zur Darstellung wird die Anwendung GernEdiT verwendet, welche zusammen mit den reinen GermaNet-Daten hauptsächlich für Änderungszwecke an der Datenbank zur Verfügung gestellt wird.

```
Die           /O
Olympischen   /I-ORG
Winterspiele  /I-ORG
2014          /O
in            /O
Sotschi       /I-LOC
haben         /O
begonnen      /O
.             /O
```

Abbildung 2.5 — Ausgabe des Stanford NER mit dem deWaC-Klassifizierer

Named Entity Recognition mit dem Stanford NER

Das Named Entity Recognition-System (NER) der Stanford University[3] bietet die Möglichkeit, in verschiedenen Sprachen die Eigenschaften bestimmter Wörter in einem Text, beispielsweise Person, Organisation oder Ort, zu ermitteln. Die eigentliche Erkennung erfolgt mit speziellen Klassifizierern, die in der Regel mit Hilfe von entsprechenden Trainingsdaten erstellt wurden. In der ausgelieferten Version unterstützt der *Stanford NER* ausschließlich die englische Sprache und beinhaltet Klassifierungsmodelle für verschiedene Charakteristika, teils ausschließlich für *Ort*, *Organisation* und *Person*, teils auch mit weiteren Kategorien wie *Zeit* oder *Geld*.

Manaal Faruqui und Sebastian Padó haben in [FP10] hingegen zwei Klassifizierer vorgestellt, die für deutsche Texte entsprechende Worte in die Kategorien Person, Organisation und Ort einteilen können. Diese wurden mittels der deWaC- und HGC-Korpora erstellt. Die Autoren empfehlen in der Dokumentation zu einer älteren Version[4] die Nutzung des HGC-Klassifierers für Nachrichtentexte und den deWaC-Klassifizierer für alle anderen Textarten. Hauptgrund hierfür ist, dass der HGC-Korpus ausschließlich aus Nachrichtentexten besteht und der deWaC-Korpus deutsche Internetinhalte ohne bestimmten Fokus auf ein Genre enthält.

Wie Abbildung 2.5 zeigt, weist der Stanford NER mit dem deWaC-Klassifizierer im Beispielsatz die *Olympischen Winterspiele* der Kategorie `I-ORG`, also einer Organisation, zu. *Sotschi* wird korrekt einem Ort, also der Kategorie `I-LOC`, zugeordnet.

Das Analyseframework GATE

Die *General Architecture for Text Engineering*, kurz GATE, ist ein Framework sowie eine grafische Entwicklungsumgebung, welche auf die Erfordernisse der Computerlinguistik zugeschnitten ist [CM02]. Zur Analyse von Texten kann auf eine Vielzahl an eingebauten Werkzeugen sowie weiterer Plugins zurückgegriffen werden. Die Bandbreite reicht hier von einfachen Tokenisierern über Instrumenten zur Named Entity Recognition bis hin zu komplexer morphologischer Analysesoftware. Die Verarbeitung erfolgt hierbei mit der Software GATE Developer. GATE Plugins zur Textanalyse werden in der Regel als sogenannte *Processing Resources* eingebunden und in

[3] `http://nlp.stanford.edu/software/CRF-NER.shtml` - abgerufen am 09. März 2014
[4] `http://www.nlpado.de/~sebastian/software/ner/README.1.1.1` - abgerufen am 09. März 2014

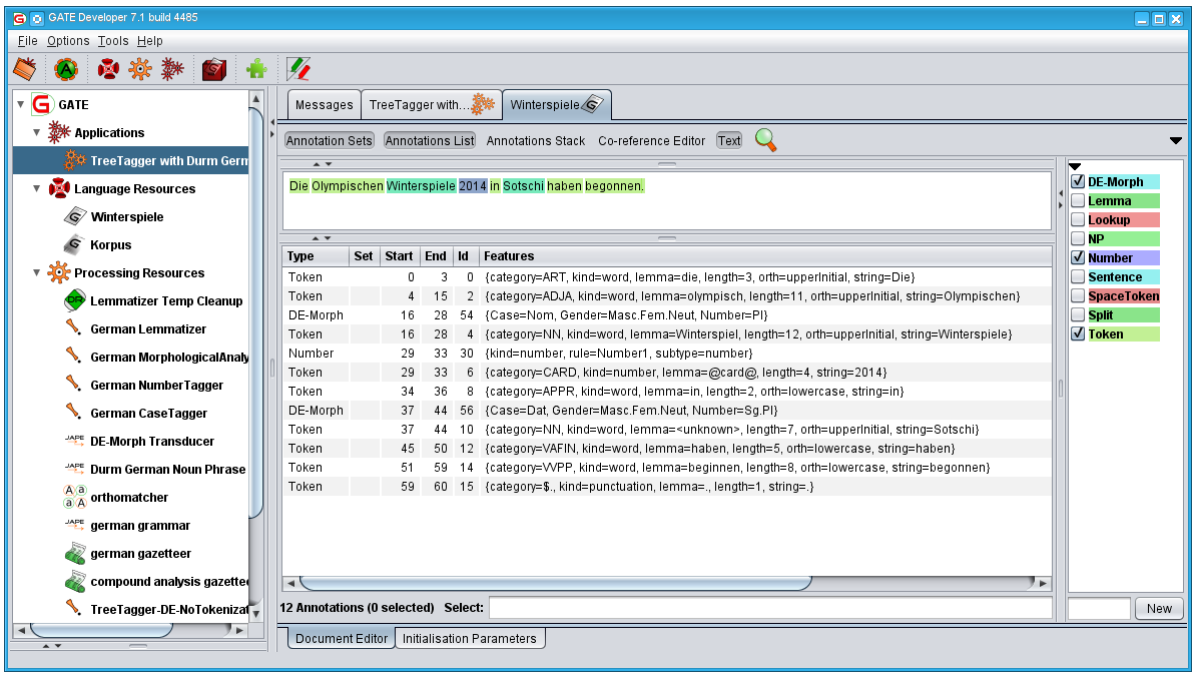

Abbildung 2.6 — Ausgabe von GATE Developer für den Beispielsatz

einer Pipeline zu einer Anwendung hintereinander geschaltet. Die verschiedenen Komponenten einer solchen Anwendung können so auch voneinander abhängen. Am Anfang einer solchen Pipeline steht in der Regel ein Tokenisierer, darauf kann beispielsweise ein POS-Tagger die einzelnen Wortarten ermitteln, worauf sich eine weitere Komponente bei der Ermittlung der verwendeten Kasus im Text verlässt.

Die zu analysierenden Texte, aber auch etwa die zugrundeliegenden Wörterbücher, werden als *Language Resources* eingebunden. GATE ist hierbei dafür ausgelegt, die Analyse nicht auf einzelnen Texten, sondern auf ganzen Textkorpora durchzuführen. Für die Prozessierung müssen also die entsprechenden Texte in einen Korpus aufgenommen werden. Die Ausgabe der Analyseergebnisse erfolgt dann in den *Visual Resources*. Im speziellen Fall von GATE Developer ist dies beispielsweise der Texteditor der grafischen Benutzerschnittstelle.

Die Processing Ressources halten ihre Ergebnisse mit Hilfe von Annotationen am entsprechenden Text, bzw. an den einzelnen Wörtern, fest. Annotationen haben einen *Typ* und beinhalten *Features*, welche die Ergebnisse zum Wort oder Textabschnitt näher beschreiben. Mengen von Annotationen werden in GATE wiederum *Annotation Sets* genannt. Wird beispielsweise der TreeTagger zur Wortartanalyse in GATE eingebunden, so setzt es eine vorhergehende Tokenisierung des Textes voraus. Jedes einzelne Token muss also mit einer Annotation des Typs `Token` versehen sein. Die TreeTagger-Komponente versieht dann jede Token-Annotation mit weiteren Features für den POS-Tag und das gefundene Lemma. Die Features werden dabei mit einem Schlüssel-Wert-Paar, einer eindeutigen ID, sowie mit Positionsinformationen am Text hinterlegt.

Abbildung 2.6 zeigt die Ausgabe von GATE Developer für den Beispielsatz. Hierbei wurde eine Anwendung mit verschiedenen Komponenten genutzt, welche aus einem Tokenisierer, dem TreeTagger, einer Wortartanalyse-Software sowie einem Plugin zur morphologischen Analyse von deutschen Texten besteht. Zur Named Entity-Recognition wurden hierbei verschiedene Elemente der GATE-eigenen Informationsextraktionssoftware ANNIE (*A Nearly-New Information Extraction System*[5]) verwendet, welche unter anderem auch Orte mit einer Kombination aus grammatikalischen Regeln und Wörterbüchern, sogenannten *Gazetteers*, erkennen kann. Im konkreten Fall schlägt dies aber fehl, da *Sotschi* nicht mit einer Annotation der Kategorie `Location` versehen wurde. Die erfolgreiche Einbindung des TreeTaggers lässt sich an den passenden Features der `Token`-Annotationen ablesen. Hinzukommen noch morphologische Informationen über Teile des Satzes, welche in den Ausprägungen des Typs `DE-Morph` hinterlegt wurden.

Die Einbindung weiterer Komponenten als Plugins erfolgt in der Regel als CREOLE-Ressourcen (*Collection of REusable Objects for Language Engineering*[6]). Diese Ressourcen bestehen meistens aus einfachen Java-Archiven (JARs), die über eine XML-Konfgurationsdatei dem Framework bekannt gemacht werden und so bei Bedarf geladen werden können. GATE-Komponenten können allerdings auch dazu genutzt werden, eigene Textanalysesoftware zu entwickeln. Hierzu muss die GATE Embedded-Bibliothek eingebunden werden, welche über eine dokumentierte Benutzerschnittstelle angesprochen werden kann, um beispielsweise Annotationen auszulesen, oder einen Korpus mit eigenen Annotationen anzureichern. Weiterhin bietet GATE auch die Möglichkeit, Annotationen mittels der Auszeichnungssprache JAPE (*Java Annotation Patterns Engine*[7]) manipulieren zu können. Auf Basis von regulären Ausdrücken können hier Regeln definiert werden, mit deren Hilfe Annotationen im Text erstellt oder verändert werden können. Diese Technik wird in vielen von GATE ausgelieferten Plugins genutzt, um beispielsweise auf Basis von JAPE-Regeln und zusätzlich angebundenen Gazetteers eine Named Entity-Recognition auf Texten durchführen zu können.

2.3 Verwandte Arbeiten

2.3.1 Allgemeines zum Geographischen Information Retrieval

Wie bereits zu Beginn der Arbeit dargelegt, ist das Ermitteln des tatsächlichen Informationsbedürfnisses des Benutzers einer Suchmaschine besonders bei Anfragen mit einem geographischen Bezug von zentraler Bedeutung. Henrich und Lüdecke haben sich in [HL07] mit der Charakterisierung dieser Bedürfnisse befasst. Hierbei stellen sie klar, dass geographische Informationsbedürfnisse immer aus einer konzeptuellen und einer geographischen Komponente bestehen. Der konzeptuelle Teil besteht dabei aus einem Objekt - dem, wonach der Nutzer sucht - und einem Prädikat - dem, was der Nutzer mit dem Objekt vorhat. Beispiele für ein Objekt können Güter oder Informationen sein, während ein Prädikat etwa *kaufen* oder *hinfahren* sein kann. Der geographische Aspekt besteht dabei aus vier Eigenschaften: der Intention, der Abdeckung, dem Umriss und der Distanz.

[5] `http://gate.ac.uk/sale/tao/splitch6.html#x9-1310006` - abgerufen am 15. März 2014
[6] `http://gate.ac.uk/sale/tao/splitch4.html#x7-700004` - abgerufen am 15. März 2014
[7] `http://gate.ac.uk/sale/tao/splitch8.html#x12-2180008` - abgerufen am 15. März 2014

Yi, Raghavan und Leggetter befassen sich in [YRL09] ebefalls mit der Ermittlung geographischer Informationsbedürfnisse von Nutzern. Ausgehend von einem Sprachenmodell für Städte versuchen sie die geographischen Informationsbedürfnisse von Nutzern zu identifizieren und im Anschluss herauszufinden, ob sich die Anfrage auf etwas im Umkreis der aktuellen Position des Nutzers bezieht. Die Arbeit liefert zudem einige Details zum Anteil an geographischen Suchanfragen an der Gesamtheit sowie Informationen über implizite geographische Referenzen in Suchen.

Eine eher allgemeine Abhandlung über geographische Bezüge in Texten liefern Radding und Western in [RW10]. Sie liefern einige Beispiele hierfür, wie spezifische Toponyme im Laufe der Zeit aus allgemeinen Ortsbezügen oder sogar normalen Nomen entstehen können. Als Muster für solche Entwicklungen dienen *Le Havre* (der Hafen) in Frankreich oder *Buffalo* (Büffel) im US-Bundesstaat Texas. Einige wichtige Erkenntnisse der Arbeit für die Toponymerkennung und -auflösung sind dabei, dass Toponyme in vielen Bereichen sprachspezifisch sein können und sich die Bedeutung der Orte über die Zeit signifikant ändern können.

2.3.2 Toponymerkennung und -auflösung

Andogah, Bouma und Nerbonne liefern in [ABN12] eine ausführliche Übersicht über verschiedene Arbeiten speziell zum Themenbereich Toponymauflösung. Sie kategorisieren hierbei zum einen die verschiedenen Möglichkeiten, einem Dokument einen geographischen Rahmen zu geben. Zum anderen listen sie verschiedene Kriterien auf, wie Toponyme den vom Autor gemeinten geographischen Koordinaten zugewiesen werden können. Stokes und andere betonen hierzu in [SLMR08], wie zentral sowohl eine Named Entity-Recognition als auch ihre Klassifizierung im Prozess der Toponymerkennung und -auflösung sind. Die Toponymauflösung könne hierbei mit der *Word Sense Disambiguation* (WSD) anderer Wörter verglichen werden, bei dem auch der Kontext des Worts für die Bestimmung der Semantik des Worts herangezogen wird. Jochen Leidner griff diesen Aspekt bereits zuvor in seiner Arbeit über die Evaluation der Toponymauflösung auf und liefert zudem noch einige weitere Details zu der genauen Abgrenzung mit der WSD [Lei06].

In seiner späteren Dissertation gibt Leidner dazu noch eine viel detailliertere Übersicht über die Aufgaben und Herausforderungen bei der Toponymauflösung [Lei07]. Hierbei nimmt er Bezug auf die unterschiedlichen Motivationen, die hinter weiterer Forschung und Entwicklung in diesem Bereich stehen können. Angefangen mit einer geographischen Suche über Navigation im Speziellen bzw. ortsbasierten Diensten im Allgemeinen bis hin zu einer Verbesserung der Beantwortung von Fragen mit geographischen Bezügen ist hier vieles denkbar. Für weitere Arbeiten in diesem Bereich sind hier vor allem seine Argumente für die Art und Weise der Erstellung der Erkennungssoftware und die möglichen Fehler der Toponymauflösung an den verschiedenen Stellen seiner Toolchain von Bedeutung. In einer späteren Arbeit fasst er mit Michael Lieberman nochmals die aktuellen Möglichkeiten der Toponymauflösung zusammen und gibt vor allem noch eine Übersicht über die momentan offenen Forschungsthemen in diesem Bereich, etwa die Erkennung unbekannter Toponyme [LL11].

Eine weitere Dissertation in diesem Themenbereich wurde bereits zur Begriffserklärung in Kapitel 1.2.1 herangezogen [LÏ0]. Volker Lüdecke gibt hier unter anderem eine detaillierte Auflistung der aktuell eingesetzten Techniken zur Toponymerkennung und unterscheidet diese in listen-

basierte, regelbasierte, statistische und hybride Verfahren. Hierbei nimmt er auch zu bereits existierenden Möglichkeiten zur Toponymerkennung von deutschen Texten Stellung. Kurz geht er zudem auf Toponyme ein, die in einem übertragenen Sinn eingesetzt werden. Da seine Arbeit einen anderen Fokus hat, geht Lüdecke hierbei aber nicht weiter ins Detail.

Ähnlich wie Lüdecke spezialisiert sich auch Tobias Brunner in seiner Masterarbeit auf die Toponymerkennung und -auflösung in deutschsprachigen Texten [Bru08]. Das Ziel seiner Arbeit war es, die geographischen Bezüge von Zeitungsartikeln herauszufinden. Hierzu nutzt er unter anderem das GATE-Framework zur Sprachanalyse sowie GeoNames[8] als Gazetteer. Allerdings werden auch hier Toponyme, welche im übertragenen Sinn in Texten verwendet werden, nicht eigenständig sondern wie andere Toponyme behandelt. Weiterhin klammert die Arbeit die Toponyme aus, die als Adjektive gebraucht werden.

Eine weitere zentrale Fragestellung in der Forschung stellt das Thema Ambiguitäten sowohl in einer Benutzeranfrage an eine Suchmaschine, als auch in den zu durchsuchenden Dokumenten dar. Die Herausforderungen in diesem Bereich wurden bereits in Kapitel 1.2.2 angesprochen. Bordogna und andere betonen in ihrer Arbeit über die Modellierung des Benutzerkontexts im geographischen Information Retrieval ausdrücklich, dass die Bedeutung von vagen Anfragen wie *in der Nähe* eines Ortes stark vom Kontext und der Wahrnehmung des entsprechenden Nutzers abhängen. Weiterhin müssten die verschiedenen Namen eines Ortes in verschiedenen Sprachen und in verschiedenen Jahrhunderten berücksichtigt werden [BGP12]. Wie diese Ambiguitäten eliminiert bzw. aufgelöst werden können, zeigen neben ihnen unter anderem auch Overell und Rüger [OR08], die Kookkurrenzmodelle zur besseren Unterscheidung nutzen. Ein weiteres Modell und zudem einige konkrete Beispiele für die Lokalisierung von vagen Benutzeranfragen wie dem *Weißwurstäquator* liefern Henrich und Lüdecke in [HL08]. Schlieder und Henrich fassen existierende Modelle zur Auflösung ungenauer Ortsangaben zusammen. Sie heben besonders hervor, dass eine Region sowohl physisch als auch sozial abgegrenzt werden kann [SH11]. Buscaldi unterteilt zudem die gängigen Methoden zur Vermeidung von Ambiguitäten in die drei Kategorien *kartenbasiert*, *wissensbasiert* und *datengetrieben*, wobei die letzte Variante Methoden zusammenfasst, die maschinelles Lernen verwenden [Bus11].

2.3.3 Toponymcharakterisierung

Unabhängig von der konkreten Charakterisierung von Ortsbezügen haben sich bereits viele Autoren mit der Bestimmung rhetorischer Figuren und der darauf aufbauenden Strukturierung von Sätzen und ganzen Texten befasst. Da sich die dort verwendeten Methoden und Ergebnisse auch in einigen Fällen auf die Toponymcharakterisierung übertragen lassen, werden im Folgenden sowohl allgemeine Untersuchungen als auch spezielle Arbeiten vorgestellt. Eine Einordnung und Klassifizierung der einzelnen rhetorischen Figuren und Stilmittel wurde bereits in Kapitel 2.1 durchgeführt. Kelly und andere gehen in [KAH+10] noch einen Schritt darüber hinaus und stellen den Stand ihrer Arbeit bei der Definition einer ganzen Ontologie von rhetorischen Figuren vor, bei der sie diese gleich unter mehreren Gesichtspunkten und Dimensionen betrachten und einordnen. Im weiteren Verlauf dieser Masterarbeit wird sich die Analyse jedoch auf die eingangs erläuterten Typen beschränken.

[8] `http://www.geonames.org/`

Abseits der Theorie hat sich Daniel Marcu bereits 1997 mit der Untersuchung von Sätzen und deren Aufteilung in rhetorische Strukturen befasst [Mar97]. Er entwickelt hier zwei Algorithmen, die es erlauben, einen Satz in einen sogenannten Diskursbaum von abhängigen rhetorischen Strukturen zu untergliedern. Jakub Gawryjolek entwickelte für seine Masterarbeit ein eigenes Werkzeug zur Annotation und Visualisierung von rhetorischen Stilfiguren in englischen Texten und nutzte hierbei eine Vielzahl verschiedener Werkzeuge wie das lexikalisch-semantische Netz WordNet oder eine grammatikalische Abhängigkeitsanalyse der Satzteile [Gaw09]. Christin Schätzle untersuchte in ihrer Masterarbeit literarische Texte in deutscher Sprache auf das Vorkommen von klassischen Genitiven, nutzte hier aber ausschließlich eine Wortarterkennung mit dem TreeTagger, was in einigen Fällen zu Erkennungsproblemen führte [Sch13].

Markert und Hahn haben sich bei der Analyse und Interpretation von rhetorischen Stilfiguren in der englischen Sprache auf die Metonymie beschränkt [MH02]. Allerdings ist diese Einordnung nicht mit der in Kapitel 2.1 vorgenommenen Unterteilung zu vergleichen, da die Metonymie als eine Art Überbegriff verwendet wird. Sie stellen sich in ihrer Arbeit zudem ausdrücklich in einen Gegensatz zu vorigen Arbeiten, indem sie die Verwendung von Worten im übertragenen und eigentlichen Sinne auf eine Stufe stellen und beide unabhängig voneinander analysieren und interpretieren. Hierzu nutzen sie verschiedene sprachliche Modelle, etwa zur Einordnung von Teile-Ganzes-Beziehungen von Worten. In einer weiteren Arbeit stellt Markert zusammen mit Nissim einen Algorithmus zur Erkennung von Metonymien vor, der auf überwachtem maschinellen Lernen basiert [NM03]. In der Einleitung konkretisieren sie die in der vorhergehenden Arbeit eingeführte Nutzung der Kategorie Metonymie anhand verschiedener Beispiele u.a. mit einigen Toponymen weiter. Auch bei der Lösung der SemEval-2007-Aufgabe griffen beide diese Kategorisierung wieder auf und evaluierten verschiedene Ansätze zur Erkennung rhetorischer Figuren [MN09]. Hierbei nutzten sie unter anderem ANNIE aus dem GATE-Framework zur Erkennung von Toponymen und Organisationsnamen.

Einige weitere Arbeiten befassen sich im besonderen Maße mit der Charakterisierungen von Toponymverwendungen, allerdings wie auch bei den zuvor genannten Arbeiten ausschließlich für die englische Sprache. Buscaldi und Rosso konzentrieren sich hierbei auf Teile-Ganzes-Beziehungen von Orten und nutzen WordNet [BR08]. Zentrales Ziel war eine bessere Toponymauflösung im Falle von Geo-Geo-Ambiguitäten, wobei nach eigener Aussage WordNet als Basis für die Analyse nur begrenzt geeignet war. Leveling und Hartrumpf stellen in [LH08] Details eines Klassifizierers vor, welcher Toponyme erkennen soll, die metonymisch verwendet werden. Hierbei nutzen sie die Kategorien, die Markert und Hahn in [MH02] vorgestellt hatten. Sie stellen heraus, dass es für das geographische Information Retrieval von zentraler Bedeutung ist, Toponymverwendungen im eigentlichen und übertragenen Sinn voneinander zu trennen. Der Grund hierfür ist, dass im engeren Sinn nicht in allen Fällen wirklich Orte referenziert werden, wenn Toponyme als Metonymien vorliegen. Darauf aufbauend beziehen Nastase und andere WikiNet, das auf Daten der Wikipedia basiert, zur Metonymie-Erkennung ein [NJMS12], um die Erkennungsraten weiter zu verbessern. Weiterhin erzielten sie nach eigenen Angaben vielversprechende Ergebnisse bei ersten Experimenten mit unüberwachten Lernalgorithmen zur Toponymcharaktierisierung.

3 Toponymerkennung und Satzanalyse

3.1 Technische Infrastruktur

3.1.1 Grundlagen

Das Sprachanalysewerkzeug GATE wurde bereits in Kapitel 2.2.2 eingehend beschrieben. Aufgrund seiner modularen Architektur und den bereits existierenden Plugins für die Analyse der deutschen Sprache dient GATE als Grundlage für die programmatische Umsetzung der Charakterisierung von Toponymverwendungen im Rahmen dieser Masterarbeit. Hierfür wird die zum Zeitpunkt der Entwicklung aktuelle Version 7.1 vom 30. November 2012 verwendet[1]. GATE ist in der Programmiersprache Java entwickelt und bietet deshalb auch für in dieser Sprache entwickelte Plugins die beste Unterstützung an. Aus diesem Grund sind die funktionalen Erweiterungen, die im Folgenden beschrieben werden, immer in Java implementiert, wenn nichts anderes angegeben ist.

Allerdings sind einzelne der hier und in Abschnitt 3.2 vorgestellten Komponenten im Gegensatz zu Java als Sprache und GATE als Framework plattformabhängig und laufen nur auf einzelnen Architekturen und Betriebssystemen korrekt bzw. bedürfen spezieller Anpassungen. Die Integration und die daraus folgenden Ergebnisse lassen sich deshalb nicht ohne Weiteres auf andere Plattformen übertragen. Die Integration von Fremdkomponenten und die in Kapitel 4 vorgestellte Entwicklung der Toponymcharakterisierung erfolgte auf einer openSUSE 12.3 x86_64-Installation. Als Java-Laufzeitumgebung wurde das Oracle Java SE Development Kit (JDK) 7 verwendet.

3.1.2 Integration abhängiger Komponenten

POS-Tagging mit dem TreeTagger

GATE bietet mit dem CREOLE-Plugin *TaggerFramework* die Möglichkeit, verschiedene externe Programme zur Wortarterkennung ohne zusätzliche Programmierung einzubinden[2]. Hierzu muss ausschließlich eine auf das Tool abgestimmte Konfiguration vorgenommen werden und der Pfad dazu korrekt angegeben werden. Mit der Standardauslieferung bietet GATE bereits verschiedene vorgefertigte Applikationen an, darunter eine für den TreeTagger. Um den TreeTagger korrekt einbinden und für deutsche Texte nutzen zu können, muss er allerdings zusammen mit

[1] `https://gate.ac.uk/download/` - abgerufen am 15. März 2014
[2] `http://gate.ac.uk/sale/tao/splitch22.html#x27-53100022.3` - abgerufen am 15. März 2014

der passenden Parameterdatei für Deutsch heruntergeladen werden[3]. Nach erfolgter Einrichtung kann dessen Funktionalität mit einer der mitgelieferten Anwendungen, die im Verzeichnis `<GATE_DIR>/plugins/Tagger_Framework/resources/TreeTagger/` zu finden sind, getestet werden (`<GATE_DIR>` bezeichnet im Folgenden immer das Installationsverzeichnis von GATE).

Named Entity Recognition mit dem German-Plugin

Ist der TreeTagger erfolgreich eingebunden, ist zunächst nur eine Wortarterkennung für die deutsche Sprache auf Basis des STTS-Tagsets [STT99] möglich. Für eine eingehendere Analyse des Textes sind allerdings noch weitere Komponenten nötig. Aufbauend auf dem TreeTagger bietet GATE hierzu mit dem Plugin *German* bereits im ausgelieferten Paket verschiedene Komponenten zur Analyse deutscher Texte an[4]. Neben der Aufteilung des Textes nach Sätzen und einiger weiterer Werkzeuge ist hier vor allem die Möglichkeit der Named Entity Recognition zu nennen, welche auf diversen JAPE-basierten Regeln und mitgelieferten Gazetteers für Organisationen, Personen, Orten und Eigennamen aufbaut. Beispiele für korrekt konfigurierte Anwendungen des Plugins, sowohl mit einer TreeTagger-Einbindung als auch ohne, werden im Verzeichnis `<GATE_DIR>/plugins/Lang_German/resources/` bereitgestellt.

Erkennung von Nominalphrasen mit dem MuNPEx

Mit der Named Entity Recognition-Komponente von GATE konnten jetzt zwar einzelne Worte oder Wortgruppen einer Organisation oder einem Ort zugeordnet werden, allerdings beinhaltet bislang keine Pipeline eine Komponente zur morphologischen Analyse des Textes. Als Vorstufe werden hierzu in der Computerlinguistik zunächst die einzelnen Bauelemente eines Satzes analysiert. Diese werden in der Regel als *Phrasen* bezeichnet. Phrasen können entweder aus einem einzelnen oder aus mehreren Wörtern bestehen, die an ein entsprechendes *Kernwort* gebunden sind [KÖ5]. Man unterscheidet hierbei verschiedene Phrasentypen, wobei die sogenannte Nominalphrase von zentraler Bedeutung für die weitere (morphologische) Satzanalyse ist. Eine Nominalphrase, englisch *noun phrase*, ist wiederum der Überbegriff für Substantivphrasen und Pronominalphrasen. Eine Substantivphrase beinhaltet ein Substantiv, eine Pronominalphrase ein Pronomen als Kern. Hierfür folgen nun einige Beispiele, jeweils mit einem Wort bzw. mit mehreren, die sich um einen Kern gruppieren:

Substantivphrasen

Spanien ist schön. (Einzelnes Wort)
Die **Schweiz** hat vier Amtssprachen. (Substantiv mit Artikel)
Das schöne **Heidelberg** bietet viel. (Substantiv mit Artikel und Adjektiv)

Pronominalphrasen

Ich fahre nach Dänemark. (Einzelnes Wort)
Du *selbst* wolltest doch nach Berlin fliegen. (Zwei Pronomen)

[3] `http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/` - abgerufen am 15. März 2014
[4] `http://gate.ac.uk/sale/tao/splitch15.html#x20-40900015.3` - abgerufen am 15. März 2014

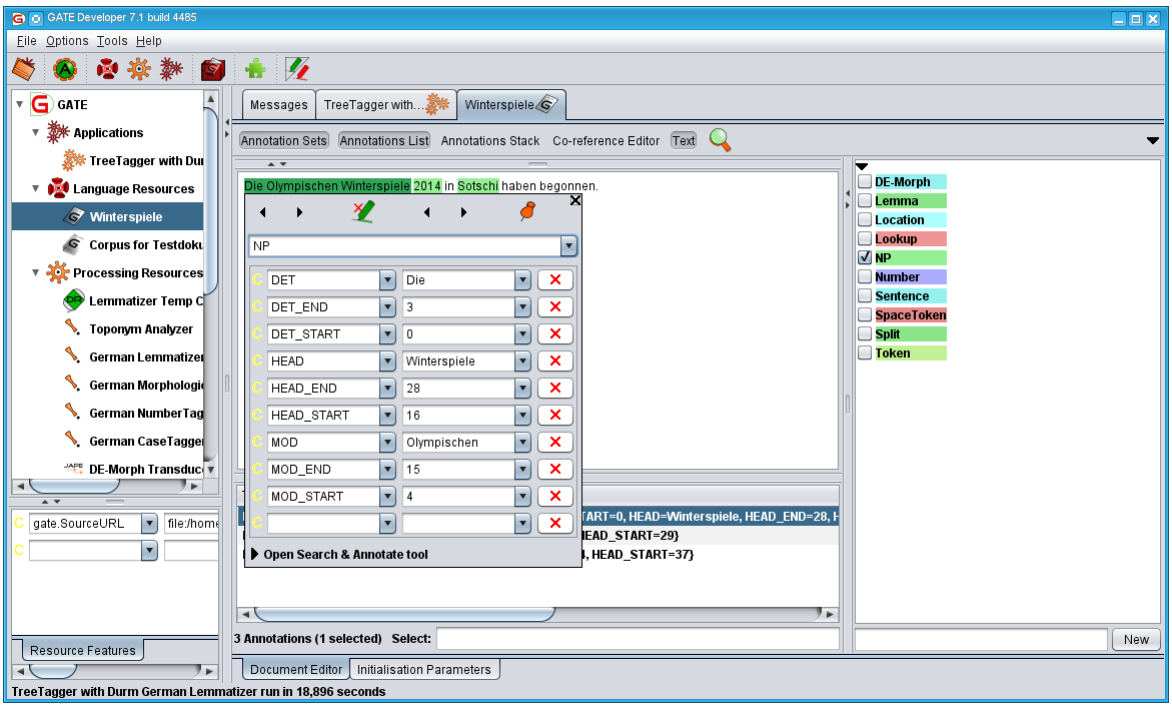

Abbildung 3.1 — Annotationen des MuNPEx-Plugins für Nominalphrasen des Beispielsatzes

Zur Erkennung solcher Nominalphrasen bietet das Semantic Software Lab der Concordia University in Montréal (Kanada) den *Multi-lingual Noun Phrase Extractor* (MuNPEx) als Plugin für GATE an[5]. MuNPEx unterstützt unter anderem die deutsche Sprache, weswegen diese Software auch als Komponente eingebunden wird. Es ist allerdings nicht integraler Bestandteil von GATE und muss deshalb getrennt heruntergeladen und an einen selbst definierten Ort für Fremd-Plugins installiert werden. Nach erfolgter Einrichtung kann aber auch hier eine mitgelieferte Beispielanwendung genutzt werden. Abbildung 3.1 zeigt nun, welche Nominalphrasen MuNPEx im Beispielsatz erkennen konnte. Hervorgehoben ist die Nominalphrase „Die Olympischen Spiele“, bei der „Winterspiele“ als Kernwort (hier: `HEAD`) identifiziert wurde. Die zwei übrigen Worte gruppieren sich darum und wurden auch entsprechend kategorisiert. „Olympischen“ wurde als Adjektivphrase (hier: `MOD`) und „Die“ als Artikel- bzw. Determinansphrase (hier: `DET`) erkannt.

Morphologische Analyse mit dem Durm German Lemmatization System

Das Durm German Lemmatization System basiert sowohl auf dem TreeTagger als auch auf MuNPEx und ist in der Lage, neben einer im Vergleich zum TreeTagger verbesserten Lemmatisierung (vgl. auch [PW05]) zusätzlich auch eine morphologische Analyse für deutsche Texte durchführen zu können. Hierbei können sowohl Kasus, als auch Numerus und Genus der einzelnen Satzteile erkannt werden. Die genannten Komponenten werden wie MuNPEx vom Semantic Software Lab zum Download angeboten, allerdings sind zur Nutzung in einer GATE-Anwendung einige ma-

[5] `http://www.semanticsoftware.info/munpex` - abgerufen am 15. März 2014

nuelle Schritte notwendig[6]. Ein Beispielergebnis dieses Systems wurde bereits in Abbildung 2.6 gezeigt, die Ausgabe erfolgt hier über die Annotation `DE-Morph`.

3.1.3 Vor- und Nachteile der bestehenden Komponenten

Die vorgestellten Komponenten machen im Wesentlichen die Werkzeuge aus, die nach eingehender Recherche verfügbar und die für die Problemstellung dieser Masterarbeit von Relevanz sind. Zusammenfassend ist festzustellen, dass speziell mit der Integration des TreeTaggers, des GATE-Plugins für die deutsche Sprache und der morphologischen Satzanalyse mit dem Durm German Lemmatization System bereits zentrale Teile der in Kapitel 2.2.2 vorgestellten Werkzeuge ohne zusätzlichen Entwicklungsaufwand für GATE verfügbar sind und Texte damit entsprechend analysiert werden können. Allerdings konnte auch nach umfangreichen Nachforschungen keine GATE-Komponente gefunden werden, die eine grammatikalische Abhängigkeitsanalyse für deutsche Texte unterstützt oder eine externe Komponente wie den zuvor beispielhaft angeführten ParZu einbindet. Es existiert zwar eine über das TaggerFramework realisierte Anbindung an den *Stanford Natural Language Parser* (Stanford NLP)[7], allerdings unterstützt dieser für Deutsch auch in der zum Zeitpunkt der Erstellung dieser Arbeit aktuellen Version 3.3.1 keine Abhängigkeitsanalyse[8]. Aktuell ist ausschließlich eine Wortart-, sowie eine Phrasenerkennung implementiert, die bereits 2008 von Rafferty und Manning anhand von Referenzkorpora evaluiert wurden [RM08]. Wie die grafische Ausgabe des Stanford NLP in Abbildung 3.2 zeigt, beschränkt er sich bei den Phrasen allerdings nicht nur auf Nominalphrasen wie MuNPEx, sondern erkennt zudem auch andere wie Verb- oder Präpositionalphrasen.

Darüber hinaus konnten in ersten Tests bereits einige Schwachstellen der GATE-eigenen Named Entity Recognition für deutsche Texte ausgemacht werden. So wurde nicht nur der Ort *Sotschi* im Beispielsatz nicht als solcher erkannt. Auch in weiteren exemplarisch überprüften Texten wurden insbesondere kleinere Orte und Toponyme außerhalb des deutschen Sprachgebiets nicht erkannt. Wie eine Analyse der verfügbaren grammatikalischen Regeln und Wörterbüchern in den Unterverzeichnissen `gazetteer` und `grammar` des deutschen Sprachplugins (`<GATE_DIR>/plugins/Lang_German/resources/`) zeigt, scheint die Ursache hauptsächlich an der eingeschränkten Anzahl an hinterlegten Ortsnamen und feingranularen Regeln zu liegen. So beinhaltet der Gazetteer für Städte nur knapp über 2.000 Einträge, wobei hier oft noch unterschiedliche Schreibweisen eines Ortes berücksichtigt wurden. So liegt die Anzahl tatsächlich abgedeckter Städte noch weit darunter. Ähnlich verhält es sich mit der Liste für Länder und sonstige geographische Bezüge.

Da eine erfolgreiche Charakterisierung von Toponymverwendungen zunächst eine gute Erkennung von Toponymen voraussetzt, werden im Folgenden zunächst Möglichkeiten vorgestellt, GATE diesbezüglich zu erweitern und seine Erkennungsrate zu verbessern. Weiterhin wird eine Möglichkeit vorgestellt, wie die Ergebnisse der Abhängigkeitsanalyse mit ParZu in GATE genutzt werden können. Eine direkte Integration der eingangs vorgestellten GermaNet-Datenbank

[6] `http://www.semanticsoftware.info/durm-german-lemmatizer` - abgerufen am 15. März 2014
[7] `http://nlp.stanford.edu/downloads/lex-parser.shtml` - abgerufen am 15. März 2014
[8] `http://nlp.stanford.edu/downloads/parser-faq.shtml#c` - abgerufen am 15. März 2014

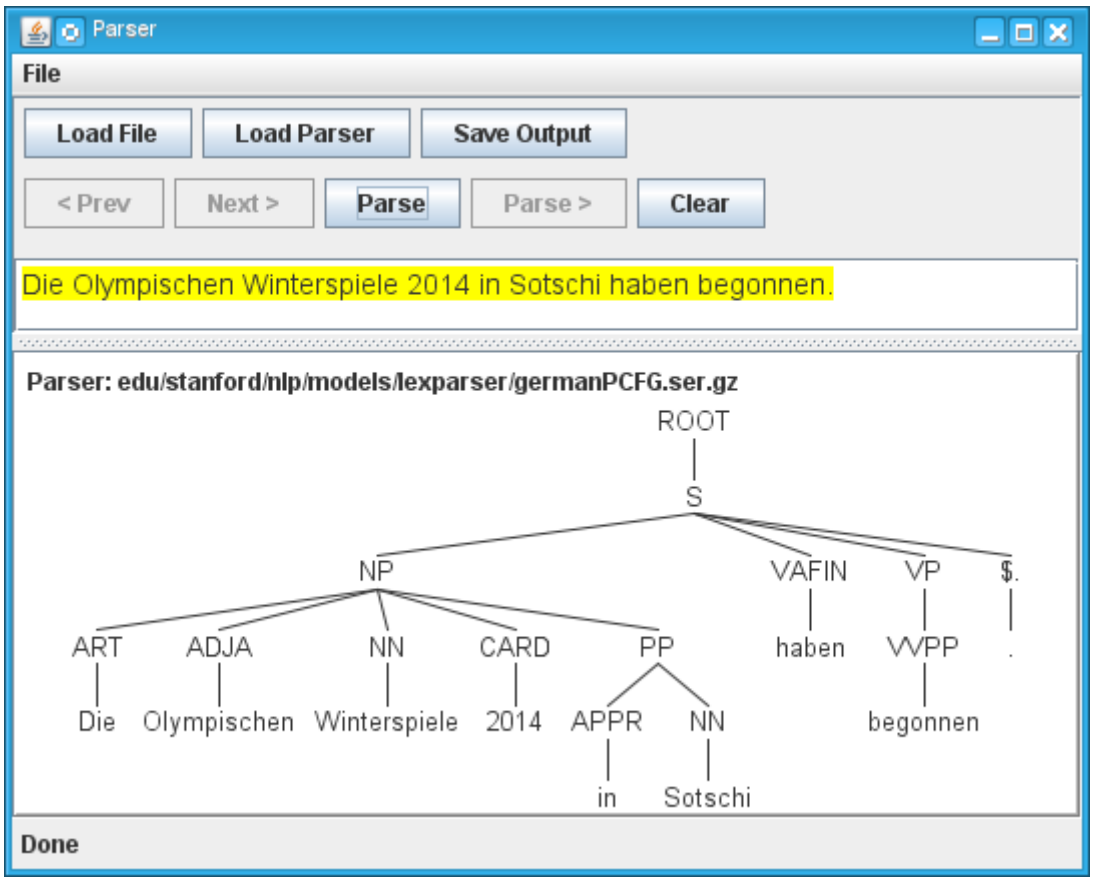

Abbildung 3.2 — Grafische Ausgabe des Stanford NLP für den Beispielsatz

ist an dieser Stelle allerdings noch nicht nötig, da diese Informationen erst in der in Kapitel 4 vorgestellten Charakterisierung von bereits erkannten Toponymen relevant sind.

3.2 Weiterentwicklung der Infrastruktur

3.2.1 Integration von ParZu

In Kapitel 2.2.2 wurden bereits am Beispiel des *Zurich Dependency Parser for German* (ParZu) einige Möglichkeiten aufgezeigt, die eine Abhängigkeitsanalyse für die Charakterisierung von Toponymverwendungen bietet. Zwar können viele der in Kapitel 2.1 vorgestellten Stilmittel auch ohne eine entsprechende Komponente erkannt werden, allerdings können sich hier speziell in komplexen Satzkonstruktionen, wie etwa mit Haupt- und Nebensätzen, Fehler bemerkbar machen, die mit einer entsprechenden Analyse vermeidbar gewesen wären. Weiterhin gibt es eine relevante Zahl an Charakteristika, die ausschließlich auf Basis der Abhängigkeiten eingeordnet werden können. Aus diesen Gründen erfolgte zur Textanalyse die Integration von ParZu in die GATE-Infrastruktur.

ParZu kann als freie Software im Quelltext aus dem GitHub-Projekt des Hauptautores Rico Sennrich heruntergeladen werden[9]. Allerdings sind zum Einrichten der Software einige Abhängigkeiten zu erfüllen, unter anderem wird ein Linux-Betriebssystem vorausgesetzt. Zur morphologischen Analyse eines Textes kann sich ParZu aus aktuell drei verschiedenen Alternativen bedienen. Die erste Alternative ist die kommerzielle Software GERTWOL der Firma Lingsoft, welche allerdings nicht weiter empfohlen wird und in Zukunft nicht mehr weiter unterstützt werden könnte. Als zweite Möglichkeit kann die quelloffene Software Mophisto[10] eingesetzt werden, was zu Beginn der Ausarbeitung dieser Arbeit noch die empfohlene Variante war. Die dritte, mittlerweile vom Autor empfohlene Variante ist die Nutzung der von Sennrich entwickelten Software *Zurich Morphological Analyzer for German* (Zmorge), deren Grammatikdefinitionen wie Morphisto auf SMOR (vgl. [SFH04]) beruhen, hier aber einigen Modifikationen unterliegen und dessen Lexikon auf Wiktionary[11] aufbaut. Da die Arbeit an der in Kapitel 4 vorgestellten Implementierung bereits früher begann und die Grundlagen zu Zmorge erst Ende Mai 2014 auf der Language Resources and Evaluation Conference (LREC) in Reykjavík vorgestellt werden[12], basieren die Ergebnisse dieser Arbeit ausschließlich auf ParZu mit Morphisto.

Da es das in Abschnitt 3.1.2 kurz beschriebene CREOLE-Plugin TaggerFramework erlaubt, externe Anwendungen in GATE einzubinden, wird dies zur Integration von ParZu verwendet. Der ParZu-Tagger soll hierbei in einer GATE-Anwendung nach erfolgter Tokenisierung des zu analysierenden Textes eingebunden werden. Um dies zu ermöglichen, müssen nach erfolgter Einrichtung von ParZu einige Parameter in der Datei `<PARZU_DIR>/config.ini` angepasst werden. Der Parameter `morphology` wird auf `morphisto` gesetzt, zudem wird der korrekte Pfad zum Morphisto-Modell angegeben. Hier war die Version vom 02. Februar 2011 (`morphisto-02022011.a`) aktuell. Als Eingabeformat wird `tokenized`, als Ausgabeformat `conll` verwendet. Weitere Anpassungen sind an der Konfiguration nicht notwendig, allerdings kann das mit ParZu ausgelieferte Skript `parzu` nicht direkt als *taggerBinary* im TaggerFramework-Plugin verwendet werden, da so die von GATE ermittelten Token nicht korrekt zur weiteren Prozessierung übergeben werden können. Hierfür wird stattdessen ein neues Shellskript namens `parzu-gate` verwendet, dessen Quelltext in Abbildung 3.3 dargestellt wird.

Wie der Quelltext zeigt, ist zur korrekten Einbindung von ParZu in GATE nach der entsprechenden Anpassung der Konfigurationsdatei im Grundsatz nur noch das vom TaggerFramework-Plugin ermittelte Token entsprechend der von ParZu verlangten Syntax `./parzu < Token` als Argument zu übergeben. Die Ausgabe der Analyse erfolgt dann auf der Kommandozeile, die von GATE anhand bestimmter Muster ausgewertet werden kann.

In die bereits bestehende GATE-Anwendung (vgl. Abschnitt 3.1.2) wird eine neue Instanz der Processing Resource (PR) *GenericTagger* nach dem *GATE Unicode Tokenizer* und dem *ANNIE Sentence Splitter* eingefügt. Somit ist sichergestellt, dass bereits Annotationen des Typs `Token` existieren, welche dann mit neuen Features auf Basis der ParZu-Ergebnisse ausgestattet werden. Hierzu muss in der Konfiguration der PR sowohl der `inputAnnotationType` als auch der `outputAnnotationType` auf `Token` sowie der Parameter `updateAnnotations` auf `true` geändert

[9] `https://github.com/rsennrich/ParZu` - abgerufen am 17. März 2014
[10] `http://code.google.com/p/morphisto/` - abgerufen am 17. März 2014
[11] `http://www.wiktionary.org/`
[12] `http://lrec2014.lrec-conf.org/en/conference-programme/list-accepted-papers/` - abgerufen am 17. März 2014

```
#!/bin/sh
# ParZu connector for GATE Developer

# Set the correct location of your ParZu installation here
PARZU_DIR=/home/ygriega/Dokumente/Master-Studium/Master-Arbeit/ParZu-master/

cd ${PARZU_DIR}
LD_LIBRARY_PATH=.; export LD_LIBRARY_PATH
./parzu < $1
```

Abbildung 3.3 — Quelltext des Shellskripts `parzu-gate`

Runtime Parameters for the "ParZu Tagger" GenericTagger:

Name	Type	Required	Value
debug	Boolean	✓	false
encoding	String	✓	utf-8
failOnMissingInputAnnotations	Boolean		true
failOnUnmappableCharacter	Boolean	✓	false
featureMapping	FeatureMap	✓	{morph=6, coarseTag=4, position=1, headPosition=7, dependencyTag=5, dependencyLemma=3, string=2, class=8}
inputASName	String		
inputAnnotationType	String	✓	Token
inputTemplate	String	✓	${string}
outputASName	String		
outputAnnotationType	String	✓	Token
regex	String	✓	(.+) (.+) (.+) (.+) (.+) (.+) (.+) (.+) (.+) (.+)
taggerBinary	URL	✓	file:/home/ygriega/Dokumente/Master-Studium/Master-Arbeit/ParZu-master/parzu-gate
taggerDir	URL		
taggerFlags	List	✓	[]
updateAnnotations	Boolean	✓	true

Abbildung 3.4 — Konfigurierte GenericTagger Processing Resource für ParZu

werden, falls dies nicht bereits voreingestellt ist. Das Encoding des Textes wird darüber hinaus auf `utf-8` verändert und der Pfad zum vorgestellten Shellskript `parzu-gate` korrekt dem Parameter `taggerBinary` übergeben.

Die zentralen Anpassungen erfolgen allerdings an den Parametern `regex` und `featureMapping`. Die Ausgabe im CoNLL-Format erfolgt je Token in einer Zeile, wobei die einzelnen erkannten Features mit einem Tabulator voneinander unterschieden werden[13]. Aktuell unterstützt ParZu in dieser Ausgabe zehn verschiedene Felder, einen Auszug hiervon zeigt Abbildung 2.2. Diese Felder können nun mittels des dazu passenden regulären Ausdrucks

```
(.+) (.+) (.+) (.+) (.+) (.+) (.+) (.+) (.+) (.+)
```

als `regex`-Parameter übernommen werden. Ein einzelnes Feld, das aus mindestens einem beliebigen Zeichen bestehen kann, wird hierbei mit einer Klammer umschlossen. Einzelne Felder werden

[13] `http://nextens.uvt.nl/depparse-wiki/DataFormat` - abgerufen am 17. März 2014

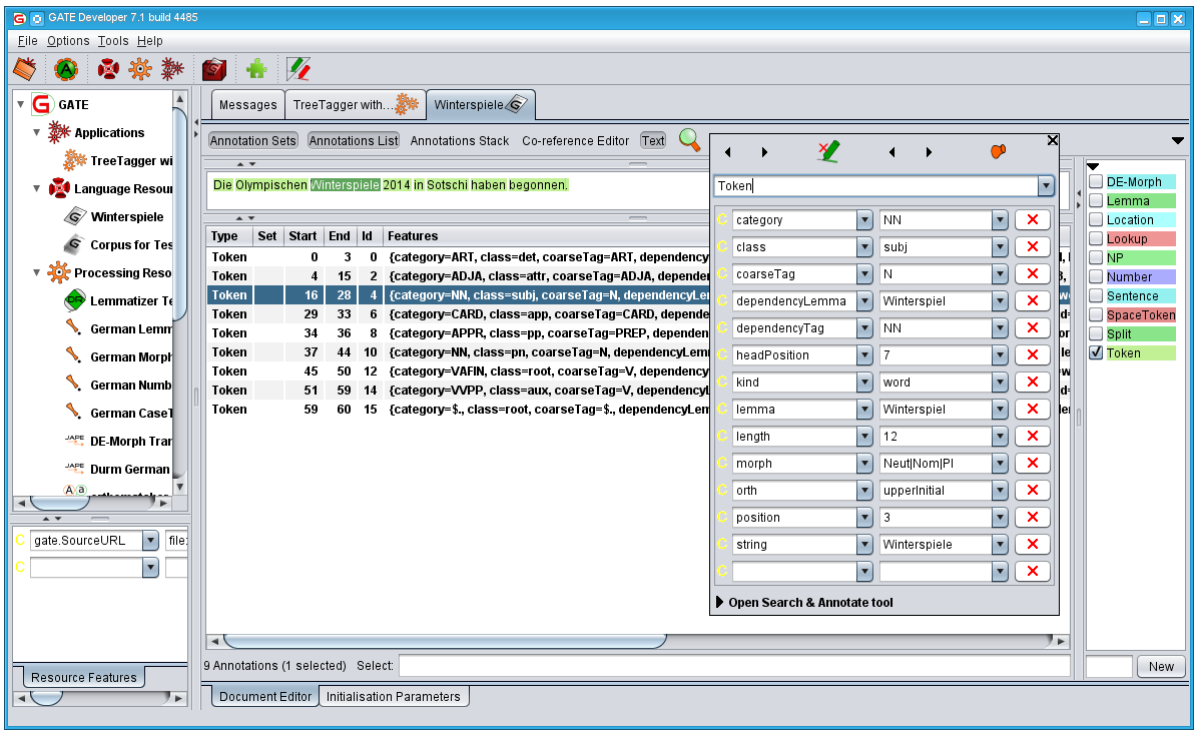

Abbildung 3.5 — Features einer Token-Annotation nach erfolgreicher ParZu-Integration

auch im regulären Ausdruck direkt mit einem Tabulatorzeichen getrennt. Die Verwendung einer Escape-Sequenz wie `\t` ist nicht notwendig, da GATE die Tabulatorzeichen direkt verwenden kann. Die einzelnen Felder werden vom TaggerFramework-Plugin dann Nummern zugeordnet, die mit dem `featureMapping`-Parameter entsprechenden Features zugeordnet werden können. Abbildung 3.4 zeigt die komplette Konfiguration. Für die einzelnen Felder wurden im Vergleich zur Dokumentation leicht unterschiedliche Namen verwendet, um einerseits die Verständlichkeit zu erhöhen und andererseits mit den Annotationen anderer Komponenten wie der direkten TreeTagger-Integration in GATE nicht in Konflikt zu geraten. Abbildung 3.5 zeigt ein Token des Beispielsatzes nach erfolgreicher ParZu-Integration.

3.2.2 Integration des Stanford NER Taggers

Wie bereits in Kapitel 2.2.2 gezeigt, kann die Textanalyse speziell mit Hinblick auf erkannte Toponyme vom Stanford NER-Tagger mit einem Klassifizierer für die deutsche Sprache profitieren. Allerdings existiert auch hier keine direkte Integration für GATE. Diese muss deshalb neu implementiert werden. Die Integration erfolgt hier analog zu der des ParZu-Taggers mit einer neuen Instanz der GenericTagger-PR. Verwendet wird die zurzeit aktuelle Version 3.3.1 vom 04. Januar 2014.

Beim Stanford NER-Tagger sind zwar im Vergleich zu ParZu keine umfangreichen Konfigurationsschritte nötig, um dessen grundsätzliche Funktion sicherzustellen, allerdings sind auch hier beim Aufruf bestimmte Parameter zu berücksichtigen. So muss der Tagger zum einen von der

```
#!/bin/sh
# Stanford NER Tagger connector for GATE Developer

# set the correct location of your Stanford NER installation here
NER_DIR=/home/ygriega/Dokumente/Master-Studium/Master-Arbeit/Stanford/
stanford-ner-2014-01-04/

cd ${NER_DIR}
LD_LIBRARY_PATH=.; export LD_LIBRARY_PATH
java -Xms1g -cp "stanford-ner.jar:." edu.stanford.nlp.ie.crf.CRFClassifier
-loadClassifier classifiers/dewac_175m_600.crf.ser.gz -testFile $1
```

Abbildung 3.6 — Quelltext des Shellskripts `stanford_ner-gate`

Java-Laufzeitumgebung mit ausreichend Speicher versorgt werden, zum anderen akzeptiert er nur dann Texte im von GATE verwendeten Format, wenn der Parameter `-testFile` verwendet wird, der eigentlich zum Training des Modells genutzt wird. Der entsprechend aufgebaute Aufruf wird hier wiederum über ein angepasstes Shellskript verschalt, dessen Quelltext in Abbildung 3.6 dargestellt wird.

Wie der Aufruf zeigt, wird analog zum Beispiel aus Abbildung 2.5 auch hier der deWaC-Klassifizierer verwendet, da der Tagger mit verschiedenen Textgenres gut funktionieren soll und keine Fokussierung auf Nachrichtentexte angestrebt wird. Der Stanford NER-Tagger gibt die Ergebnisse des Prozesses in einem ähnlichen Format aus wie der TreeTagger und die entsprechend konfigurierte ParZu-Komponente. Im Unterschied zu ParZu beinhaltet die Ausgabe hier nur drei Felder, von denen für die weitere Analyse ausschließlich die dritte relevant ist. Damit ausschließlich erkannte Toponyme durch die neu konfigurierte GenericTagger-PR weiterverarbeitet werden, wird deshalb darauf geachtet, dass nur solche Token, welche mit `I-LOC` versehen wurden, durch den regulären Ausdruck des `regex`-Parameters erkannt werden. Als Featurewert soll hierbei ausschließlich das Schlüsselwort `LOC` übernommen werden. Der passende reguläre Ausdruck lautet deshalb wie folgt:

```
(.+) (.+) I-(LOC)
```

Als Zeichensatz wird hier wie auch bei der ParZu-Komponente UTF-8 verwendet und die bestehenden Token-Annotationen als Eingabewerte verwendet. In der Anwendungspipeline muss diese Komponente deshalb auch wieder nach dem GATE Unicode Tokenizer und dem ANNIE Sentence Splitter einsortiert werden. Die Ausgabe der Featurewerte erfolgt allerdings nicht in den bestehenden `Token`-Annotationen, sondern in neuen Annotationen des Typs `Location`, weswegen die Parameter `ouputAnnotationType` auf `Location` und `updateAnnotations` auf `false` gesetzt werden müssen. Diese Herangehensweise hat den Vorteil, dass in der weiteren Analyse durch die Named Entity Recognition mit dem German-Plugin (vgl. Abschnitt 3.1.2) keine doppelten Annotationen für einen Ortsbezug erstellt werden, sondern ausschließlich vom Stanford NER nicht erkannte Toponyme mit einer neuen `Location`-Annotation versehen werden. Darüber hinaus kann bei der Charakterisierung von Toponymverwendungen davon ausgegangen

Runtime Parameters for the "Stanford NER" GenericTagger:

Name	Type	Required	Value
debug	Boolean	✓	false
encoding	String	✓	utf-8
failOnMissingInputAnnotations	Boolean		false
failOnUnmappableCharacter	Boolean	✓	false
featureMapping	FeatureMap	✓	{kind=3, string=1}
inputASName	String		
inputAnnotationType	String	✓	Token
inputTemplate	String	✓	${string}
outputASName	String		
outputAnnotationType	String	✓	Location
regex	String	✓	(.+) (.+) I-(LOC)
taggerBinary	URL	✓	file:/home/ygriega/Dokumente/Master-Studium/Master-Arbeit/Stanford/stanford-ner-2014-01-04/stanford_ner-gate
taggerDir	URL		
taggerFlags	List	✓	[]
updateAnnotations	Boolean	✓	false

Abbildung 3.7 — Konfigurierte GenericTagger Processing Resource für Stanford NER

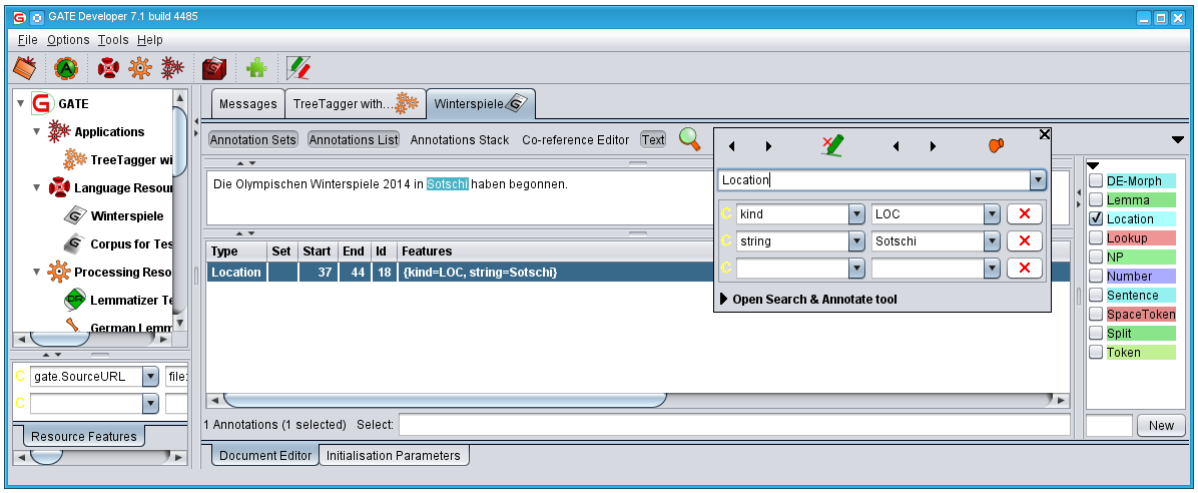

Abbildung 3.8 — Features einer Location-Annotation nach erfolgreicher Stanford NER-Integration

werden, dass die erkannten Toponyme ausschließlich über diesen Annotationstyp identifiziert werden können. Abbildung 3.8 zeigt das Ergebnis einer erfolgreichen Integration des Stanford NER-Taggers in GATE anhand des Beispielsatzes. *Sotschi* wurde jetzt korrekt als Toponym erkannt.

3.2.3 Erweiterung des GATE Gazetteers

Einführung

Auch mit der Kombination aus dem ANNIE-Named Entity Recognition-System von GATE und dem neu integrierten Stanford NER-Tagger werden immer noch einige Toponyme durch die be-

stehende Anwendungspipeline nicht erkannt. Wie in Abschnitt 3.1.3 bereits erwähnt, beinhalten die Gazetteers des GATE-Plugins für die deutsche Sprache vergleichsweise wenig Einträge, die zum Teil fehlerhaft kodiert sind, was zu einer niedrigen Erkennungsrate führt. Die fehlerhafte Kodierung lässt sich automatisiert einfach beheben, die fehlenden Einträge müssen aber über andere Quellen bezogen und so die Listen erweitert werden. Da sich diese Arbeit ausschließlich mit Texten in deutscher Sprache beschäftigt, liegt der Fokus zunächst auf der Verbesserung der Erkennungsraten für Orte im deutschsprachigen Raum. Das Projekt *OpenGeoDB* hat genau diesen Fokus und bietet darüber hinaus auch noch Informationen zu entsprechenden Geokoordinaten und Postleitzahlen[14]. Ziel ist es nun, die Ortsdatenbank von OpenGeoDB so als neuen Gazetteer in GATE zu integrieren, dass passende Toponyme automatisch vom German-Plugin (vgl. Abschnitt 3.1.2) mit einer `Location`-Annotation versehen werden.

Ausgangspunkt für diese Integration ist ein aktueller Abzug der OpenGeoDB-Datenbank. Für diese Arbeit wurde der Stand vom 18. September 2013 verwendet[15]. Da der Datenbankabzug als SQL-Export vorliegt und ANNIE die Gazetteers als einfache Listen mit jeweils einem Element pro Zeile verlangt[16], müssen die Einträge noch entsprechend konvertiert werden, bevor sie in GATE verwendet werden können. Hierfür wurde ein eigenes Java-Programm namens `OpenGeoDBExtractor` entwickelt, welches im Folgenden kurz vorgestellt wird.

Stoppwortliste

Bei einer direkten Übernahme aller Einträge aus der Datenbank in den Gazetteer könnten sich in einigen Fällen Probleme durch Geo/Non-Geo-Ambiguitäten ergeben, da es einige Orte gibt, deren Namen in der deutschen Sprache in einem anderen, nicht geographischen Kontext sehr gebräuchlich sind. Wenn diese Worte in den Gazetteer aufgenommen werden würden, würde sich somit neben der Erkennungsrate auch die Fehlerhäufigkeit durch falsch als Toponyme klassifizierte Worte erhöhen. Um dies auszuschließen, sollen nur die Datenbankeinträge übernommen werden, die nicht den am häufigsten verwendeten Worten der deutschen Sprache zuzuordnen sind. Die Universität Leipzig hat hierzu diverse Listen der häufigsten Worte der deutschen, englischen, niederländischen und französischen Sprache veröffentlicht[17]. Um möglichst viele Ambiguitäten durch den neuen Gazetteer zu vermeiden, wird hierbei auf die Liste der 10.000 häufigsten Wörter der deutschen Sprache zurückgegriffen und alle Einträge von `OpenGeoDBExtractor` als Stoppworte bei der Datenbankextraktion behandelt. Sollte es also gleichlautende Einträge in OpenGeoDB geben, werden diese ignoriert.

Extraktion der Ortsnamen

Die Ortsnamen werden in OpenGeoDB im Feld `text_val` der Tabelle `geodb_textdata` abgelegt[18]. Hierbei gibt es verschiedene Varianten, etwa eine, die ausschließlich aus Großbuchstaben besteht. Die Alternativen sind hierbei über das Feld `text_type` eindeutig zu identifizieren, wobei

[14] `http://opengeodb.org/wiki/OpenGeoDB` - abgerufen am 18. März 2014
[15] `http://fa-technik.adfc.de/code/opengeodb/dump/` - abgerufen am 21. Dezember 2013
[16] `http://gate.ac.uk/sale/tao/splitch13.html#sec:gazetteers:anniegaz` - abgerufen am 18. März 2014
[17] `http://wortschatz.uni-leipzig.de/html/wliste.html` - abgerufen am 21. Dezember 2013
[18] `http://opengeodb.org/images/e/ed/Opengeodb-db-modell.png` - abgerufen am 18. März 2014

die gängige Schreibweise eines Ortes unter der ID `500100000` abgelegt wird. Mit diesen Informationen lässt sich der Ortsname aus dem entsprechenden SQL-Befehl programmatisch extrahieren, hier ein Beispiel für den entsprechenden Eintrag der Stadt *Walldorf (Baden)*:

```
INSERT INTO geodb_textdata VALUES(25611,500100000,'Walldorf (Baden)','de',1,1,
null,null,'3000-01-01',300500000);
```

Sonderbehandlungen

Dieses Beispiel zeigt, dass es auch bei Einträgen, die nicht auf der Stoppwortliste stehen, bestimmte Sonderbehandlungen nötig sein können. Das Beiwort *Baden* ist zum Beispiel für eine Toponymauflösung, allerdings nicht für die bloße Toponymerkennung, relevant. Da es bei der Erweiterung der Gazetteers ausschließlich um die Erkennung geht, können diese Beiworte bei der Aufnahme ignoriert werden. Weiterhin gibt es einige Orte, die nach Heiligen benannt sind, etwa *St. Anna*. Da diese Orte ohne das voran gestellte *St.* in vielen Fällen einen gängigen Personennamen bezeichnen, der sich aber ggf. nicht in der Stoppwortliste befindet, kann es hier auch zu falsch erkannten Treffern kommen. Aus diesem Grund werden alle Einträge, die auf den Namen eines Heiligen hindeuten, ebenfalls nicht in den neuen Gazetteer aufgenommen.

Auf der anderen Seite gibt es einige Orte, die wie z.B. Doppelnamen aus mehreren Teilen bestehen und deren Bestandteile in Texten durchaus auch unabhängig voneinander vorkommen können, etwa wenn explizit nur ein Teilort gemeint ist. Aus diesem Grund werden in solchen Fällen alle Teile zusammen mit dem eigentlichen Datenbankeintrag in den Gazetteer übernommen. Als Teiler werden hierbei Bindestriche, Leerzeichen und Kommata akzeptiert. Allerdings müssen auch bei dieser Aufteilung gewisse Eigenarten von Orten berücksichtigt werden. So beinhalten einige Orte einen Präfix wie *Oberes* oder *Klein* in den entsprechend möglichen Deklinationen. Mit Hilfe einer weiteren Stoppwortliste werden diese ortsspezifischen Sonderfälle von einer Aufnahme in den Gazetteer ausgenommen.

Implementierung

Die Implementierung des Programms `OpenGeoDBExtractor` besteht aus den zwei Klassen `OpenGeoDBExtractor` und `OpenGeoDB`. Die Extraktorklasse übernimmt hierbei nach dem Start das Einlesen der Stoppwortliste der häufigsten deutschen Wörter sowie das Parsen des Datenbankabzugs von OpenGeoDB. Ist ein Datenbankeintrag, ggf. nach der Anwendung von den angesprochenen Sonderregeln, als neuer Eintrag für den Gazetteer bestimmt, wird dieser an die Klasse `OpenGeoDB` übergeben. Dies geschieht jedoch nur, nachdem überprüft wurde, ob der Eintrag noch nicht vorhanden ist oder keine ortsspezifischen Sonderfälle zu betrachten sind. Der ermittelte Ort wird dabei nicht nur mit seinem eigentlichen Namen in die interne Datenstruktur übernommen, sondern zusätzlich nochmal mit dem Suffix *er* und *s* für attributive Verwendungen wie **Bamberger** INNENSTADT oder **Stuttgarts** BAHNHOF, sodass pro Toponym drei Einträge geschrieben werden. Sind alle Datenbankeinträge ausgelesen worden, ruft `OpenGeoDBExtractor` die komplette Liste der ermittelten Gazetteereinträge von der `OpenGeoDB`-Klasse ab und speichert sie in eine

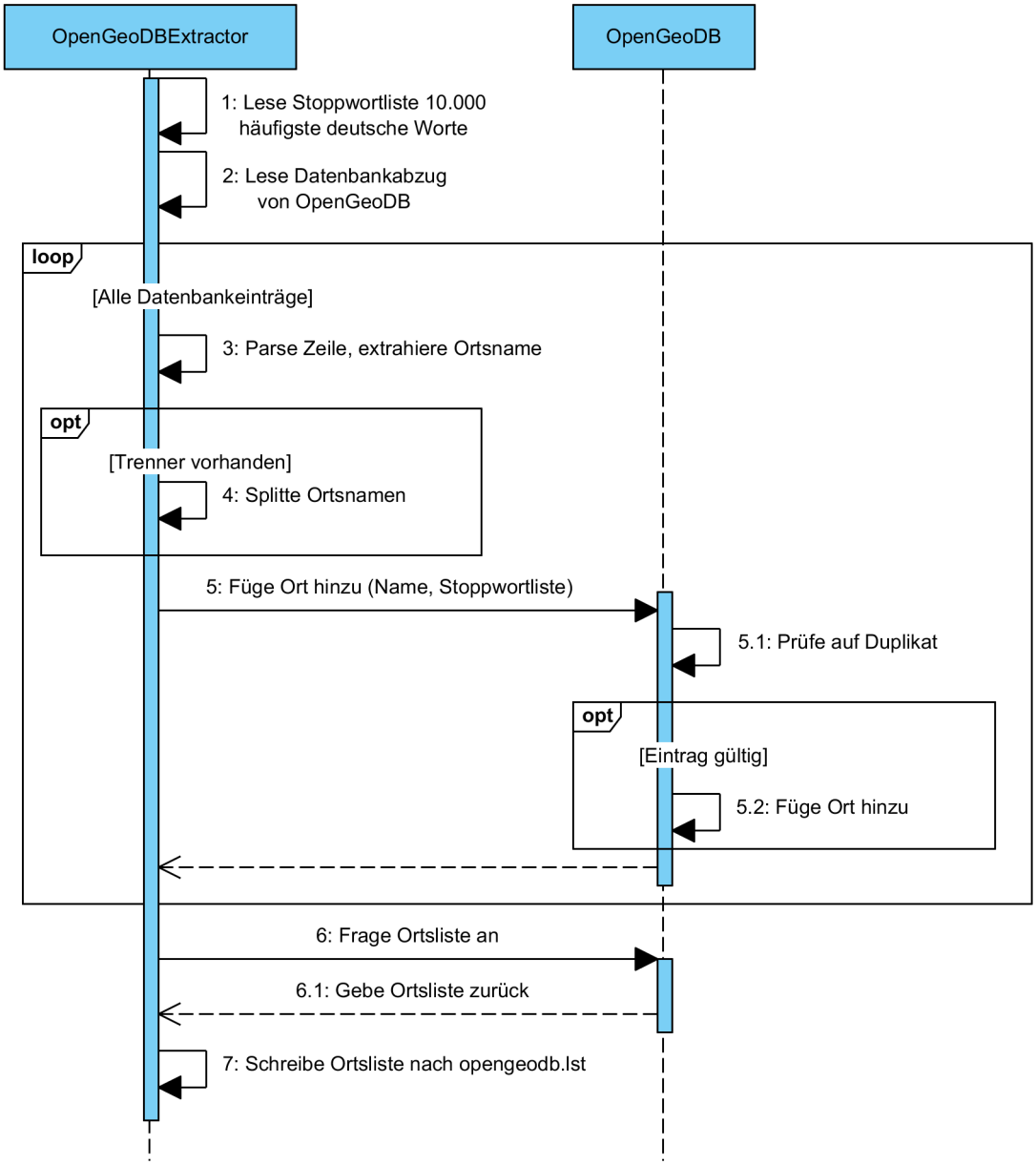

Abbildung 3.9 — Sequenzdiagramm des Programms `OpenGeoDBExtractor`

Textdatei mit dem Namen `opengeodb.lst`. Das Sequenzdiagramm 3.9 veranschaulicht diesen Prozess.

Die fertige Liste beinhaltet dann etwas mehr als 182.000 Einträge, womit die mit GATE ausgelieferten Gazetteers um ein Vielfaches erweitert werden können. Allerdings muss sie dazu noch in die bestehende Infrastruktur integriert werden.

```
Rule: OpenGeoDB
(
    ({Lookup.minorType == "opengeo"})
):geoname
-->
:geoname.Location =
    {kind = "Location", subtype = "geoLocation", rule = "OpenGeoDB"}
```

Abbildung 3.10 — Quelltext des JAPE-Regel für OpenGeoDB-Einträge

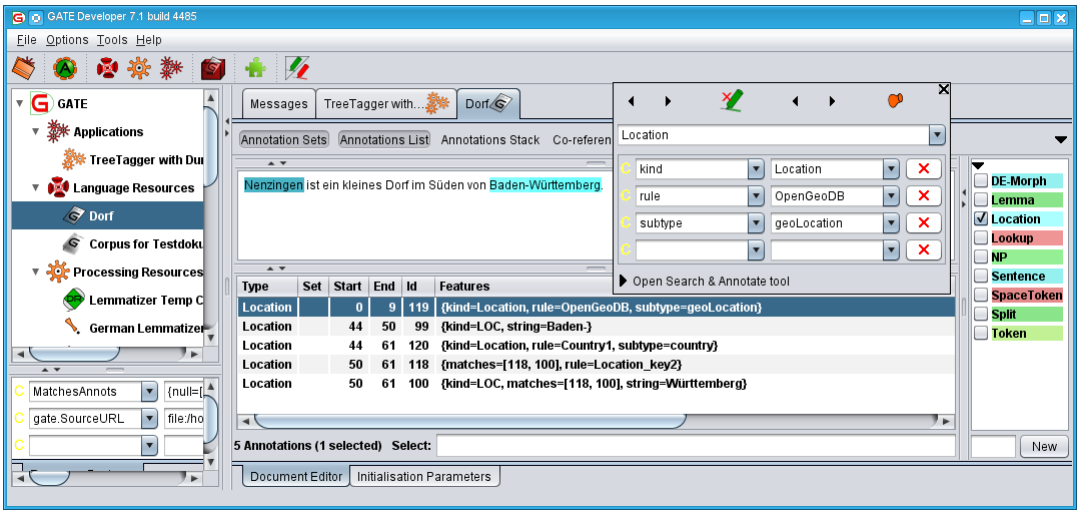

Abbildung 3.11 — `Location`-Annotation für einen mittels OpenGeoDB erkannten Ort

Integration in das bestehende GATE-Plugin

Die Integration erfolgt in mehreren Schritten. Zunächst muss die Datei `opengeodb.lst` in das Verzeichnis `<GATE_DIR>/plugins/Lang_German/resources/gazetteer/` kopiert werden und dem Plugin mit dem folgenden neuen Eintrag in der Datei `lists.def` im gleichen Verzeichnis bekannt gemacht werden:

```
opengeodb.lst:location:opengeo:german
```

Diese Zeile besagt, dass in der Datei `opengeodb.lst` ein Gazetteer für den Haupttyp `location`, den Untertyp `opengeo` in der Sprache `german` zu finden ist[19]. Der Haupttyp `location` sorgt dafür, dass direkt in der bereits existierenden Grammatikdatei `location.jape` im Verzeichnis `<GATE_DIR>/plugins/Lang_German/resources/grammar/` eine neue Regel für OpenGeoDB-Toponyme angefügt werden kann, damit beim nächsten Aufruf des Plugins die übereinstimmenden Token auch mit einer `Location`-Annotation versehen werden. Die entsprechende Regel wird in Abbildung 3.10 dargestellt.

[19] `http://gate.ac.uk/sale/tao/splitch13.html#sec:gazetteers:anniegaz` - abgerufen am 18. März 2014

Analysiert nun die bestehende Anwendungspipeline einen Text, welcher zuvor nicht erkannte Toponyme enthält, diese aber in der OpenGeoDB enthalten sind, werden diese von GATE korrekt annotiert. Abbildung 3.11 zeigt das Ergebnis an einem entsprechenden Satz.

3.2.4 Toponymerkennung mit Geonames.org

Einführung

Trotz der großen Erweiterung der von GATE mitgelieferten Gazetteers existieren immer noch Erkennungslücken speziell bei Orten außerhalb des deutschen Sprachgebiets. Um auch diese bei Bedarf zuverlässig erkennen zu können, ist noch ein weiterer Anschluss an eine Datenbank erforderlich, die eine gute Abdeckung für Orte weltweit hat. *Geonames.org* besitzt in seiner Datenbank nach eigenen Angaben Informationen zu Millionen von geographischen Namen, welche über verschiedene Quellen, etwa einer Web Service-API oder ähnlich wie bei OpenGeoDB über einen Datenbankabzug abrufbar sind[20]. Aufgrund der Größe der Datenbank wurde für die Integration in GATE eine Anbindung mittels der Web Service-API gewählt. Eine solche Integration ist nun im Gegensatz zur Gazetteer-Erweiterung mit den OpenGeoDB-Daten nicht mehr über wenige Modifikationen und Erweiterungen an der bestehenden Infrastruktur möglich, da programmatisch auf die entsprechenden GeoNames-Schnittstellen zugegriffen werden muss. Aus diesem Grund wird hierfür eine eigene GATE Processing Resource (PR) entwickelt, welche `Location`-Annotationen auf Basis der Informationen von geonames.org erstellen soll.

Die Entwicklung dieser PR erfolgt in dem neuen Plugin *YGRIEGA*, welches auch weitere Neuentwicklungen im Zuge dieser Masterarbeit beinhaltet, die in Kapitel 4 vorgestellt werden. YGRIEGA steht hierbei für „**Y**GRIEGA **G**eoreferenced **R**hetorical figures **I**dentifier, **E**xtractor and **G**rammar **A**nalyzer“. Die Funktionalität der neuen Processing Resource beschränkt sich im Wesentlichen auf die Möglichkeiten, welche die Such-API von GeoNames bietet. Zur Erkennung von Toponymen eignet sich hier die Volltextsuche. Diese ist auch über eine Java-Anwendungsbibliothek zugreifbar und wird folglich hier verwendet[21][22]. Da diese Volltextsuche nicht nur Informationen darüber zurückliefert, ob für eine Anfrage ein Toponym gefunden werden konnte, sondern deren Antwort auch eventuell verfügbare geographischen Informationen wie Ortskoordinaten, das Land oder die Region beinhaltet, kann hier durchaus von einer Toponymauflösung gesprochen werden. Daher wird die neue GATE PR *YGRIEGA Geonames.org Toponym Resolver* genannt.

Voraussetzungen

Der Toponym Resolver soll hierbei pro Token herausfinden, ob es sich um ein potentielles Toponym handelt und wenn ja, dieses als Anfrage an den GeoNames-Web Service weiterleiten. Da für die Charakterisierung der Toponyme grundsätzlich keine geographische Auflösung nötig ist, wird für bereits erkannte Toponyme auf eine Anfrage an GeoNames verzichtet.

[20] `http://www.geonames.org/about.html` - abgerufen am 19. März 2014
[21] `http://www.geonames.org/export/geonames-search.html` - abgerufen am 19. März 2014
[22] `http://www.geonames.org/source-code/` - abgerufen am 19. März 2014

```
input  : Annotations by Durm German Lemmatization System (type Lemma) or by
         TreeTagger (type Token)
output: searchQuery of type String
1 if Annotations CONTAINS Lemma then
2 |  searchQuery ← Annotations.Lemma.String;
3 else
4 |  if Annotations.Token CONTAINS Lemma then
5 |  |  searchQuery ← Annotations.Token.Lemma;
6 |  else
7 |  |  searchQuery ← Annotations.Token.String;
8 |  end
9 end
```

Abbildung 3.12 — Pseudocode der Ermittlung der GeoNames-Suchanfrage

Um das Auftreten von falschen Fundstellen aufgrund von Geo/Non-Geo-Ambiguitäten so weit wie möglich zu vermeiden, wird auch für den Anschluss an GeoNames das bereits bei der OpenGeoDB-Integration verwendete Wörterbuch mit den 10.000 am häufigsten genutzten Wörtern der deutschen Sprache verwendet. So sollen für die Suche nur diese Token verwendet werden, welche nicht in dieser Liste enthalten sind. Darüber hinaus werden ausschließlich Nomen und Adjektive als potentielle Toponyme angesehen. Andere Wortarten werden folglich bei GeoNames-Suchanfragen nicht berücksichtigt. Ist ein Token als potentielles Toponym identifiziert, wird dessen lemmatisierte Fassung als Suchanfrage für GeoNames verwendet. Tests haben gezeigt, dass das Durm German Lemmatization System in der Regel bessere Ergebnisse liefert als die anderen auf TreeTagger basierenden Lösungen. Falls also dieses System ein Lemma erkannt hat, existiert eine Annotation vom Typ `Lemma`, was dann auch für die Suchanfrage verwendet wird. In anderen Fällen wird auf die TreeTagger-Annotation vom Typ `Token` zurückgegriffen. In den meisten Fällen existiert in dieser Annotation dann ein Feature `Lemma`, was dann als Suchanfrage benutzt werden kann. Falls aber auch der TreeTagger kein Lemma ermitteln konnte, wird auf das ursprüngliche Wort aus dem Text zurückgegriffen. Abbildung 3.12 zeigt den Pseudocode der Ermittlung dieser Suchanfrage.

Diese Voraussetzungen machen es nötig, dass die neue Processing Resource in der Anwendungspipeline möglichst nach erfolgter Wortarterkennung und der Named Entity Recognition ausgeführt wird. Nur so kann die Komponente auf das Lemma eines Tokens und seine Charakteristika zurückgreifen sowie erkennen, ob es bereits mit einer `Location`-Annotation versehen ist.

Implementierung

Die Implementierung besteht auch hier aus zwei Klassen, der `ToponymResolverPR` und dem `GeoNamesConnector`. Die erste erweitert hierbei die abstrakte Klasse `AbstractLanguageAnalyzer` des CREOLE-Plugin-Frameworks und ist somit die Einstiegsklasse dieser Processing Resource. Sie liest die Stoppwortliste der häufigsten deutschen Wörter ein, bestimmt auf Basis der existie-

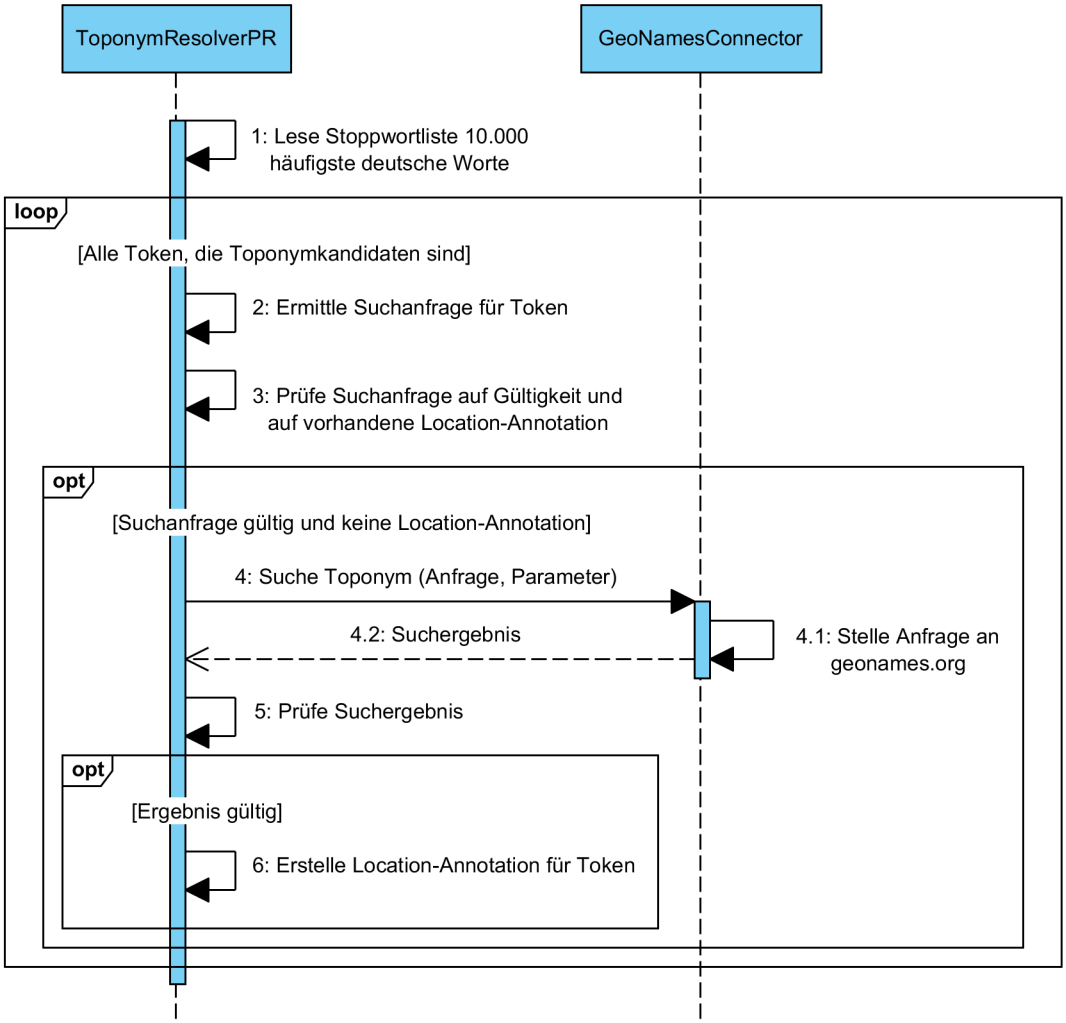

Abbildung 3.13 — Sequenzdiagramm der Processing Resource Geonames.org Toponym Resolver

renden Annotationen, für welches Token eine Suche mit welcher Suchanfrage gestartet wird und versieht bei erfolgreicher Suche das entsprechende Token mit einer `Location`-Annotation und eventuellen Zusatzinformationen über Land oder Bevölkerung in passenden Features.

Als Einstiegsklasse verwaltet es auch die Laufzeitparameter der PR, welche z.B. über GATE Developer verändert werden können. Da zur Nutzung des GeoNames-Web Service ein gültiger Benutzername nötig ist, definiert die Klasse den verpflichtenden Eingabeparameter `geoNamesUserName`. Weiterhin kann der Nutzer vor der Ausführung in einer GATE Anwendungspipeline mit den Parametern `onlyPopulatedPlaces` und `populationThreshold` entscheiden, ob sich die Suche auf bevölkerte Orte beschränken soll und wenn ja, ob der Ort eine Mindestanzahl an Einwohnern besitzen muss, damit er als gültiges Ergebnis akzeptiert wird. Weiterhin kann mit der Option `resultNameMustMatchQuery` bestimmt werden, ob als gültiges Ergebnisse nur solche Orte akzeptiert werden sollen, deren Name exakt der Verwendung im Text entspricht. Wird diese ausgeschaltet, können beispielsweise auch Token wie *Konstantinopel*, das heutige

```
<CREOLE-DIRECTORY>
  <JAR SCAN="true">toponym-resolver.jar</JAR>
  <JAR SCAN="true">geonames-1.1.11.jar</JAR>
  <JAR SCAN="true">jdom-1.0.jar</JAR>
</CREOLE-DIRECTORY>
```

Abbildung 3.14 — Inhalt der `creole.xml`-Datei für die Geonames.org Toponym Resolver PR

Istanbul, korrekt als Toponym identifiert werden. Alle vorgestellten Restriktionen dienen dazu, mögliche falsche Treffer aufgrund von Ambiguitäten zu vermeiden und können so vom Nutzer je nach Text oder Korpus individuell eingestellt werden. Darüber hinaus kann auch bei dieser Processing Resource wie bei CREOLE-Plugins üblich das Ein- und Ausgabe-Annotation Set (`inputAnnotationSetName`, `outputAnnotationSetName`) verändert werden.

Die Klasse `GeoNamesConnector` übernimmt die eigentliche Anbindung an den Web Service, führt also die Suche auf Basis der von `ToponymResolverPR` übergebenen Parameter durch. Wurden Ergebnisse vom Web Service gefunden, prüft die Klasse diese und gibt sie an die Einstiegsklasse zurück, wenn sie gültig sind. Das Sequenzdiagramm 3.13 zeigt das Zusammenspiel beider Klassen.

Integration in GATE

Zur Integration der neuen Processing Resource in GATE müssen beide Klassen kompiliert und in ein Java-Archiv verpackt werden. Dieses Archiv wird zusammen mit den benötigten Archiven der GeoNames-API und der JDOM-Bibliothek sowie der Datei `top10000de.txt` in das Verzeichnis `<GATE_DIR>/plugins/YGRIEGA/` kopiert. Damit das neue Plugin auch von GATE erkannt wird, muss im gleichen Verzeichnis eine Datei mit dem Namen `creole.xml` angelegt werden, in der die Archive korrekt referenziert werden. Abbildung 3.14 zeigt den Inhalt dieser Datei.

Nach einem Neustart von GATE kann das YGRIEGA-Plugin mit dem CREOLE Plugin-Manager geladen werden, sodass die Geonames.org Toponym Resolver PR zur Verfügung steht. Die geladene Ressource kann nun in die bestehende Anwendungspipeline eingefügt werden. Hierbei ist darauf zu achten, dass sie nach allen Ressourcen zur Wortarterkennung und Named Entity Recognition ausgeführt wird. Im Anschluss müssen noch der eigene geonames.org-Benutzername als Laufzeitparameter gesetzt und die anderen Parameter nach Bedarf angepasst werden. Abbildung 3.15 zeigt, dass die PR hier deshalb nach dem Durm German Lemmatization System und vor dem finalen Entfernen nicht benötigter Annotationen ausgeführt wird.

Wird die Anwendungspipeline nun für einen Text ausgeführt, der zuvor nicht erkannte Toponyme enthält, die aber über die GeoNames-Datenbank identifiziert werden können, werden diese von GATE mit der entsprechenden `Location`-Annotation versehen. Weitere Charakteristika wie das zugehörige Land und die Bevölkerungszahl werden als Features hinzugefügt. Abbildung 3.16 zeigt einen solchen Fall.

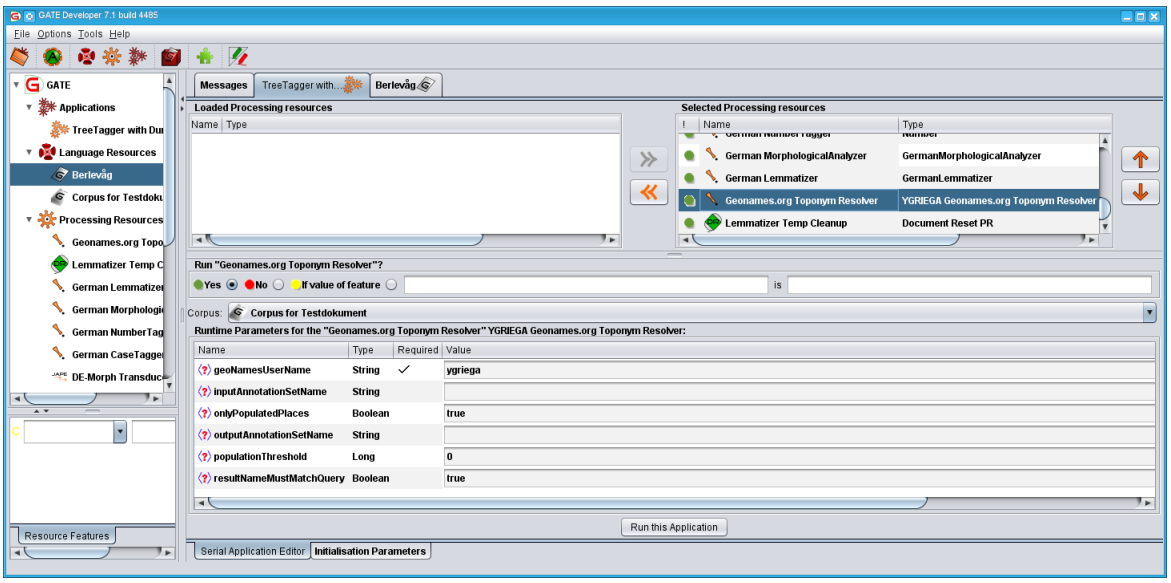

Abbildung 3.15 — Integration der Geonames.org Toponym Resolver PR in GATE

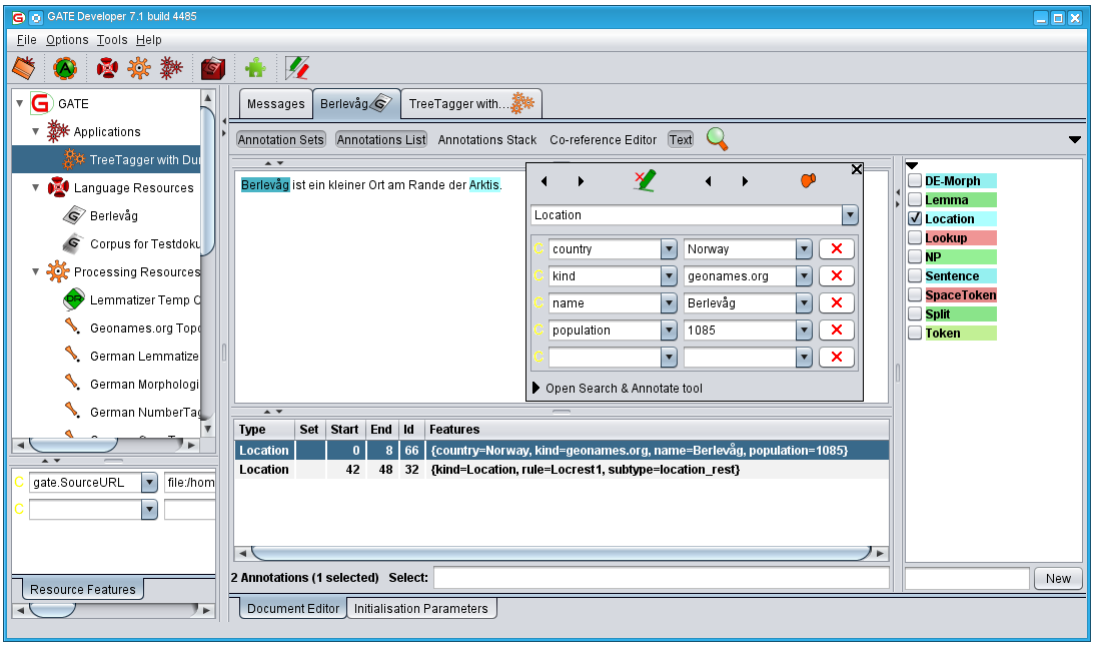

Abbildung 3.16 — `Location`-Annotation für einen mittels geonames.org erkannten Ort

4 Charakterisierung von Toponymverwendungen

4.1 Einführung - Integration in GATE

Mit den verschiedenen Erweiterungen für eine bessere Toponymerkennung kann nun darauf aufbauend eine GATE-Erweiterung für die eigentliche Charakterisierung implementiert werden. Diese soll die in Kapitel 2.1 aufgeführten und im Kontext von Toponymverwendungen relevanten Stilmittel erkennen und wie die anderen Erweiterungen die Fundstellen entsprechend annotieren. Die neue Komponente wird wie der Toponym Resolver aus Kapitel 3.2.4 Bestandteil des im Zuge dieser Masterarbeit entstandenen Plugins YGRIEGA. Allerdings wird sie als eigene Processing Resource mit dem Namen *YGRIEGA Toponym Analyzer* eingebunden.

Die Einstiegsklasse dieser Ressource namens `ToponymAnalyzerPR` erweitert analog die Klasse `AbstractLanguageAnalyzer` des GATE CREOLE Plugin-Frameworks. Im Gegensatz zur GeoNames-Integration sind hier allerdings keine expliziten Konfigurationsparameter vorgesehen, die Einfluss auf die Analyse nehmen. Ausschließlich die für GATE-PRs üblichen Möglichkeiten zur Änderung der Ein- und Ausgabe-Annotation Sets, sowie ein optionaler Parameter zur Integration des lexikalisch-semantischen Netzes GermaNet, werden unterstützt. Der zugehörige Parameter `germaNetXMLPath` erwartet einen korrekten Pfad zu den im XML-Format abgespeicherten Dateien der GermaNet-Datenbank, die von der Universität Tübingen bezogen werden können[1]. Für diese Masterarbeit wird die zum Zeitpunkt der Ausarbeitung aktuelle Version 8.0 vom April 2013 verwendet. Diese Datenbank ist nur für Teile der im Folgenden beschriebenen Analysen notwendig. Sollte sie also nicht verfügbar sein, kann dieser Parameter leer bleiben und der Toponym Analyzer wird dann die von GermaNet abhängigen Funktionen abschalten.

Da diese Processing Resource auf den bereits beschriebenen vorherigen Komponenten aufbaut, wertet sie die verschiedenen Typen von Annotationen entsprechend aus. Für eine erfolgreiche Analyse müssen deshalb zumindest Elemente der Kategorie `Token`, `Sentence`, `DE-Morph` und `Location` für den zu analysierenden Text vorhanden sein. Optional sind Annotationen vom Typ `Lemma` sowie Features von `Token`, welche von ParZu hinzugefügt werden. Die Funktionalität der Analysekomponente ist allerdings ohne die ParZu-Informationen stark eingeschränkt, weswegen deren Integration stark empfohlen wird.

Die Integration der kompletten Processing Resource, deren Aufbau und Funktionsweise in den Abschnitten 4.2 und 4.3 im Detail vorgestellt werden, erfolgt ebenfalls wie beim Geonames.org Toponym Resolver. Alle Bestandteile müssen als Java-Archiv im Verzeichnis `<GATE_DIR>/plugins/YGRIEGA/` vorliegen und mit entsprechenden Einträgen in der `creole.xml` dem GATE

[1] `http://www.sfs.uni-tuebingen.de/lsd/index.shtml` - abgerufen am 20. März 2014

```
<CREOLE-DIRECTORY>
  <JAR SCAN="true">toponym-resolver.jar</JAR>
  <JAR SCAN="true">toponym-analyzer.jar</JAR>
  <JAR SCAN="true">geonames-1.1.11.jar</JAR>
  <JAR SCAN="true">jdom-1.0.jar</JAR>
  <JAR SCAN="true">GermaNetApi8.0.jar</JAR>
</CREOLE-DIRECTORY>
```

Abbildung 4.1 — Inhalt der `creole.xml`-Datei für beide YGRIEGA Processing Resources

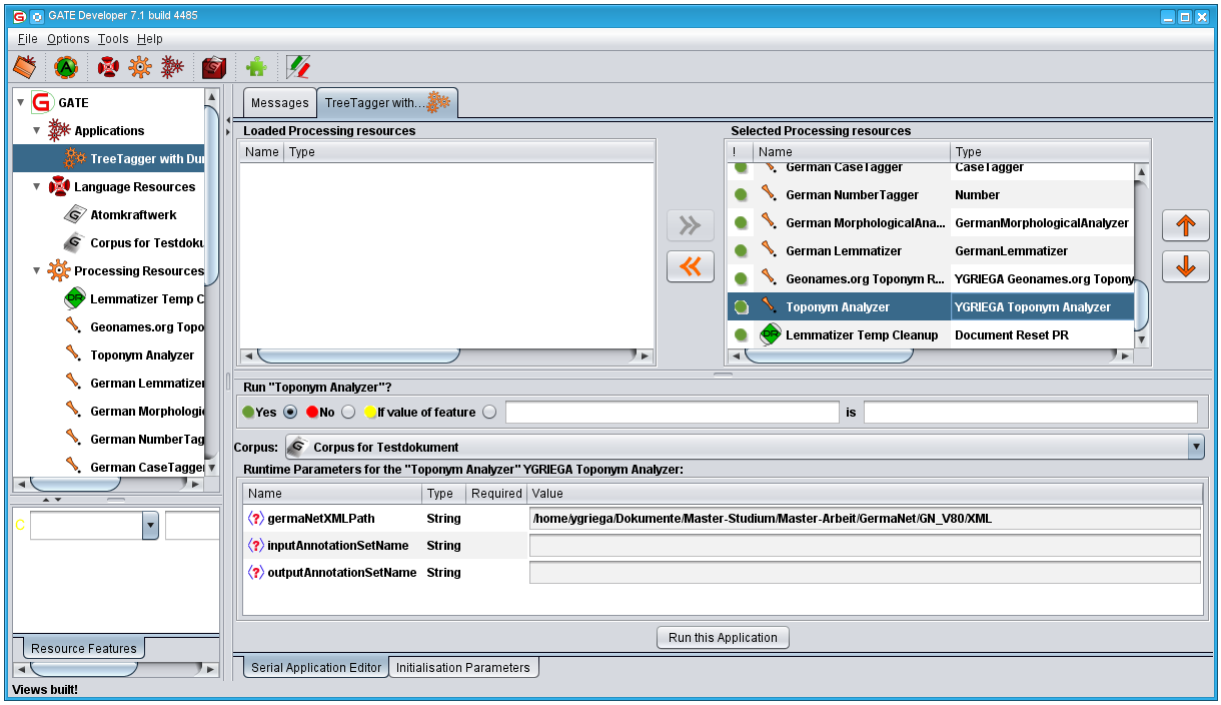

Abbildung 4.2 — Der Toponym Analyzer in einer GATE Anwendungspipeline

Plugin-Framework bekannt gemacht werden. Es kommen also zwei weitere Dateien, eine für die eigentliche Ressource und eine für die API-Komponenten von GermaNet[2], hinzu. Die erweiterte Datei wird in Abbildung 4.1 dargestellt.

Nach einem Neustart von GATE steht dann auch diese Ressource zur Verfügung und sollte folglich nach den relevanten anderen Elementen in die Anwendungspipeline eingefügt werden. Im Beispiel in Abbildung 4.2 wird der Toponym Analyzer direkt nach der GeoNames-Komponente und vor dem schlussendlichen Aufräumen nicht benötigter Annotationen aufgerufen.

[2] `http://www.sfs.uni-tuebingen.de/lsd/tools.shtml` - abgerufen am 20. März 2014

4.2 Architektur

4.2.1 Grundlagen

Im Gegensatz zu den beiden zuvor vorgestellten Neuentwicklungen im Zuge dieser Masterarbeit ist zur Charakterisierung von Toponymverwendungen eine deutlich umfangreichere Logik notwendig. Zum einen ist die Problemstellung mit Hinblick auf die grammatikalische und semantische Analyse deutlich komplexer als bei den Beispielen zuvor. Darüber hinaus muss die Logik zur Verarbeitung der bestehenden Informationen an die Heterogenität und optionale Verfügbarkeit gewisser Teile der Textanalyse angepasst sein, um eine effiziente Weiterverarbeitung sicherstellen zu können.

Jegliche zuvor ausgeführte Komponente trägt ihren Teil zur Problemlösung - der korrekten Annotation verschiedener Verwendungsarten und Stilmittel - bei, indem sie selbst Token oder entsprechende Teile des Textes annotiert bzw. existierende Annotationen erweitert. Ihre Ausführung wird dabei von einer zentralen Komponente - hier GATE Developer und der Konfiguration der Anwendungspipeline - gesteuert, damit diese Informationen wegen eventueller Abhängigkeiten in der richtigen Reihenfolge ausgewertet und aktualisiert werden können.

Überträgt man diese Herangehensweise auf Problemstellungen in der realen Welt, ergeben sich Parallelen zu dortigen Problemlösungsstrategien. Aus einschlägigen Kriminalromanen und -filmen ist beispielsweise bekannt, dass die Ermittler bereits existierende Informationen zu Opfern, Zeugen und Verdächtigen an einer Pinnwand befestigen und dort miteinander in Beziehung setzen. Nach und nach ergibt sich ein Bild des kompletten Tathergangs und ggf. bisher verborgener Informationen, welche von den Polizisten zur Lösung des Falls verwendet werden können.

In der Softwareentwicklung wurde bereits in den 70er Jahren des 20. Jahrhunderts ein dazu passendes Architekturmuster beschrieben und in entsprechenden Systemen umgesetzt. Das sogenannte *Blackboard-Modell* fand so in Systemen zur Spracherkennung und Ozeanüberwachung seine ersten Einsatzmöglichkeiten [Nii86]. Dieses Modell besteht laut Nii in der Regel aus drei Komponenten, den *Wissensquellen* („Knowledge Sources“), einer *Tafel als globaler Datenbank* („Blackboard Data Structure“) sowie einer *Kontrolleinheit*. Die *Wissensquellen* repräsentieren so auch in der programmatischen Umsetzung die verschiedenen, eventuell ganz unterschiedlich agierenden Komponenten, die zur Problemlösung gebraucht werden. Diese werden voneinander unabhängig modelliert und implementiert. Allerdings müssen sie ihr Wissen und damit ihren Beitrag zur Problemlösung in einer zentralen Datenbank, dem *Blackboard*, ablegen, damit es von anderen Komponenten weiterverwendet werden kann. So trägt die Ausführung jeder neuen Wissenquelle nach und nach dazu bei, der Lösung des Problems einen Schritt näherzukommen. Dabei kommunizieren die einzelnen Komponenten nicht untereinander, sondern arbeiten ausschließlich über das bereits auf dem Blackboard gesammelte Wissen indirekt miteinander zusammen. Die zentrale Kontrollinstanz wacht in diesem Kontext über die Instanziierung und den Zustand des Blackboards, sowie über die korrekte Ausführungsreihenfolge der einzelnen Wissensquellen [Nii86].

Dieses Konzept wird nun bei der Realisierung des Toponym Analyzers entsprechend berücksichtigt. Die Einstiegsklasse `ToponymAnalyzerPR` übernimmt hierbei die Rolle der Kontrollinstanz, baut also das Blackboard auf und ruft die verschiedenen Wissensquellen in der richtigen Rei-

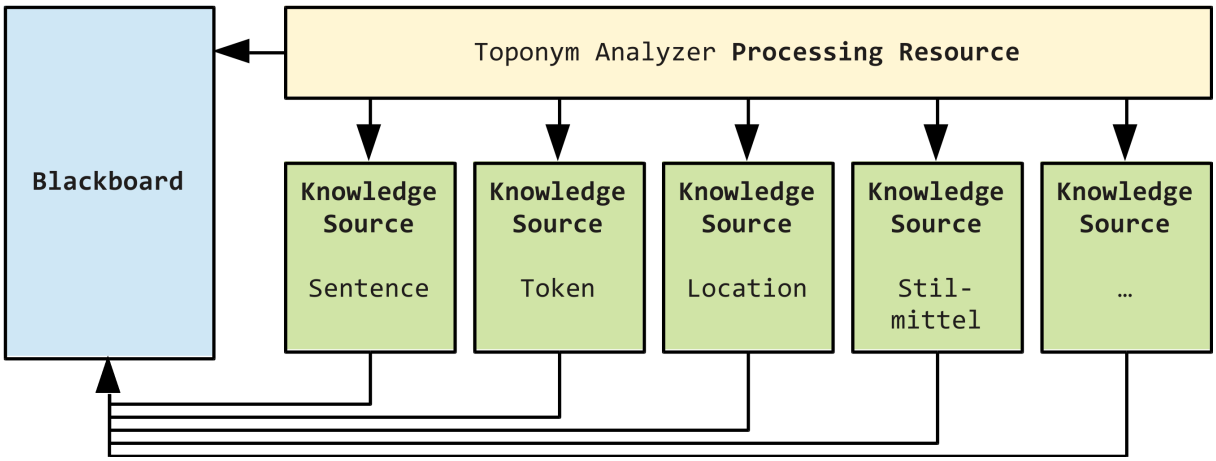

Abbildung 4.3 — Blackboard-Architektur des Toponym Analyzers

henfolge auf, sodass diese die Ergebnisse der einzelnen Analysen in der zentralen Datenbank ablegen können. Die Wissensquellen werden zunächst anhand der einzelnen Annotationstypen modelliert. So kümmert sich beispielsweise eine Knowledge Source um den Typ `Sentence`, also die Sätze, während andere die einzelnen Token, morphologischen Informationen oder Elemente mit Ortsbezug (`Location`) aus dem GATE-Framework auslesen und auf dem Blackboard verfügbar machen. Sind die bestehenden Informationen ausgelesen, können weitere Wissensquellen diese Informationen vom Blackboard nutzen, um die Verwendungsarten von Toponymen oder die Stilmittel in deren Kontext zu erkennen und die Token oder andere Textteile mit entsprechenden Annotationen zu versehen. Abbildung 4.3 visualisiert diese Blackboard-Architektur, wie sie im Toponym Analyzer zum Einsatz kommt.

4.2.2 Implementierung

Basiselemente

GATE bietet über die mitgelieferte Klassenbibliothek bereits viele Möglichkeiten auf die Annotationen, ihre Features und weitere Informationen zuzugreifen, die von den verschiedenen Komponenten in der Anwendungspipeline ermittelt wurden. Allerdings beinhalten die entsprechenden Klassen, die von GATE Embedded zur Verfügung gestellt werden, ausschließlich generische Zugriffsmethoden, die nicht auf den konkreten Anwendungsfall dieser Arbeit zugeschnitten sind. So besteht zum Beispiel semantisch eine Teile-Ganzes-Beziehung zwischen den Annotationen vom Typ `Sentence` und anderen wie zum Beispiel `Token` oder `Location`. Für eine effiziente Analyse sollte die Information, welche Teile welchen Typs in einem Satz enthalten sind, schnell und einfach zugreifbar sein, was ohne weitere Modellierung mit der von GATE zur Verfügung gestellten Infrastruktur nur umständlich über mehrere Methodenaufrufe möglich ist. Ähnlich verhält es sich mit anderen für die Analyse wichtigen Informationen, etwa über die Position einer Annotation in einem Satz oder aller Annotationen nach einem bekannten Token.

Aus diesen Gründen implementieren die Basiselemente des Toponym Analyzers das Entwurfsmuster *Fassade* für die verschiedenen Typen von Annotationen. So soll sichergestellt werden, dass die Verwendung der technischen Schnittstellen der GATE Embedded-Infrastruktur stark vereinfacht wird, indem eine einheitliche Schnittstelle hierfür bereitgestellt wird [GHJV12]. Die verschiedenen Methodenaufrufe der GATE-Schnittstellen werden so bildlich gesprochen hinter einer einheitlichen Fassade versteckt, sodass die weitere Analysefunktionalität in der Regel nur noch die einfachen Methoden der Toponym Analyzer-Basiselemente verwenden muss.

Hierfür wird für jeden Annotationstyp eine eigene `AnnotationFacade` angelegt, welche jeweils speziell zugeschnittene Zugriffsmethoden anbietet. Allerdings gibt es auch Funktionalität, welche für jede Kategorie zur Verfügung gestellt werden muss, weswegen diese in einheitlichen Schnittstellen und Oberklassen definiert bzw. implementiert wird. Die Wurzelschnittstelle ist deshalb `IAnnotationFacade`, welche die beiden Methoden `getAnnotation()` und `getAnnotationType()` definiert, die einen Zugriff auf das GATE-eigene Interface `Annotation` zulassen bzw. den mit einer passenden Enumeration kategorisierten Typ der verschalten Annotation zurückliefern. Als Unterklassen sind für die spätere Analyse die `Sentence`-Annotationen von allen anderen zu unterscheiden, da alle anderen erkannten Annotationen Teil eines Satzes sind, ein Satz selbst aber nicht von einem weiteren umgeben werden kann. Die Charakeristika eines Satzes werden folglich von der Schnittstelle `ISentence` und die gemeinsamen Eigenschaften aller anderen Annotationen von `ISentencePart` definiert. Die spezifische Funktionalität der einzelnen Satzteile wird dann von Schnittstellen modelliert, welche `ISentencePart` erweitern.

Konkret implementiert werden alle Fassaden in entsprechenden Java-Klassen, welche sich auf die zugehörigen Schnittstellen beziehen und zusätzlich die abstrakte Klasse `AbstractAnnotationFacade` erweitern, welche generische Sortierfunktionalität für alle Fassaden zur Verfügung stellt. Abbildung 4.4 zeigt ein Klassendiagramm der im Toponym Analyzer verfügbaren Basiselemente. Zur besseren Übersichtlichkeit wurde auf die Darstellung der Methoden der annotationsspezifischen Fassaden verzichtet, da deren Anzahl je nach Typ recht umfangreich sein kann. In der Regel existiert für jedes mögliche Feature einer Annotation eine dedizierte `setFeature()`- bzw. `getFeature()`-Methode, also etwa für das `Token` die `getCategory()`- und `setCategory()`-Methoden, welche das entsprechende STTS-Tag eines Tokens auslesen oder ändern. Für alle entsprechenden Methoden gilt, dass eine Änderung nicht nur auf der Fassade, sondern auch über die entsprechenden GATE-Methoden direkt in den Features einer Annotation durchgeführt wird. Weiterhin existieren bei einzelnen Fassaden Methoden, die den Zugriff auf andere Fassaden erleichtern, die für das gleiche Token erstellt wurden. So kann etwa vom `Toponym` mit der `getLocation()`-Methode auf die `Location`-Fassade oder von der `Location` mit der `getToken()`-Methode auf die `Token`-Fassade für das gleiche Wort zugegriffen werden. Auch bei diesen Abhängigkeiten wurde auf eine Darstellung im Klassendiagramm zugunsten der Übersichtlichkeit verzichtet.

Wie in der Grafik ersichtlich ist, wird mit dem Toponym Analyzer neben den bereits bekannten Typen von Annotationen zusätzlich die Annotation `Toponym` eingeführt. In den Features dieses neuen Typs werden ausschließlich die neu erkannten Verwendungsarten und Stilmittel hinterlegt, die im Kontext der entsprechenden Ortsbezüge verwendet wurden. Diese Trennung dient allein der besseren Visualisierung der Unterschiede zwischen der Erkennung und Auflösung von Toponymen sowie deren Charakterisierung im Rahmen dieser Masterarbeit, da die `Location`- und `Toponym`-Annotationen sonst das gleiche Token bzw. den gleichen Satzteil beschreiben. In

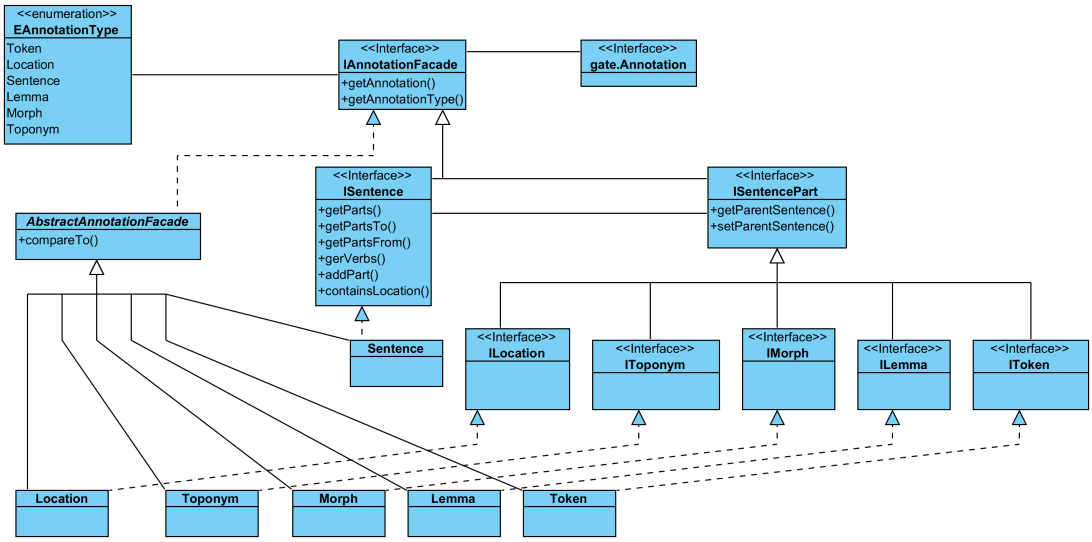

Abbildung 4.4 — Klassendiagramm der Toponym Analyzer-Basiselemente

einer produktiven Fassung des Toponym Analyzers sind die neuen Charakterisierungsfeatures dann auch dem gängigen Typ `Location` zuzuordnen. Auf einen eigenen Annotationstyp kann so verzichtet werden.

Blackboard und Knowledge Sources

Mit den nun verfügbaren Basiselementen kann die in Abschnitt 4.2.1 vorgestellte Blackboard-Architektur entsprechend implementiert werden. Als `Blackboard` fungiert eine gleichnamige Java-Klasse, welche die Instanzen der verschiedenen Annotation Facades sortiert nach ihrer ID[3] abspeichert. Dieses Blackboard wird von der Einstiegsklasse `ToponymAnalyzerPR` instanziiert und verwaltet. `Blackboard` stellt darüber hinaus einige Servicemethoden zur Verfügung. So kann beispielsweise zentral auf der Tafel abgelesen werden, ob die Anwendungspipeline ParZu beinhaltet und somit Informationen über Abhängigkeiten zwischen Tokens verfügbar sind. Weiterhin können für eine übergebene Instanz einer Fassade eventuell weitere Fassaden ermittelt werden, die sich im Text an der gleichen Position befinden, also das gleiche Token beschreiben. Die Listen mit den verfügbaren Fassaden werden nach ihrem Typ unterteilt als finale Attribute an der Klasse zur Verfügung gestellt.

Die Erstellung der konkreten Instanzen und das Ablegen auf dem Blackboard übernehmen die entsprechenden Knowledge Sources. In der Regel existiert für jede Annotation Facade eine entsprechende Wissensquelle. Einzige Ausnahme ist die neue `Toponym`-Annotation, für die zwei Knowledge Sources, eine für die Umwandlung der `Location`-Annotationen in `Toponym`-Annotationen und die Ermittlung der Verwendungsarten (`ToponymKnowledgeSource`) sowie eine für die Ermittlung der Stilmittel (`RhetoricalFiguresKnowledgeSource`) implementiert wurden. Jede Wissensquelle muss die Methode `contribute(Blackboard)` implementieren, welche

[3] `http://gate.ac.uk/sale/tao/splitch7.html#sec:api:annotations` - abgerufen am 22. März 2014

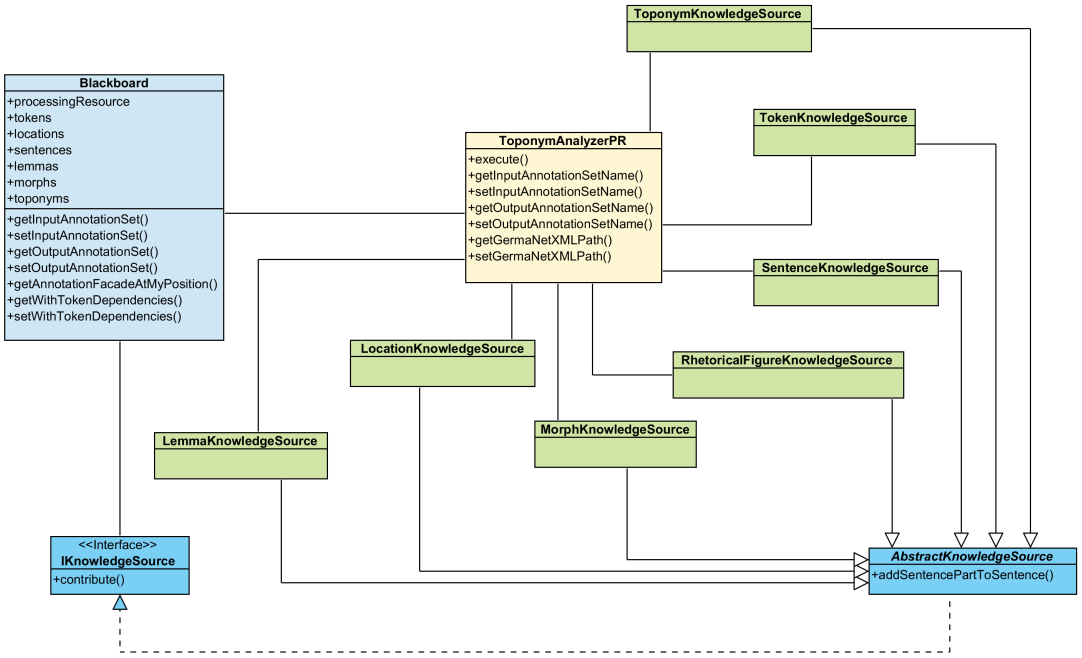

Abbildung 4.5 — Klassendiagramm der Blackboard-Architektur

über die zentrale Schnittstelle `IKnowledgeSource` vorgegeben wird. Sie wird von der Einstiegsklasse des Toponym Analyzers, welche als Kontrollinstanz agiert, aufgerufen. Da die Instanz des Blackboards mit allen bisher gesammelten Informationen übergeben wird, kann in der Implementierung der Knowledge Source auf alle dort abgelegten Informationen zugegriffen werden und die Liste mit den Fassaden-Instanzen der eigenen Annotation entsprechend der von GATE zur Verfügung gestellten Annotationen und den bereits existierenden Fassaden auf dem Blackboard aktualisiert werden. Wissensquellen, welche Informationen über Satzteile beschaffen, haben zudem die Möglichkeit, über die Methode `addSentencePartToSentence` der abstrakten Oberklasse `AbstractKnowledgeSource` neu instanziierte Fassaden ohne eigene Implementierung der entsprechenden Satz-Fassade als Teil hinzufügen zu können. Abbildung 4.5 zeigt ein Klassendiagramm der konkreten Implementierung der Blackboard-Architektur ohne die Basiselemente, welche zuvor bereits vorgestellt wurden. Auch auf die Darstellung der einzelnen Methoden der Wissensquellen wurde aus Übersichtsgründen verzichtet.

Da die einzelnen Knowledge Sources direkt von der Processing Resource aufgerufen werden, bestimmt diese auch die Ausführungsreihenfolge und muss hierbei speziell die Abhängigkeiten der einzelnen Fassaden untereinander berücksichtigen. Damit beispielsweise die Satzteile den entsprechenden Sätzen zugeordnet werden können, müssen die Annotation Facades für `Sentence` zuerst auf dem Blackboard hinterlegt werden. Weiterhin müssen etwa die Instanzen für den Typ `Token` vor denjenigen für den Typ `Location` angelegt werden, weil aus den Fassaden für die Ortsbezüge das jeweils zugeordnete Token referenziert wird. Da alle Wissensquellen zudem auf die GATE-Rohdaten der Annotationen zugreifen müssen, werden diese direkt von der Processing Resource auf das Blackboard geschrieben, bevor die erste Knowledge Source ausgeführt wird. Die Ergebnisse der vorhergehenden Komponenten aus der Applikationspipeline werden

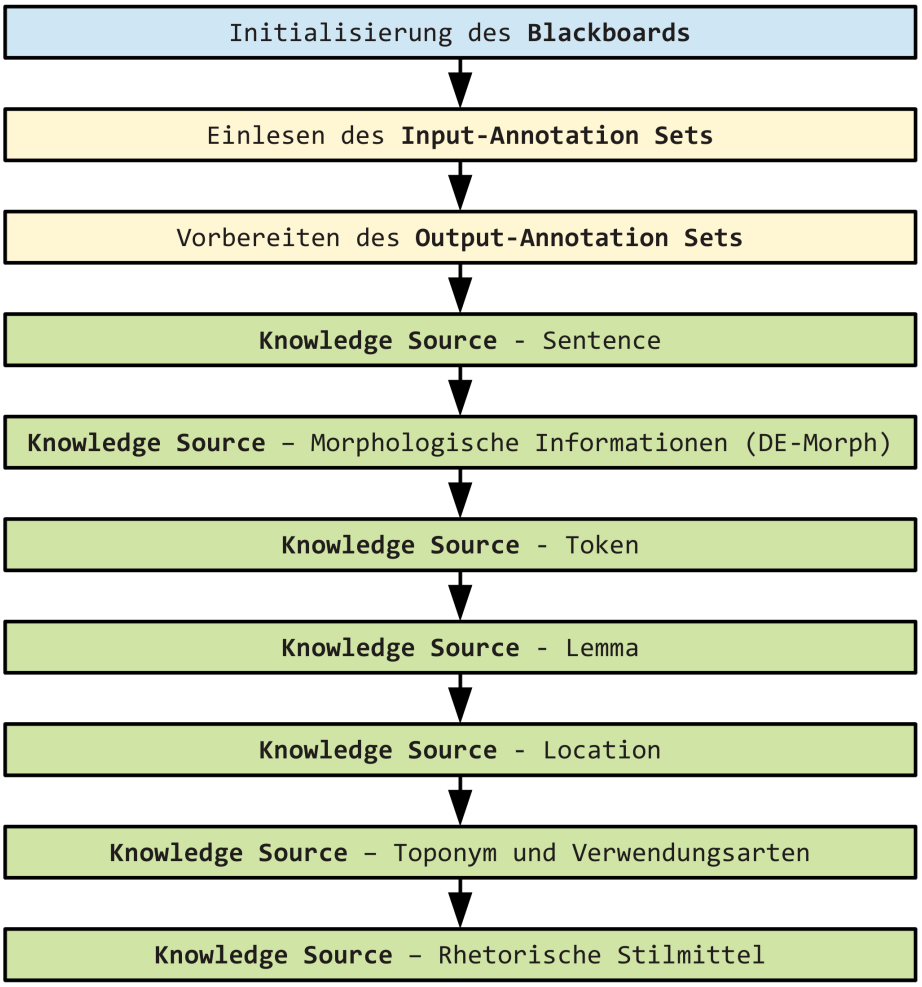

Abbildung 4.6 — Ausführungsreihenfolge der Wissensquellen im Toponym Analyzer

hierbei als `inputAnnotationSet` zur Verfügung gestellt. Alle Resultate des Toponym Analyzers inklusive der bereits vorhandenen Daten werden dann über das `outputAnnotationSet` weiteren nachfolgenden Komponenten zur Verfügung gestellt. Beide Annotation Sets können über die entsprechenden Laufzeitparameter an der Processing Resource in GATE Developer konfiguriert werden. Abbildung 4.6 zeigt einen Überblick über die Ausführungsreihenfolge der einzelnen Wissensquellen im Toponym Analyzer.

Auslesen der vorhandenen Informationen

Die einzelnen Wissensquellen lesen zu Beginn der Ausführung die Rohdaten zu den entsprechenden Annotationen aus dem Input-Annotation Set aus, welches zuvor auf dem Blackboard abgelegt wurde. Die so ausgelesenen Objekte vom Typ `gate.Annotation` können dann dem Konstruktor der passenden Annotation Facade übergeben werden, um ein gültiges Objekt der Fassade zu erzeugen. Wenn etwa wie beim `Sentence` keine weiteren Abhängigkeiten oder weitere

Logik zu beachten ist, können die neu angelegten Objekte zusammen mit ihrer ID in der passenden Datenstruktur auf dem Blackboard abgelegt werden. Selbst bei einigen weiteren Typen wie den morphologischen Informationen oder der `Lemma`-Annotation muss lediglich noch eine Verbindung zwischen dem neu angelegten Objekt und dem zugehörigen `Sentence`-Objekt hergestellt werden, da es sich hier um Satzteile handelt. Dies gilt natürlich auch für die übrigen Fassaden, allerdings wird hier noch zusätzliche Funktionalität benötigt. So kann die `Token`-Fassade eine bereits zuvor instantiierte Fassade mit morphologischen Informationen zum entsprechenden Token referenzieren, um die spätere Analyse zu erleichtern. Beinhaltet die Anwendungspipeline zudem Abhängigkeitsinformationen zwischen den einzelnen Token, die mittels ParZu ermittelt wurden, ordnet die `TokenKnowledgeSource` die einzelnen Fassaden in einer Baumstruktur analog zur Abbildung 2.3 an. An einer gegebenen `Token`-Fassade können dann im Anschluss die Elternfassade mit der Methode `getParent()` und die hiervon abhängigen Token mit der Methode `getChildren()` ermittelt werden.

Weiterhin unterstützt die Fassade für die `Location`-Annotation die GATE-Funktionalität, gleichnamige Token miteinander in Beziehung zu setzen. Wenn also in einem untersuchten Text beispielsweise mehrmals der Ort *Bamberg* vorkommt, erkennt GATE dies und referenziert in seinen Rohdaten die entsprechenden Annotationen. Diese Referenzen werden entsprechend in die zugehörigen Fassaden übernommen. Darüber hinaus wird direkt an der `Location`-Fassade der eigentliche Name des Orts abgelegt, sofern er durch die Anwendungspipeline noch nicht als Feature hinzugefügt wurde. Die entsprechende Information wird entweder direkt aus der `Token`- oder der `Lemma`-Fassade für das entsprechende Wort ermittelt.

Sind alle Fassaden für die bestehenden Informationen aufgebaut worden, folgt die zentrale Funktionalität des Toponym Analyzers, die Erstellung von `Toponym`-Annotationen für alle `Location`-Annotationen und das eigentliche Charakterisieren der gefundenen Toponyme. Hierzu wird zunächst von der `ToponymKnowledgeSource` für jeden vorhandenen Ortsbezug eine `Toponym`-Fassade angelegt, die auch gleich entsprechende GATE-Rohdaten am Output-Annotation Set erstellt, damit die Ergebnisse der Charakterisierung in GATE Developer oder in weiteren Komponenten der Anwendungspipeline sicht- und verwertbar werden. Im Anschluss ermittelt die `ToponymKnowledgeSource` zusätzlich die in Kapitel 2.1.1 definierten Verwendungsarten für die Toponyme. Den Abschluss der Analyse bildet die `RhetoricalFiguresKnowledgeSource`, welche die Ortsbezüge auf die in Kapitel 2.1.2 und 2.1.3 vorgestellten Figuren und Tropen untersucht. Beide Wissensquellen und die zur Erkennung der einzelnen Verwendungsarten und Stilmittel verwendeten Regeln und Algorithmen werden im Detail im folgenden Kapitel 4.3 vorgestellt.

4.3 Umsetzung

4.3.1 Verwendungsarten

Features der Toponym-Annotation

Analog zu den anderen Annotationen werden auch beim `Toponym` entsprechende Attribute als eigene *Features* hinterlegt. Die Verwendungsarten werden hierbei dem Feature `kind` zugeordnet und können in Anlehnung an die Definition in Kapitel 2.1.1 die Ausprägungen `standard`, `refe-`

```
   input  : tokenFacades in a Sentence before the toponym under investigation
   output: none
1  for tokenFacade ← tokenFacades[i ← tokenFacades.size] down to i = 0 do
2      i ← i - 1;
3      if tokenFacade.isAdposition() then
4          toponym.setKind(EToponymKind.reference);
5          toponym.setKindRule(EToponymKindRule.prePosition);
6          return;
7      end
8      if tokenFacade.isAdjective() or tokenFacade.isArticle() or
       tokenFacade.isToponymName() then
9          continue;
10     else
11         return;
12     end
13 end
```

Abbildung 4.7 — Pseudocode des Algorithmus der Verwendungsart *Referenz* mit Prä-/Zirkumposition

`rence` und `attribute` annehmen. Zusätzlich wird mit dem Feature `kindRule` die Regel bzw. der Algorithmus gespeichert, der bei der Identifikation angewendet wurde. Bei der Instanziierung werden alle Toponyme zunächst grundsätzlich mit der Verwendungsart *Standard* markiert. Nur wenn eine Regel für eine der beiden anderen Arten zutrifft, wird diese entsprechend verändert. So ist sichergestellt, dass jedes Toponym entsprechend zugeordnet wird.

Referenz

Jede Toponymverwendung, die zusammen mit einer Adposition vorkommt, ist als *Referenz* definiert (vgl. Kapitel 2.1.1). Um festzustellen, ob diese Verwendungsart z.B. mit einer Präposition vorliegt, wird ein Algorithmus angewendet, dessen Pseudocode Abbildung 4.7 zeigt. Hierzu müssen alle Tokens, die in einem Satz vor dem Toponym vorkommen, analysiert werden (Zeile 1). Diese Menge an Tokens kann in einer Schleife absteigend untersucht werden, indem das jeweils nächste Token auf seine Charakteristika geprüft wird. Steht direkt vor dem Toponym eine Adposition (Zeile 3), ist die *Referenz* festgestellt. Hat es einen anderen Typ, wird die Schleife nur fortgesetzt, wenn das Wort ein Adjektiv, ein Artikel oder ein anderes Toponym ist, da sich diese Kategorien von Worten noch auf das Toponym beziehen können(Zeile 8). Im Falle einer anderen Wortart kann es sich um keine *Referenz* mehr handeln und die Schleife bricht ab (Zeile 11). Zirkumpositionen können genau so erkannt werden, weswegen beide Unterkategorien unter der Regel `prePosition` zusammengefasst werden. Für Postpositionen konnte auch mit der Abhängigkeitsanalyse durch ParZu kein zuverlässiger Algorithmus implementiert werden, weshalb im Toponym Analyzer ausschließlich Prä- und Zirkumpositionen erkannt werden.

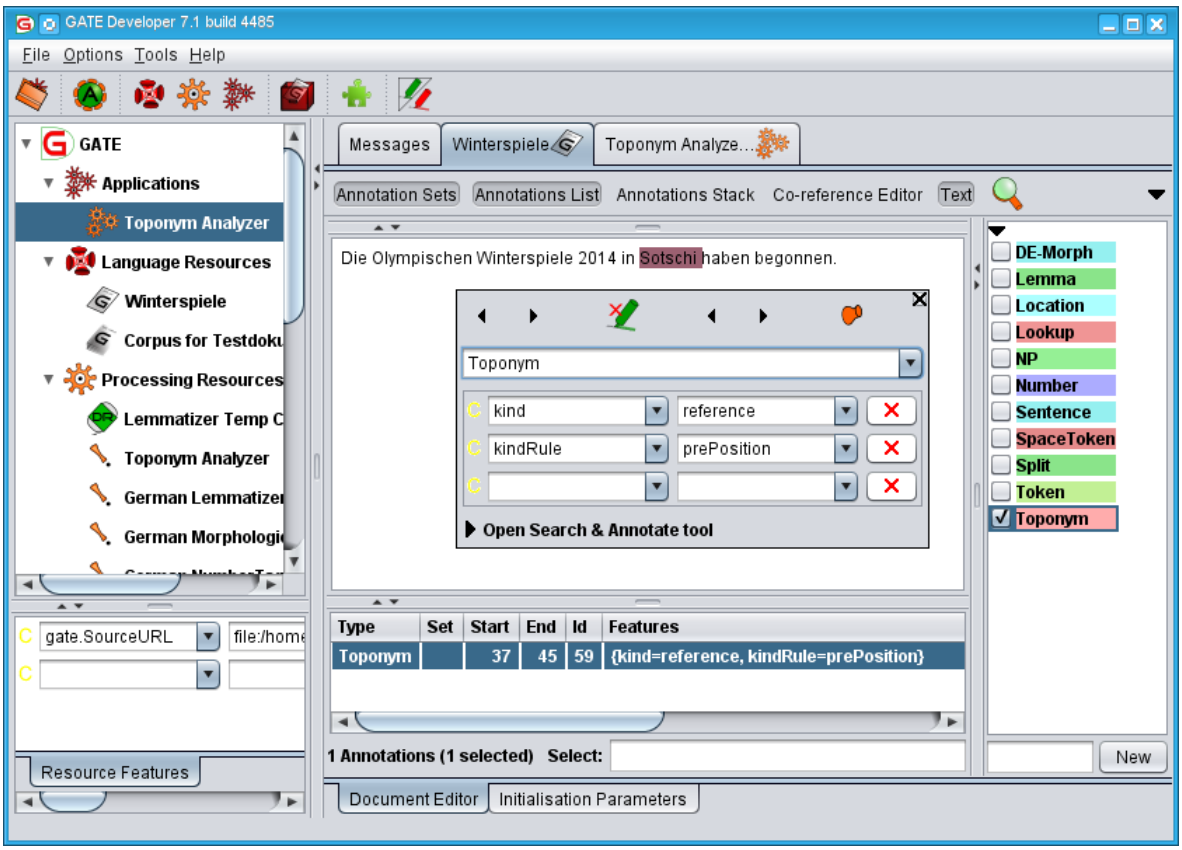

Abbildung 4.8 — Erfolgreiches Auffinden der Verwendungsart *Referenz* im Beispielsatz

Wird nun die komplette Anwendungspipeline mit dem Toponym Analyzer für den Beispielsatz ausgeführt, zeigt GATE Developer erwartungsgemäß eine `Toponym`-Annotation für das Wort *Sotschi* an. Diese Annotation wurde durch den beschriebenen Algorithmus mit den Features `kind=reference` sowie `kindRule=prePosition` versehen. Abbildung 4.8 zeigt das Ergebnis eines solchen Durchlaufs.

Attribut

Für die Verwendungsart *Attribut* gibt es verschiedene Möglichkeiten, wie sie im Text zusammen mit einem Toponym vorkommen kann. Diese werden entsprechend mit unterschiedlichen Algorithmen analysiert. Der Pseudocode in Abbildung 4.10 zeigt eine leicht vereinfachte Variante des Algorithmus zur Erkennung eines *Attributs* mit Zwischenworten. Als `kindRule` wird hier `intermediateAndCase` verwendet. Zunächst werden die Toponyme auf ihren Kasus und ggf. vorangegangene Präpositionen und Artikel hin untersucht. Das Vorgehen ist hier vergleichbar mit dem Algorithmus, der für das Auffinden von *Referenzen* zuständig ist. Für Toponyme, die entweder im Genitiv oder im Dativ stehen, werden alle vorhergehenden Token in einem Satz auf ihre Eigenschaften hin überprüft (Zeile 1). Werden vor dem Toponym nur gültige Zwischenwör-

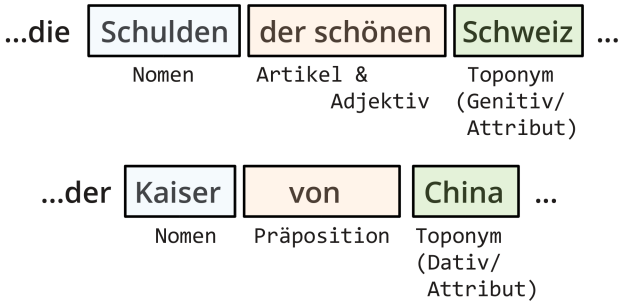

Abbildung 4.9 — Beispiele für die Verwendungsart *Attribut* mit Zwischenworten

```
   input  : tokenFacades in a Sentence before the toponym under investigation
   output: none
1  if toponym.getCase().equals(Case.Genitive) or toponym.getCase().equals(Case.Dative)
   then
2      for tokenFacade ← tokenFacades[i ← tokenFacades.size] down to i = 0 do
3          i ← i - 1;
4          if tokenFacade.isValidIntermediate() then
5              continue;
6          else
7              return;
8          end
9          if tokenFacade.isNoun() then
10             toponym.setKind(EToponymKind.attribute);
11             toponym.setKindRule(EToponymKindRule.intermediateAndCase);
12             return;
13         end
14     end
15 end
```

Abbildung 4.10 — Pseudocode des Algorithmus der Verwendungsart *Attribut* mit Zwischenworten

ter wie Adjektive (Zeile 4) gefunden und im Anschluss wiederum ein Nomen (Zeile 9), wird das Toponym als ein Attribut dieses Nomens betrachtet. Beispiele hierfür zeigt Abbildung 4.9.

Steht das Toponym im Genitiv und folgt direkt auf ein anderes Nomen oder wird selbst von einem anderen Nomen gefolgt, kommen andere Regeln zum Einsatz. Steht nach dem Toponym also ein weiteres Nomen und ggf. zugehörige Worte, trifft die Regel `genitiveBefore` zu. Ein Beispiel hierfür wäre „**Islands** kleine POLIZEI“. Die Regel `genitiveAfter` trifft analog auf Abschnitte wie „die SCHULDEN **Spaniens**“ zu.

Wie in Kapitel 2.1.1 beschrieben, werden Toponyme, die als Adjektive verwendet werden, ebenfalls als *Attribute* betrachtet. Adjektive werden bereits bei der Wortarterkennung durch den TreeTagger erkannt, folglich können alle erkannten Toponyme, die gleichzeitig Adjektive sind,

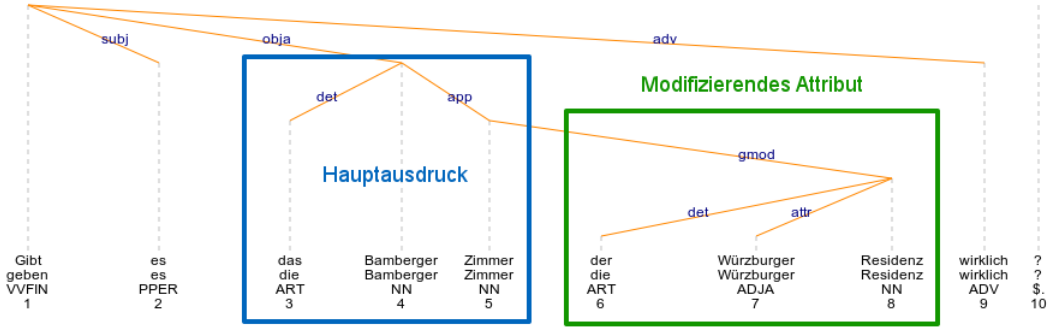

Abbildung 4.11 — Ausgabe von ParZu für ein Toponym mit modifizierendem Attribut

mit der Verwendungsart *Attribut* markiert werden. Die zugehörige Regel `adjective` sucht zudem das zugehörige Nomen aus dem entsprechenden Satz heraus, indem sie untersucht, ob nach dem Toponym ein Substantiv ggf. mit einem gültigen Beiwort wie einem weiteren Adjektiv folgt, oder nicht. Wenn das zutrifft, wird dem Toponym zusätzlich ein weiteres Feature namens `referencedToken` hinzugefügt, welches den Wortlaut des entsprechenden Nomens aus dem Text enthält. Ein Beispiel für die Regel `adjective` wäre „**Nordkoreanische** MEDIEN berichteten...“. Hier wird dann das Wort *Medien* als `referencedToken` betrachtet.

Zwei weitere Varianten des *Attributs* können nur erkannt werden, wenn die Anwendungspipeline die Abhängigkeitsanalyse mit ParZu beinhaltet. Der zentrale Grund ist, dass verschiedene Verwendungen von Toponymen von der Wortarterkennung als Substantive erkannt werden, obwohl sie semantisch mit einem Adjektiv gleichzusetzen sind. Beispiele hierfür sind hier etwa das „**Bamberger** ZIMMER“ oder die „**Würzburger** RESIDENZ“. Ohne eine Abhängigkeitsanalyse könnte hier nicht zweifelsfrei festgestellt werden, ob sich zwei nebeneinanderstehende Substantive aufeinander beziehen. Falls allerdings eine solche Konstruktion erkannt wurde, wird auch hier die Regel `adjective` als Feature eingetragen.

In einigen Fällen kann zudem ein Nomen, also auch ein Toponym, durch eine Genitivkonstruktion modifiziert werden, etwa um das Nomen exakter zu beschreiben. Ein Beispiel hierfür wäre „das **Bamberger** ZIMMER der **Würzburger** RESIDENZ“, worin das Zimmer durch seine Zugehörigkeit zur Residenz eindeutig definiert ist. Auch diese Variante kann nur dann erkannt werden, wenn Abhängigkeitsinformationen zwischen den einzelnen Token vorhanden sind. Hierzu wird noch ein Feature `subKind` eingeführt, das in diesem Fall mit `withModifier` versehen wird. Abbildung 4.11 zeigt hierzu die entsprechende ParZu-Baumstruktur, in der die Modifikation über die Klasse `gmod` erkannt wurde, welche den Hauptausdruck und das modifizierende Attribut verbindet. Abbildung 4.12 zeigt das Ergebnis eines Toponym Analyzer-Laufs für den gleichen Satz.

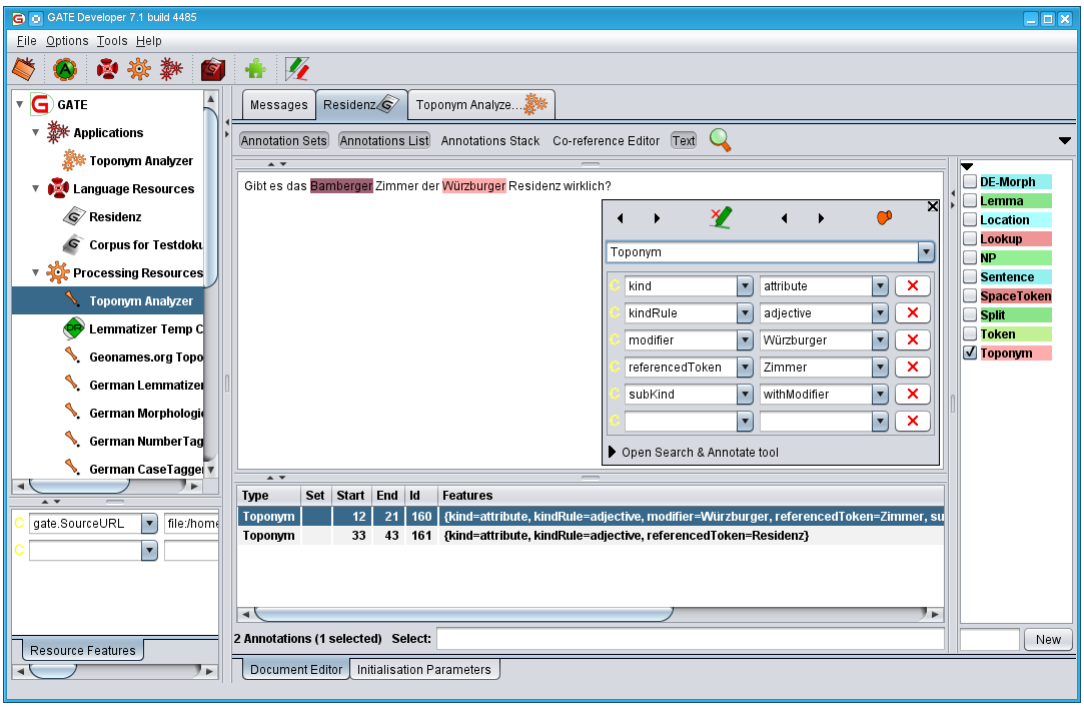

Abbildung 4.12 — Ausgabe von GATE für ein Toponym mit modifizierendem Attribut

4.3.2 Figuren

Features der Toponym-Annotation

Da die rhetorischen Figuren unabhängig von den beschriebenen Verwendungsarten auftreten und zudem verschiedene Stilmittel gleichzeitig vorkommen können, werden diese in einem neuen Feature namens `figure` abgelegt, welches zudem für mehrere Einträge vorbereitet ist. Auch hier wird die jeweils angewendete Regel als separates Feature abgespeichert, für Figuren und Tropen ist dies `figureRule`.

Aus den in Kapitel 2.1.2 vorgestellten, für Toponyme relevanten Figuren, werden bis auf das *Epitheton Ornans* alle von der Knowledge Source `RhetoricalFigureKnowledgeSource` unterstützt. Jede Figur wird mit ihrem entsprechenden englischen Namen, also `alliteration`, `geminatio`, `anaphora`, `epiphora`, `anadiplosis` und `comparison` (für *Similitudo*), ausgezeichnet.

Alliteration

Die Klangfigur der *Alliteration* lässt sich bereits nach erfolgtem Tokenizing sowie der Identifikation der entsprechenden Toponyme mit einem einfachen Algorithmus erkennen. Hierzu müssen ausschließlich die Anfangsbuchstaben der Wörter vor und nach dem Toponym mit dem Anfangsbuchstaben des Toponyms verglichen werden. Ist mit dem darauffolgenden oder dem davor stehenden Wort eine Übereinstimmung gefunden worden, wird das Toponym mit der rhetori-

```
input  : tokenFacades in a Sentence before the toponym under investigation
output: none
firstCharacterToponym ← toponym.toString().substring(0, 1);
tokenFacade ← tokenFacades[toponym.index − 1];
firstCharacterToken ← tokenFacade.toString().substring(0, 1);
if firstCharacterToken.equals(firstCharacterToponym) then
    toponym.addRhetoricalFigure(EToponymRhetoricalFigure.alliteration);
    toponym.addRhetoricalFigureRule(
    EToponymRhetoricalFigureRule.alliterationSingleCharacter);
end
```

Abbildung 4.13 — Pseudocode des Algorithmus zur Erkennung einer Alliteration

schen Figur `alliteration` markiert sowie als Regel `alliterationSingleCharacter` festgehalten. Der Pseudocode in Abbildung 4.13 zeigt diesen Algorithmus für Wörter vor dem Toponym. Für Wörter danach wird der gleiche Algorithmus verwendet, einziger Unterschied ist das Token nach dem Toponym als Eingabewert bei der Bestimmung der Stilfigur.

Geminatio

Die Positionsfigur der *Geminatio* lässt sich ähnlich wie die Alliteration ausschließlich mit Tokenizing und nach einer Toponymerkennung identifizieren. Abbildung 4.14 zeigt einen vereinfachten Pseudocode des zugehörigen Algorithmus. Im Gegensatz zur *Alliteration* können hier nicht zwei direkt aufeinander folgende Token daraufhin untersucht werden, ob beide den gleichen Ort referenzieren, da zwischen beiden noch ein Komma als Trennzeichen für eine Aufzählung folgen muss, welche eigenständige Token darstellen (siehe Zeile 10). Darüber hinaus können mehr als zwei gleiche Toponyme aufeinander folgen, die zudem alle mit der rhetorischen Figur `geminatio` und der Regel `standardGeminatio` annotiert werden müssen. Der Algorithmus sucht deshalb zunächst die gleichnamigen Toponyme in einem Satz und fährt mit der Analyse nur fort, wenn mindestens ein Duplikat erkannt wurde (Beginn bis Zeile 8). Ausgehend vom ersten Toponym im Satz wird analysiert, ob in einem Token danach ein Komma vor dem Prädikat folgt, was auf eine Aufzählung hindeutet. Alle gleichnamigen Toponyme nach dem Komma, aber vor dem Prädikat (Abbruch in Zeile 14), werden dann als *Geminatio* identifiziert.

Anapher

Ähnlich wie die *Geminatio* ist die Positionsfigur der *Anapher* durch eine Wortwiederholung charakterisiert. Diese kann sich allerdings sowohl auf mehrere Teile eines Satzes, aber auch auf mehrere Sätze beziehen, was in der Implementierung entsprechend zu berücksichtigen ist. Da die Datenstrukturen der Blackboard-Architektur des Toponym Analyzers auf einzelne Sätze ausgerichtet sind, müssen beide Varianten auch unterschiedlich implementiert werden. Die Erkennung der *Anaphern*, die sich auf verschiedene Sätze erstrecken, zeigt der Pseudocode in Abbildung 4.15.

```
input  : tokenFacades in a Sentence after the toponym under investigation, all
         toponymFacades of the current sentence
output: none
for toponymFacade ← toponymFacades[i ← 0] to i = toponymFacades.size() do
    if toponymFacade.equals(toponym) then
        sameToponyms + +;
    end
end
if sameToponyms < 2 then
    return;
end
for tokenFacade ← tokenFacades[i ← 0] to i = tokenFacades.size() do
    if token.isComma() then
        commaFound ← true;
    end
    if token.isVerb() then
        return;
    end
    if commaFound and tokenFacade.equals(toponym.getToken()) then
        toponym.addRhetoricalFigure(EToponymRhetoricalFigure.geminatio);
        toponym.addRhetoricalFigureRule(
        EToponymRhetoricalFigureRule.standardGeminatio);
    end
end
```

Abbildung 4.14 — Pseudocode des Algorithmus zur Erkennung einer Geminatio

Hier wird über alle `Sentence`-Fassaden iteriert (Zeile 1) und untersucht, ob sich zum einen mindestens ein Toponym in dem entsprechenden Satz befindet und zum anderen, ob sich dieses am Beginn des Satzes befindet (Zeilen 2 bis 6). Damit dieses Kriterium erfüllt ist, dürfen vor dem Toponym als Beiworte ausschließlich Artikel oder Präpositionen stehen (Zeile 10). Konnte ein solches Toponym identifiziert werden, wird der folgende Satz mit dem gleichen Muster analysiert (Zeile 14). Wurde eine Übereinstimmung gefunden, werden beide Toponyme mit dem Wert `anaphora` für das Feature `figure` und `anaphoraAcrossSentences` für das Feature `figureRule` versehen. Dieser Vergleich wird für folgende Sätze so lange wiederholt und entsprechende Toponyme annotiert, bis keine Übereinstimmung mehr gefunden werden kann.

Zur Erkennung von *Anaphern*, die sich zu Beginn verschiedener Teile eines Satzes befinden, lässt sich dieser Algorithmus mit wenigen Anpassungen analog anwenden. Zur Unterscheidung von der *Geminatio* muss zunächst festgestellt werden, ob sich nach dem ersten und vor dem folgenden Toponym ein Verb befindet. Wenn das nicht der Fall ist, bricht der Algorithmus ab, damit *Geminatios* nicht auch als *Anaphern* markiert werden. Als Trennzeichen der unterschiedlichen Satzteile wird darüber hinaus nicht nur ein Komma akzeptiert, sondern auch jegliche andere Interpunktion, da eine *Anapher* beispielsweise auch nach Bindestrichen vorkommen kann. Die entsprechenden Toponyme werden dann als *Anaphern* und mit der Regel `anaphoraWithinSentence` markiert, wenn sich weitere gleichartige Ortsbezüge zu Beginn der durch Interpunktion getrennten Satzteile befinden.

Epipher

Die *Epipher* als weitere Figur der Wortwiederholung entspricht im Wesentlichen der *Anapher*, bezeichnet aber gleiche Worte - bzw. im Kontext dieser Masterarbeit Toponyme, welche am Ende aufeinanderfolgender Sätze bzw. Satzteile auftreten. Auch hier müssen beide Varianten in verschiedenen Algorithmen erkannt werden und erhalten analog auch verschiedene Regeln. Als `figure` werden beide Alternativen mit `epiphora` versehen, während das Auftreten in verschiedenen Sätzen mit der `figureRule=epiphoraAcrossSentences` und Wiederholungen in einzelnen Satzteilen mit `epiphoraWithinSentence` annotiert werden. Abbildung 4.16 zeigt den zugehörigen Algorithmus für *Epiphern* über verschiedene Sätze hinweg, der folglich große Parallelen zu seinem Gegenstück für *Anaphern* aufweist. Als zentraler Unterschied werden hier außer einer Interpunktion als Satzende keine weiteren Token nach dem letzten Toponym im Satz akzeptiert (Zeile 10).

Anadiplosis

Bei der *Anadiplosis* als letzte unterstützte Positionsfigur handelt es sich wie in Kapitel 2.1.2 erläutert um eine Kombination der beiden zuvor eingeführten Stilmittel *Anapher* und *Epipher*. Wie diese kann sich auch die *Anadiplosis* über zwei Sätze erstrecken, oder nur in einem Satz bei der Verknüpfung zweier Satzteile Verwendung finden. Folglich gibt es auch hier wiederum zwei Algorithmen, welche beide Varianten abdecken. Abbildung 4.17 zeigt den Pseudocode des Algorithmus für die Erkennung einer *Anadiplosis* über Satzgrenzen hinweg. Zur Erkennung wird zunächst der Algorithmus zur Erkennung eines *Epipher*-Kandidaten am Ende eines Satzes oder

```
input  : sentenceFacades of all sentences in the text
output: none
for sentenceFacade ← sentenceFacades[i ← 0] to i = sentenceFacades.size() do
    if sentenceFacade.containsToponym() then
        firstToponym ← sentenceFacade.getFirstToponym();
    else
        continue;
    end
    isAnaphoraCandidate ← true;
    tokenFacades ← sentenceFacade.getTokensBeforeToponym(toponym);
    for tokenFacade ← tokenFacades[j ← tokenFacades.size] down to j = 0 do
        if !(tokenFacade.isAdposition() or tokenFacade.isArticle()) then
            isAnaphoraCandidate ← false;
        end
    end
    if isAnaphoraCandidate then
        if anaphoraCandidate = null then
            anaphoraCandidate ← firstToponym;
        else
            if firstToponym.equals(anaphoraCandidate) then
                anaphoraCandidate.addRhetoricalFigure(
                EToponymRhetoricalFigure.anaphora);
                anaphoraCandidate.addRhetoricalFigureRule(
                EToponymRhetoricalFigureRule.anaphoraAcrossSentences);
                firstToponym.addRhetoricalFigure(
                EToponymRhetoricalFigure.anaphora);
                firstToponym.addRhetoricalFigureRule(
                EToponymRhetoricalFigureRule.anaphoraAcrossSentences);
            else
                anaphoraCandidate ← firstToponym;
            end
        end
    end
end
```

Abbildung 4.15 — Pseudocode des Algorithmus zur Erkennung einer Anapher

```
input  : sentenceFacades of all sentences in the text
output: none
for sentenceFacade ← sentenceFacades[i ← 0] to i = sentenceFacades.size() do
    if sentenceFacade.containsToponym() then
        lastToponym ← sentenceFacade.getLastToponym();
    else
        continue;
    end
    isEpiphoraCandidate ← true;
    tokenFacades ← sentenceFacade.getTokensAfterToponym(toponym);
    for tokenFacade ← tokenFacades[j ← tokenFacades.size] to j = 0 do
        if !(tokenFacade.isPunctuation()) then
            isEpiphoraCandidate ← false;
        end
    end
    if isEpiphoraCandidate then
        if epiphoraCandidate = null then
            epiphoraCandidate ← lastToponym;
        else
            if lastToponym.equals(epiphoraCandidate) then
                epiphoraCandidate.addRhetoricalFigure(
                EToponymRhetoricalFigure.epiphora);
                epiphoraCandidate.addRhetoricalFigureRule(
                EToponymRhetoricalFigureRule.epiphoraAcrossSentences);
                lastToponym.addRhetoricalFigure(
                EToponymRhetoricalFigure.epiphora);
                lastToponym.addRhetoricalFigureRule(
                EToponymRhetoricalFigureRule.epiphoraAcrossSentences);
            else
                epiphoraCandidate ← lastToponym;
            end
        end
    end
end
```

Abbildung 4.16 — Pseudocode des Algorithmus zur Erkennung einer Epipher

Satzteils angewendet (Zeilen 5 bis 16) und daraufhin mit dem Algorithmus zur Erkennung eines *Anapher*-Kandidaten im darauf folgenden Satz oder Satzteil nach dem gleichen Toponym gesucht (Zeilen 19 bis 26). Ist eine Übereinstimmung gefunden, werden beide Topoynme mit dem Typ `anadiplosis` und der der Verwendungsart des Stilmittels entsprechenden Regel `anadiplosisAcrossSentences` oder `anadiplosisWithinSentence` annotiert.

Similitudo

Die Größe von Gebieten wird oft mit der von bekannten Orten, Ländern oder Regionen verglichen, weswegen die *Similitudo*, also der Vergleich, eine zentrale Stilfigur im Kontext von Toponymen darstellt. Allerdings ist es ausschließlich mit POS-Tagging in vielen Fällen nur schwer zu erkennen, ob sich das erkannte Vergleichspartikel auch auf eines der Toponyme im Satz bezieht. Aus diesem Grund wird die Erkennung von Vergleichen durch den Toponym Analyzer ausschließlich dann eingeschaltet, wenn Abhängigkeitsinformationen durch ParZu verfügbar sind. Wie Abbildung 4.18 zeigt, ist genau dann ein Vergleich mit einem Toponym gegeben, wenn das Elterntoken eines Toponyms ein Vergleichspartikel (`KOKOM`) ist und ParZu dieses in der zugehörigen Abhängigkeitsklasse (`kom`) entsprechend identifiziert hat.

Der zugehörige Algorithmus kann deshalb auch sehr kurz gehalten werden. Für ein erkanntes Toponym wird das Elterntoken ermittelt und die ParZu-Abhängigkeitsklasse geprüft. Ist diese ein *Vergleich*, erhält das Toponym den Wert `comparison` als Stilmittel und `parentIsComparativeConjunction` als Regel zugewiesen. Abbildung 4.19 zeigt den Pseudocode hierzu.

4.3.3 Tropen

Synekdoché

Wenn ein Toponym als *Synekdoché* eingesetzt wird, bezeichnet der Ort in vielen Fällen eine dort ansässige Regierung bzw. Verwaltung oder die hier lebende Bevölkerung. Eine Variante dieses Stilmittels existiert für die genauere Charakterisierung von Gesetzestexten. So bezieht sich in das Toponym „**Brüsseler** VORSCHRIFTEN“ auf die dort ansässige EU-Kommission bzw. das EU-Parlament. Zur Erkennung kann der entsprechende Algorithmus auf die bereits erkannten Verwendungsarten zurückgreifen, da es sich hier um ein Toponym handelt, das als *Attribut* verwendet wird. Da das vom Attribut referenzierte Token ebenfalls als Feature abgespeichert wird, muss nun nur noch festgestellt werden, ob dieses einen juristischen Text bezeichnet. Hierfür wird das lexikalisch-semantische Netz GermaNet eingesetzt. Um herauszufinden, ob ein Wort in diese Kategorie passt, müssen alle Oberbegriffe, also ihre *Hypernyme*, untersucht werden. Fällt ein Hypernym dieses referenzierten Tokens in die Rubrik *juristischer Text*, wird das Toponym als `synecdoche` annotiert und als Regel `governmentsMakeLaws` festgehalten. Abbildung 4.20 zeigt den Pseudocode dieses Algorithmus. Wie in Kapitel 2.2.2 dargestellt, werden Beziehungen in GermaNet in sogenannten Synsets modelliert. Folglich werden alle Hypernym-Synsets zum referenzierten Token ermittelt (Zeile 2) und diese dann entsprechend untersucht (Zeile 3).

Eine andere Variante ist die Verwendung eines Toponyms als Subjekt eines Satzes, bei dem das Prädikat des Satzes zu einer für ein Toponym untypischen Klasse gehört. Wenn Ortsbezüge im

```
input  : sentenceFacades of all sentences in the text
output: none
for sentenceFacade ← sentenceFacades[i ← 0] to i = sentenceFacades.size() do
    if !sentenceFacade.containsToponym() then
        continue;
    end
    if anadiplosisCandidate = null then
        lastToponym ← sentenceFacade.getLastToponym();
        isAnadiplosisCandidate ← true;
        tokenFacades ← sentenceFacade.getTokensAfterToponym(toponym);
        for tokenFacade ← tokenFacades[j ← tokenFacades.size] to j = 0 do
            if !(tokenFacade.isPunctuation()) then
                isAnadiplosisCandidate ← false;
            end
        end
        if isAnadiplosisCandidate then
            anadiplosisCandidate ← lastToponym;
        end
    else
        firstToponym ← sentenceFacade.getFirstToponym();
        if firstToponym.equals(anadiplosisCandidate) then
            isAnadiplosis ← true;
            tokenFacades ← sentenceFacade.getTokensBeforeToponym(toponym);
            for tokenFacade ← tokenFacades[j ← tokenFacades.size] down to j = 0 do
                if !(tokenFacade.isAdposition() or tokenFacade.isArticle()) then
                    isAnadiplosis ← false;
                end
            end
            if isAnadiplosis then
                anadiplosisCandidate.addRhetoricalFigure(
                EToponymRhetoricalFigure.anadiplosis);
                anadiplosisCandidate.addRhetoricalFigureRule(
                EToponymRhetoricalFigureRule.anadiplosisAcrossSentences);
                firstToponym.addRhetoricalFigure(
                EToponymRhetoricalFigure.anadiplosis);
                firstToponym.addRhetoricalFigureRule(
                EToponymRhetoricalFigureRule.anadiplosisAcrossSentences);
            end
        end
        anadiplosisCandidate ← null;
    end
end
```

Abbildung 4.17 — Pseudocode des Algorithmus zur Erkennung einer Anadiplosis

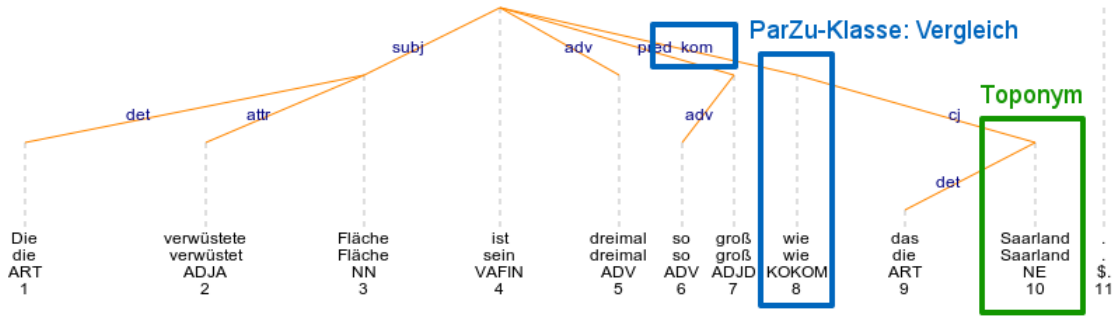

Abbildung 4.18 — Ausgabe von ParZu für einen Vergleich

```
input  : toponym under investigation
output: none
parentToken ← toponym.getParentToken();
if parentToken.getDependencyClass = ETokenDependencyClass.comparative then
    toponym.addRhetoricalFigure(EToponymRhetoricalFigure.comparison);
    toponym.addRhetoricalFigureRule(
    EToponymRhetoricalFigureRule.parentIsComparativeConjunction);
end
```

Abbildung 4.19 — Pseudocode des Algorithmus zur Erkennung eines Vergleichs

```
input  : referencedToken of a toponym that is used as attribute
output: none
hypernyms ← GermaNet.getHypernymSynsets(referencedToken.getLemma());
for hypernym ← hypernyms[i ← 0] to i = hypernyms.size() do
    if hypernym.getId() = Synsets.JuristischerText then
        toponym.addRhetoricalFigure(EToponymRhetoricalFigure.synecdoche);
        toponym.addRhetoricalFigureRule(
        EToponymRhetoricalFigureRule.governmentsMakeLaws);
    end
end
```

Abbildung 4.20 — Pseudocode des Algorithmus zur Erkennung einer Synekdoché für Gesetzestexte

eigentlichen Sinn verwendet werden, kommen nur wenige Prädikate in Frage, etwa *gegründet werden* oder *sich befinden*. Folglich wird das Auftreten einer anderen Verbklasse, also der überwiegenden Mehrheit der Verben, so interpretiert, dass das Toponym im Subjekt als *Synekdoché* verwendet wird. Um diese Variante erkennen zu können, müssen Informationen aus mehreren Wissensquellen verwendet werden. Abbildung 4.21 zeigt den Pseudocode dieses Algorithmus. Zunächst werden ausschließlich Toponyme untersucht, für die `standard` als Verwendungsart identifiziert werden konnte (Zeile 1). In den morphologischen Informationen muss für dieses Toponym zudem der Kasus *Nominativ* ermittelt worden sein (Zeile 4). Das Prädikat des Satzes wird über die ParZu-Abhängigkeitsinformationen festgestellt. Da es immer das Wurzelelement eines Satzes darstellt, muss der ParZu-Baum vom Toponym rekursiv bis zur Wurzel traversiert werden (Zeile 7). Falls die Wurzel ein Hilfsverb ist, wird für die Untersuchung noch in allen Kindelementen nach dem zugehörigen Vollverb gesucht (Zeile 14). Einzige Ausnahme hierfür ist die Verwendung von *wollen* ohne entsprechendes Vollverb (Zeile 8). Da ein Toponym in der Regel keinen eigenen Willen hat, wird in diesem Fall ohne eine weitere Analyse der Typ `synecdoche` festgestellt und als Regel `toponymsCantDoMuch` festgestellt.

Muss das Vollverb weiter analysiert werden, wird hier wiederum GermaNet zur Kategorisierung verwendet. Hier müssen die Synsets der Kategorie *Verben* ermittelt werden (Zeile 16). Können diese Synsets nicht in die Kategorien der *Schöpfung*, *Veränderung*, *Lokation* oder *Ort* eingeordnet werden (Zeile 20), konnte auch hier eine *Synekdoché* mit der Regel `toponymsCantDoMuch` erkannt werden.

Antonomasie

Die weiteren vorgestellten Stilmittel wie die Grenzverschiebungstropen *Antonomasie* und *Metonymie* oder die Sprungtropen *Metapher* und *Allegorie* lassen sich nach der Analyse verschiedener Verwendungen nicht mit entsprechenden Regeln abbilden. Hier werden die eigentlichen Ortsbezüge so verwendet, dass diese sowohl in einer wörtlichen Analyse als auch mit Datenbanken wie GermaNet oft nicht direkt mit den entsprechenden Stilmitteln in Verbindung gebracht werden können. Speziell bei Grenzverschiebungstropen können zwar in vielen Fällen die Toponyme über Gazetteers oder weitere Regeln (vgl. Kapitel 3) erkannt werden, da die Ortsnamen direkt verwendet werden. Um diese allerdings als *Antonomasie* oder *Metonymie* identifizieren zu können, sind in der Regel deutlich mehr kontextuelle Information nötig, als über eine Satzanalyse mit den angesprochenen Werkzeugen ermittelbar wäre.

Darüber hinaus ist die Grenze zwischen den Stilmitteln der *Metonymie* und der *Synekdoché* nicht ganz trennscharf zu ziehen und ist deshalb in der Fachwelt teilweise umstritten. Dies lässt sich beispielsweise an der Einordnung der Stilmittel durch Markert und andere (vgl. [MH02], [NM03], [MN09], [NJMS12]) erkennen, die alle Verwendungen von Worten im übertragenen Sinn insgesamt als *Metonymie* bezeichnet haben. So werden vom Toponym Analyzer die Stilmittel, welche nach der Definition in Kapitel 2.1.3 eigentlich als *Metonymie* klassifiziert werden müssen, oft in die Kategorie *Synekdoché* eingeordnet. Weitere Regeln für die *Metonymie* konnten nicht ermittelt werden, weswegen dieses Stilmittel vom neu entwickelten GATE-Plugin nicht direkt erkannt werden kann.

```
input  : The toponym under investigation
output: none
1  if toponym.getKind() <> EToponymKind.standard then
2  |  return;
3  end
4  if toponym.getMorph().getCase() <> EMorphCase.nominative then
5  |  return;
6  end
7  predicateToken ← toponym.getRootToken();
8  if predicateToken.getLemma() = wollen then
9  |  toponym.addRhetoricalFigure(EToponymRhetoricalFigure.synecdoche);
10 |  toponym.addRhetoricalFigureRule(
11 |  EToponymRhetoricalFigureRule.toponymsCantDoMuch);
12 |  return;
13 end
14 fullVerb ← predicateToken.getFullVerb();
15 synsets ← GermaNet.getSynsets(fullVerb.getLemma());
16 for synset ← synsets[i ← 0] to i = synsets.size() do
17 |  if !synset.inWordCategory(WordCategory.verben) then
18 |  |  continue;
19 |  end
20 |  if !(synset.inWordClass(WordClass.Schoepfung) or
   |  synset.inWordClass(WordClass.Veraenderung) or
   |  synset.inWordClass(WordClass.Lokation) or synset.inWordClass(WordClass.Ort))
   |  then
21 |  |  toponym.addRhetoricalFigure(EToponymRhetoricalFigure.synecdoche);
22 |  |  toponym.addRhetoricalFigureRule(
23 |  |  EToponymRhetoricalFigureRule.toponymsCantDoMuch);
24 |  |  return;
25 |  end
26 end
```

Abbildung 4.21 — Pseudocode des Algorithmus zur Erkennung einer Synekdoché auf Basis des Prädikats

```
Rule: Location_antonomasia
(
// Down Under --> Australien
( {Token.string == "Down"} {Token.string == "Under" } )
|
// Big Apple --> New York
( {Token.string == "Big"} {Token.string == "Apple" } )
|
// Der jüdische Staat --> Israel
( {Token.string =~ "[Jj]üdisch[e]*[rn]*"} {Token.lemma == "Staat" } )
|
// Stadt der Liebe --> Paris
( {Token.string == "Stadt"} {Token.string == "der"}
  {Token.string == "Liebe" } )
|
// Die ewige Stadt --> Rom
( {Token.string =~ "[Ee]wig[e]*[n]*"} {Token.lemma == "Stadt" } )
|
// Das fränkische Rom --> Bamberg
( {Token.string =~ "[Ff]ränkisch[e]*[sn]*"} {Token.lemma == "Rom" } )
|
// Das gelobte Land --> Israel
( {Token.string =~ "[Gg]elobt[e]*[sn]*"} {Token.lemma == "Land" } )
|
// Das heilige Land --> Israel
( {Token.string =~ "[Hh]eilig[e]*[sn]*"} {Token.lemma == "Land" } )
):antonomasia
-->
:antonomasia.Location = {kind = "Location", subType = "antonomasia",
                         rule = "Location_antonomasia"}
```

Abbildung 4.22 — Quelltext der JAPE-Regel für Antonomasien

Für die *Antonomasie* hingegen konnten einige Beispiele identifiziert werden, welche direkt über einen Gazetteer entsprechend annotiert werden können. Hierzu zählen etwa *Down Under* für Australien oder *Das Heilige Land* für Israel, weil hier auf geographische oder kulturelle Eigenarten der jeweiligen Länder Bezug genommen wird. Diese Beispiele werden über eine entsprechende JAPE-Regel der existierenden Datei `location.jape` im Verzeichnis `<GATE_DIR>/plugins/Lang_German/resources/grammar/` hinzugefügt. Abbildung 4.22 zeigt diese Regel mit den unterstützten Beispielen.

Durch diese Regel werden die entsprechenden Toponyme mit einer Annotation des Typs `Location` versehen. In dessen Feature `subType` wird dann das erkannte Stilmittel `antonomasia` hinterlegt. Der Toponym Analyzer wertet bei der Umwandlung der `Location` in `Toponym`-Annotationen

```
Rule: Location_allegory_combined
(
  // Uncle Sam --> USA
  ( {Token.string == "Uncle"} {Token.string == "Sam" } )
  |
  // Deutscher Michel --> Deutschland
  ( {Token.lemma =~ "[Dd]eutsch[e]*[rn]*"} {Token.lemma =~ "Michel[s]*" } )
):allegory
-->
:allegory.Location = {kind = "Location", subType = "allegory",
                      rule = "Location_allegory_combined"}

Rule: Location_allegory_gazetteer
(
  ({Lookup.minorType == "location_allegory"})
):allegory
-->
:allegory.Location = {kind = "Location", subType = "allegory",
                      rule = "Location_allegory_gazetteer"}
```

Abbildung 4.23 — Quelltext der JAPE-Regel für Allegorien

dieses Feature aus und setzt das `figure`-Feature hier entsprechend. Als `figureRule` wird hier der Wert `japeAnnotation` verwendet.

Allegorie und Metapher

Bei den Sprungtropen *Allegorie* und *Metapher* verhält es sich mit den Erkennungsmöglichkeiten ähnlich wie mit der *Antonomasie*, allerdings sind hier in der Regel ohne entsprechende Beispiele in Gazetteers nicht einmal mehr reine Toponyme identifizierbar, da der Ortsbezug nur über den Kontext oder entsprechendes Wissen über die passende Redewendung herstellbar ist. Infolgedessen werden sowohl für *Allegorien* als auch für *Metaphern* entsprechend eigene JAPE-Regeln implementiert. Zusätzlich zu den in Abbildung 4.23 beschriebenen Mehrwortausdrücken konnten noch einige weitere, insbesondere Nationalallegorien wie die *Helvetia* oder die *Bavaria*, in einen weiteren Gazetteer übernommen werden, der hier über die zweite Regel mit dem Typ `location_allegory` referenziert wird.

Abbildung 4.24 zeigt die Regel mit der aktuell einzigen unterstützten *Metapher*. In allen Fällen werden analog zum Vorgehen bei der *Antonomasie* im Toponym Analyzer alle Annotationen des Typs `Location` in ihre Entsprechungen vom Typ `Toponym` umgewandelt und der `subType` in das Feature `figure` kopiert. Auch hier wird in allen Fällen `japeAnnotation` als `figureRule` hinterlegt.

```
Rule: Location_metaphor
(
  // Wiege der Menschheit --> Süd/Ostafrika
  ( {Token.string == "Wiege"} {Token.string == "der"}
    {Token.string == "Menschheit" } )
):metaphor
-->
:metaphor.Location = {kind = "Location", subType = "metaphor",
                      rule = "Location_metaphor"}
```

Abbildung 4.24 — Quelltext der JAPE-Regel für Metaphern

5 Evaluation

5.1 Vorgehensweise

Im Zuge diese Masterarbeit entstanden verschiedene Modifikationen und Erweiterungen zu GATE. Um deren Qualität beurteilen zu können, müssen die Ergebnisse, welche mit den einzelnen Komponenten erzielt werden, mit einer Referenz, einem sogenannten *Goldstandard*, verglichen werden. Ein solcher Goldstandard besteht in der Regel aus einer Menge von Texten, in denen die Vorkommnisse der zu findenden Charakteristika bereits von entsprechenden Fachleuten, beispielsweise als Aufgabe in einer Fachkonferenz (vgl. u.a. Aufgabe 8 der SemEval-2007[1]), annotiert wurden und zur Messung der Erkennungsrate von Toponymen herangezogen werden. Diese Annotationen werden im Allgemeinen nicht von einer einzigen Person, sondern von einer Gruppe vorgenommen, wobei die Zuverlässigkeit der durch die Teilnehmer manuell vorgenommenen Annotationen über die Ermittlung und Interpretation statistischer Maße wie Cohens Kappa [Coh60] sichergestellt wird.

Diese *Referenzkorpora* werden in vielen Fällen immer wieder herangezogen, um die Qualität eines neuen Algorithmus zu dokumentieren, oder Verbesserungen an einer bestehenden Implementierung zu beweisen. Speziell bei der Toponymerkennung und -auflösung gibt es darüber hinaus bereits Bestrebungen, entsprechende Korpora mit einer einheitlichen Metadaten-Sprache zu annotieren [Lei06].

Wie in Kapitel 2.3.3 vorgestellt, haben sich andere Autoren nach meinem Kenntnisstand bislang ausschließlich auf die systematische Charakterisierung von Toponymen in englischen Texten beschränkt. Darüber hinaus unterscheiden sich die Charakteristika, welche für die bereits existierenden Analysen verwendet wurden, stark von denen in Kapitel 2.1 beschriebenen Kategorien. Infolgedessen gibt es für die Charakterisierung von Toponymverwendungen in Texten deutscher Sprache bislang keinen Referenzkorpus, der für die Evaluation der Ergebnisse dieser Masterarbeit herangezogen werden könnte. Die Erarbeitung eines eigenen Referenzkorpus, der in Umfang und Qualität mit den existierenden Korpora vergleichbar ist, würde allerdings den gesetzten Rahmen der Arbeit deutlich übersteigen. Deshalb wurde nur eine kleine Menge von 15 Texten manuell annotiert und diese dann als Goldstandard für die Evaluation herangezogen. Die Texte bestehen zum größten Teil aus Nachrichtentexten und Reiseberichten, beinhalten aber auch Testtexte mit seltenen Stilmitteln bzw. die erste Strophe des *Lieds der Deutschen* von August Heinrich Hoffmann von Fallersleben. Zwar ist diese historisch und politisch äußerst vorbelastet, allerdings beinhaltet sie eine der bekanntesten Verwendungen einer *Geminatio* im Zusammenhang mit Toponymen. Alle Texte erhalten einen eindeutigen, sprechenden Namen. Eine detaillierte

[1] `http://www.comp.leeds.ac.uk/markert/MetoSemeval2007.html` - abgerufen am 05. April 2014

Auflistung hierzu findet sich im Anhang. Darüber hinaus sind alle Texte mit den entsprechenden Referenzannotationen auch auf der beiliegenden DVD verfügbar.

Die hieraus gewonnenen Annotationen genügen so keinen einschlägigen Standards. Sie sollen allerdings ausschließlich dazu dienen, die Ergebnisse allgemein qualifizieren zu können. Hierbei werden sowohl die Toponymerkennung als auch ihre Charakterisierung entsprechend untersucht, da im Rahmen dieser Arbeit nicht nur der Toponym Analyzer zur Erkennung der Verwendungsarten und der Stilmittel entstanden ist, sondern auch diverse Erweiterungen an der eigentlichen Toponymerkennung in GATE implementiert wurden. Zur Evaluation dieser Erkennungsraten werden die von der Anwendungspipeline ermittelten `Location`-Annotationen mit denen des Goldstandards verglichen. Die einzelnen Features, also z.B. die Details darüber, welche Regel bei der Ermittlung zur Anwendung kam, spielen für das Ergebnis keine Rolle, weswegen nur die Annotationen an sich untersucht werden. Bei den `Toponym`-Annotationen verhält es sich allerdings anders. Da alle `Location`-Annotationen direkt in solche des Typs `Toponym` umgewandelt werden, wird es hier keine qualitativen Unterschiede geben. Die Informationen darüber, welche Toponymverwendungsart oder welches Stilmittel gefunden wurde, finden sich hier in den Features `kind` und `figure`. Deshalb werden hier diese Features in der Referenz ausgezeichnet und für die Evaluation mit den Ergebnissen des Toponym Analyzers verglichen.

Zur Messung der Ergebnisse kommen die im Information Retrieval bekannten Metriken *Recall*, *Precision* und *F-Measure* zum Einsatz. Da bei der Evaluation allerdings die von der kompletten Anwendungspipeline gefundenen Toponyme und die vom Toponym Analyzer ermittelten Charakteristika untersucht werden, beziehen sich diese Maße nicht auf komplette Dokumente, sondern auf Annotationen (`Location`) bzw. deren Features (`kind` und `figure` der Annotation `Toponym`). Da es für ein Wort oder eine Wortgruppe mehrere Stilmittel geben kann (vgl. Kapitel 4.3.2), wird jede Fundstelle eines Stilmittels getrennt voneinander betrachtet, unabhängig davon, ob ein einziges oder mehrere bei einem Wort erwartet werden.

GATE bietet in der ausgelieferten Version bereits diverse Werkzeuge zur Evaluation einer Anwendungspipeline mit einzelnen Dokumenten oder einem ganzen Korpus an[2]. Erste Tests haben allerdings gezeigt, dass es beim Vergleich zu verschiedenen Problemen kommen kann, die das Ergebnis verfälschen. Dies betraf in einigen Fällen GATE-eigene Metadaten oder manuell annotierte Arrays wie sie beispielsweise beim `figure`-Feature verwendet werden. Darüber hinaus gehen in die von GATE verwendeten Formeln für die einzelnen Metriken je nach Variante noch zusätzliche Kategorien wie *teilweise korrekt (partial)* oder *falsch (spurious)* mit ein[3]. In der Evaluation der vorliegenden Anwendungspipeline sollen die gefundenen Annotationen bzw. Features jedoch hart in *relevant* oder *nicht relevant* eingeteilt werden. Relevant sind hierbei ausschließlich die `Location`-Annotationen, die komplett einer entsprechenden Annotation aus der Referenz entsprechen. Hinzu kommen die `kind`- und `figure`-Features, die exakt so im Standard verzeichnet sind. Um die Fehler durch das GATE-Evaluationsframework auszuschließen und ein konsistentes Ergebnis auf Basis der genannten Prinzipien zu erreichen, wird das *Annotation Diff*-Werkzeug von GATE ausschließlich zur Grobbestimmung der Ergebnisse eingesetzt. Die konkrete Berechnung der Ergebnisse erfolgt dann separat mit den folgenden Formeln. Fundstellen bezeichnen wie oben erläutert entweder Annotationen oder Featurewerte. Die Stellen, welche vom Toponym

[2] `http://gate.ac.uk/sale/tao/splitch10.html#chap:eval` - abgerufen am 07. April 2014
[3] `http://gate.ac.uk/sale/tao/splitch10.html#sec:eval:prf` - abgerufen am 07. April 2014

Analyzer als Toponym annotiert wurden, oder für die eine Verwendungsart bzw. ein Stilmittel erkannt wurde, werden als *positive Fundstellen* bezeichnet. Vom Toponym Analyzer korrekt erkannte Fundstellen werden als *richtig positive Fundstellen* definiert. Die Gesamtzahl aller vom Programm gefundenen Annotationen oder Features, welche sowohl die *richtig positiven* als auch die *falsch positiven* beinhalten, werden einfach als *Fundstellen* bezeichnet.

$$Recall = \frac{\text{Anzahl richtig positiver Fundstellen}}{\text{Anzahl positiver Fundstellen}} \tag{5.1}$$

$$Precision = \frac{\text{Anzahl richtig positiver Fundstellen}}{\text{Anzahl Fundstellen}} \tag{5.2}$$

Für das *F-Measure* werden *Recall* und *Precision* gleich gewichtet, woraus sich wiederum folgende Formel ergibt:

$$F - Measure = \frac{2 \cdot Recall \cdot Precision}{Recall + Precision} \tag{5.3}$$

Zur Evaluation der Toponymerkennung werden die verschiedenen Ergebnisse, welche mit der ausgelieferten GATE-Version (vgl. Kapitel 3.1.2, *GATE pur*) erzielt werden konnten, mit denen verglichen, die unter Zuhilfenahme des mit OpenGeoDB-Daten angereicherten Gazetteers (vgl. Kapitel 3.2.3, *GATE erweitert*), sowie darüber hinaus mit Hilfe des Stanford NER-Parsers (vgl. Kapitel 3.2.2, *GATE erweitert, Stanford*) und des geonames.org-Plugins (vgl. Kapitel 3.2.4, *GATE erweitert, Stanford, geonames.org*) ermittelt werden konnten.

Da die Toponymchakterisierung auf den Ergebnissen der Toponymerkennung aufbaut, können die Ergebnisse für Verwendungsarten und Stilmittel maximal so gut sein wie die der eigentlichen Toponyme. Da speziell die Stilmittel nicht mit jedem Ortsbezug verwendet werden, wirken sich grundlegende Fehler hier besonders stark aus. Deshalb wird für die Messung der Toponymcharakterisierungen je Dokument ausschließlich die Anwendungspipeline herangezogen, welche für die Toponymerkennung das beste Ergebnis nach dem *F-Measure* erzielt hat. Zur Ermittlung des Gesamtergebnisses wird der gesamte Korpus als ein großes Dokument betrachtet und die Metriken auf die Summe der einzelnen Fundstellen in den Dokumenten angewendet. Dies entspricht dem von GATE genutzten *Micro Averaging*[4] und soll verhindern, dass Ergebnisse verhältnismäßig kurzer Dokumente überproportional stark das Ergebnis beeinflussen.

5.2 Ergebnisse

5.2.1 Toponymerkennung

In den 15 Texten konnten nach manueller Annotation insgesamt 479 Toponyme identifiziert werden. Da für jedes Toponym auch eine Verwendungsart erwartet wird, beträgt ihre Anzahl ebenfalls 479. Die Zahl der gefundenen Stilmittel liegt allerdings mit 129 deutlich darunter. Da

[4] `http://gate.ac.uk/sale/tao/splitch10.html#sec:eval:macromicro` - abgerufen am 07. April 2014

GATE pur	GATE erweitert	GATE erweitert, Stanford	GATE erweitert, Stanford, geonames.org
2	2	7	4

Tabelle 5.1 — Anzahl der Texte mit höchstem F-Measure je Anwendungspipeline

die Texte in vielen Fällen speziell mit Hinblick auf eine hohe Vorkommenshäufigkeit von Stilmitteln ausgewählt wurden, dürfte ihr Anteil in einem größeren Korpus mit einem repräsentativeren Querschnitt noch deutlich unter dem gefundenen Viertel dieses Korpus liegen.

Die Ergebnisse der Toponymerkennung unterschieden sich zwischen den vier Anwendungspipelines je nach Text teils erheblich voneinander. Konnte in wenigen Fällen schon die mit GATE ausgelieferte Lösung, welche auf bestimmten Grammatikregeln und einem Gazetteer beruht, das beste Ergebnis vorlegen, so waren in der Mehrzahl der 15 Texte entweder die Hinzuziehung des Stanford NER-Parsers oder des geonames.org-Plugins nötig, um den im Vergleich besten *F-Measure* zu erzielen. Tabelle 5.1 zeigt hier die genauen Zahlen.

Wie stark die Ergebnisse teilweise zwischen den einzelnen Texten variierten, zeigen die folgenden Beispiele, in denen *Recall*, *Precision* und *F-Measure* für die einzelnen Anwendungspipelines einzeln je Beispiel aufgelistet sind. In der Mehrzahl der Fälle konnten mit der erweiterten GATE-Infrastruktur und dem Stanford NER-Parser die besten Ergebnisse erzielt werden. Bei diesen Texten war es in der Regel so, dass mit der klassischen und der erweiterten GATE-Infrastruktur zunächst zwar hohe Werte für die *Precision* ermittelt werden konnten, allerdings der *Recall* auf einem eher niedrigen Niveau blieb. Zusammen mit dem Stanford-Parser wuchs die Erkennungsrate dann deutlich stärker als die Fehlerrate, was zum besten Ergebnis führte. Wurde hier zusätzlich noch das geonames.org-Plugin hinzugefügt, konnte oft zwar der *Recall* noch weiter gesteigert werden, allerdings fiel die *Precision* aufgrund vieler falsch erkannter Toponyme so weit ab, dass das *F-Measure* noch unter das vorige Ergebnis fiel. Abbildung 5.1 zeigt ein solches Ergebnis am Beispiel eines Nachrichtentextes zur Krimkrise 2014, welcher insgesamt 59 Toponyme beinhaltet[5].

In anderen Fällen war die Erkennung mit der klassischen oder erweiterten GATE-Infrastruktur bereits so gut, dass bereits mit dem Stanford NER-Parser die *Precision* so weit fiel, dass das Gesamtergebnis hinter dem vorigen ohne ihn zurückblieb. Das geonames.org-Plugin fügte dann noch deutlich mehr Fehler hinzu, was sich in Abbildung 5.2 zu einem Artikel über den Finanzplatz Schweiz[6] mit insgesamt 22 Toponymen sehr deutlich zeigt. Ein analoges Bild ergab sich erwartungsgemäß bei den Texten, die vor allem oder ausschließlich Toponyme aus dem deutschen Sprachraum enthielten.

Wurden jedoch in den Texten viele Toponyme verwendet, die nicht aus dem deutschen oder englischen Sprachraum kommen, konnten die Ergebnisse sowohl mit der GATE-Infrastruktur an sich als auch zusammen mit dem Stanford NER-Parser nicht überzeugen. Erst die Nutzung des geonames.org-Plugins ergab eine Erkennungsrate auf Augenhöhe mit vielen anderen Texten,

[5] `http://www.welt.de/politik/ausland/article126543003/Macht-Putin-Ernst-hat-die-Ukraine-keine-Chance.html` - abgerufen am 13. April 2014

[6] `http://www.nzz.ch/aktuell/startseite/article8NM14-1.208282` - abgerufen am 13. April 2014

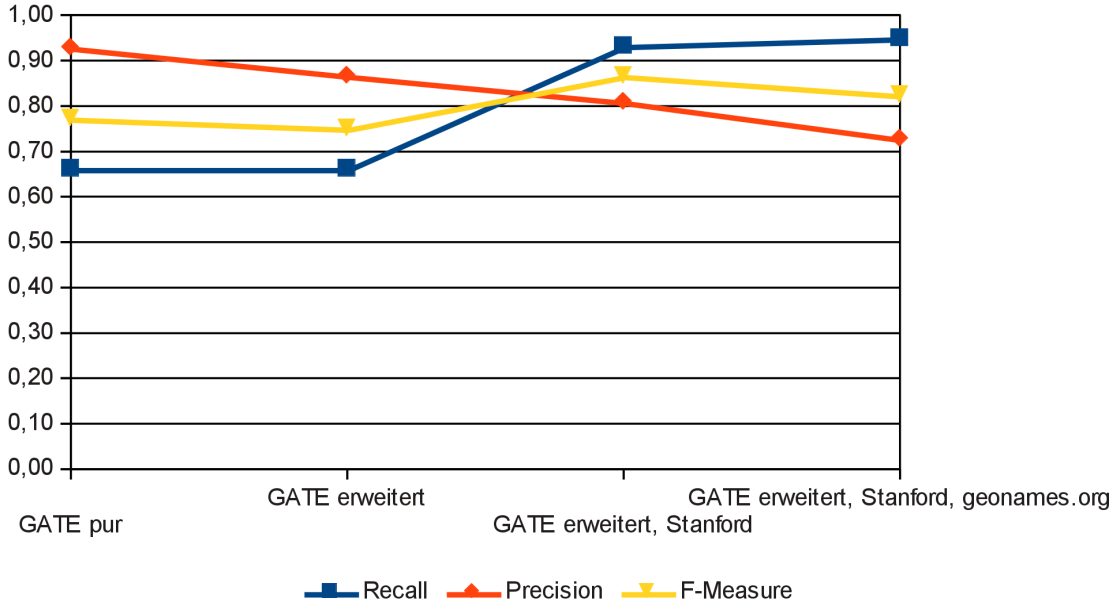

Abbildung 5.1 — Toponymerkennungsraten des Textes *putin*

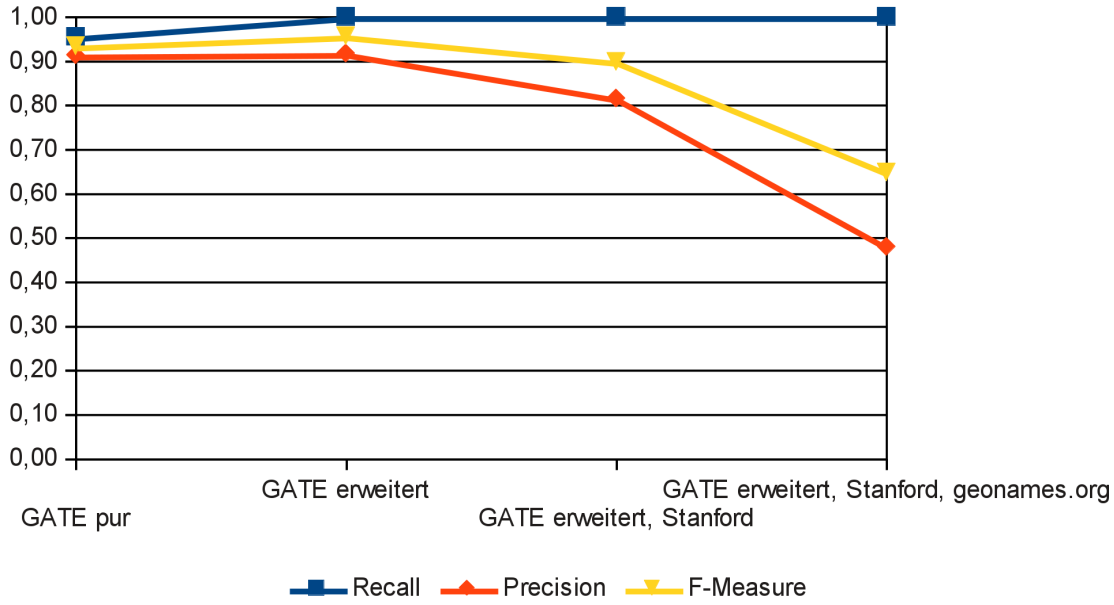

Abbildung 5.2 — Toponymerkennungsraten des Textes *uncle-sam*

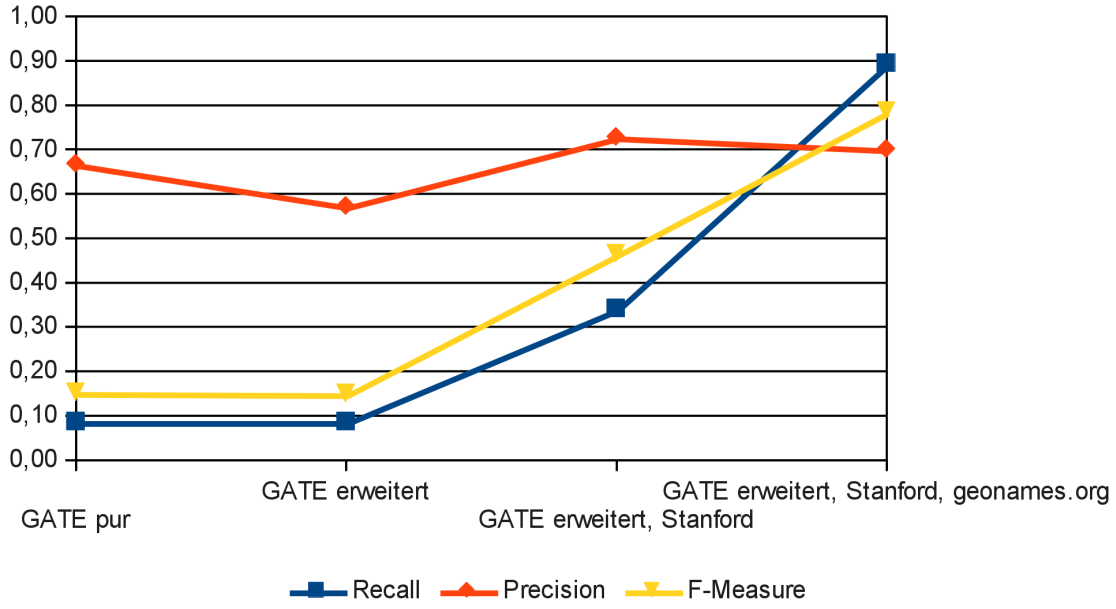

Abbildung 5.3 — Toponymerkennungsraten des Textes *island*

in denen das Plugin noch verhältnismäßig viele Fehler produzierte. Ein Beispiel hierfür ist ein Auszug aus dem Lonely Planet Reiseführer für Island mit insgesamt 47 Toponymen [PBP13], dessen Ergebnisse in Abbildung 5.3 zu sehen sind. Ein ähnliches, wenn auch nicht so ausgeprägtes Bild ergab sich beispielsweise bei einem Text über die Bergen-Bahn in Norwegen[7] oder einem Artikel über die Autoproduktion in Rumänien[8].

Wird nun der gesamte Evaluationskorpus betrachtet, ergibt sich ein Bild, das bereits beim Blick auf Tabelle 5.1 zu erwarten war. Insgesamt gesehen lieferte die Kombination aus erweiterter GATE-Infrastruktur und dem Stanford-NER-Parser mit Blick auf das *F-Measure* das beste Gesamtergebnis, wobei die mit GATE ausgelieferte Lösung die beste *Precision* und die Anwendungspipeline mit dem geonames.org-Plugin den besten *Recall* lieferte. Die Gründe hierfür sind hauptsächlich darin zu sehen, dass GATE ohne Erweiterungen zwar einen vergleichsweise kleinen Wortschatz in seinen Gazetteers mitliefert, das aber zusammen mit den zugehörigen Grammatikregeln für eine relativ zuverlässige Toponymerkennung sorgt. Je mehr Erweiterungen hinzugefügt werden, desto mehr *False Positives*, etwa aufgrund von Geo/Non-Geo-Ambiguitäten, werden gefunden. Dies trifft besonders auf das geonames.org-Plugin zu, das prinzipiell jedes Wort, welches irgendwo auf der Welt auch ein Toponym sein kann, auch als solches annotiert.

Tabelle 5.2 zeigt nun die konkreten Ergebnisse für alle Toponyme im Korpus. Die Anwendungspipeline mit dem erweiterten GATE und dem Stanford NER-Parser übertrifft mit einem *F-Measure* von 0,84 klar die anderen Varianten. Die Kombination mit dem geonames.org-Plugin folgt mit

[7] `http://www.merian.de/magazin/norwegen-bahnfahrt-von-oslo-nach-bergen.html` - abgerufen am 13. April 2014

[8] `http://www.welt.de/wirtschaft/article124211568/In-Rumaenien-liegt-die-Zukunft-des-deutschen-Autos.html` - abgerufen am 13. April 2014

	GATE pur	GATE erweitert	GATE erweitert, Stanford	GATE erweitert, Stanford, geonames.org
Recall	0,65	0,70	0,85	0,95
Precision	0,87	0,85	0,83	0,67
F-Measure	0,75	0,77	0,84	0,79

Tabelle 5.2 — Ergebnisse der Toponymerkennung des gesamten Evaluationskorpus

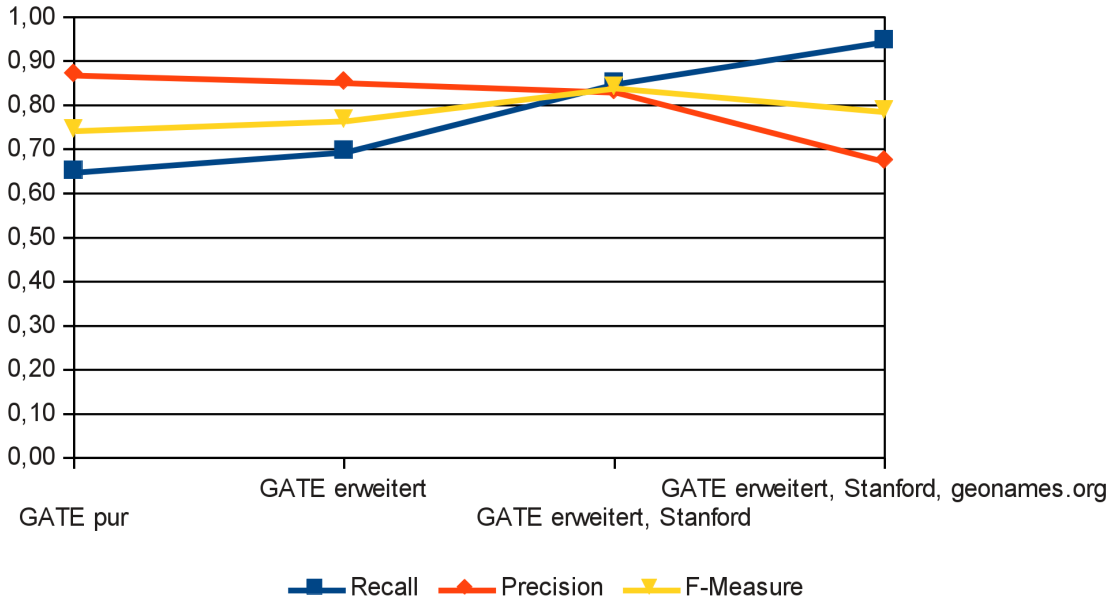

Abbildung 5.4 — Toponymerkennungsraten des gesamten Evaluationskorpus

einem Gesamtergebnis von 0,79, auf den weiteren Plätzen reihen sich das erweiterte GATE mit 0,77 und das GATE in der Auslieferungsversion mit 0,75 ein. Abbildung 5.4 visualisiert dieses Ergebnis entsprechend.

5.2.2 Toponymcharakterisierung

Verwendungsarten

Zur Erkennung der Verwendungsarten von Toponymen und der zusammen mit Ortsbezügen verwendeten Stilmittel wurde wie in Abschnitt 5.1 beschrieben immer das je Text beste Ergebnis der Toponymerkennung verwendet. Auf dieser Grundlage wurden nun die in Kapitel 4.3 vorgestellten Algorithmen mit dem Toponym Analyzer auf die 15 Texte angewendet. Die Auswertung der Ergebnisse zeigt, dass das Programm in den meisten Fällen korrekt arbeitet. So konnten

Recall	Precision	F-Measure
0,90	0,77	0,83

Tabelle 5.3 — Ergebnisse der Toponymcharakterisierung bei den Verwendungsarten

Recall	Precision	F-Measure
0,78	0,85	0,81

Tabelle 5.4 — Ergebnisse der Toponymcharakterisierung bei den Stilmitteln

von den 479 erwarteten Verwendungsarten 430 korrekt erkannt werden. Bei einer Zahl von 559 erkannten Fundstellen ergibt das, wie in Tabelle 5.3 ersichtlich ist, ein *F-Measure* von 0,83, was knapp an die 0,84 der besten Anwendungspipeline bei den Toponymverwendungen heranreicht.

Probleme ergaben sich hier fast ausschließlich bei *Referenzen* in Aufzählungen oder bestimmten *Attributen* an Substantiven. Ein Beispiel für eine solche Aufzählung ist der folgende Satz aus dem Text *down-under*:

> Die britische Königin ist Staatsoberhaupt in 15 weiteren Ländern,
> darunter IN **Australien**, **Kanada** und **Neuseeland**.

Die Präposition IN bezieht sich hierbei semantisch auf alle drei Länder. Der verwendete Algorithmus erkennt allerdings nur das erste Toponym als `reference` und charakterisiert die beiden folgenden Länder mit `standard`. Ein Beispiel für ein nicht korrekt erkanntes *Attribut* findet sich im Text *uncle-sam*:

> Auch haben die **US**-Aufseher das Recht, Prüfungsgesellschaften zu inspizieren.

Obwohl in dem zusammengesetzten Nomen der Teil **US** ein beschreibendes Attribut des Worts *Aufseher* darstellt, wird er nicht als solcher erkannt. Stattdessen wird hier die Verwendungsart `standard` gefunden.

Stilmittel

Von denen im Goldstandard manuell festgestellten 129 Stilmitteln konnten vom Toponym Analyzer 100 korrekt erkannt werden. Bei insgesamt 117 erkannten Stilmitteln ergibt das ein *F-Measure* von 0,81, die weiteren Metriken zeigt Tabelle 5.4.

Wie das Endergebnis zeigt, kann der im Rahmen dieser Masterarbeit neu entwickelte Toponym Analyzer bereits jetzt einen Großteil der Stilmittel aus dem Referenzkorpus korrekt erkennen. Bei einem Blick auf die Details ergibt sich hier jedoch ein differenziertes Bild. Bei den Klang- und Positionsfiguren gibt es nur vereinzelt Fehler, die insgesamt kaum ins Gewicht fallen. Anders stellt sich die Situation allerdings in Teilen bei den Satzfiguren und insbesondere bei den

Tropen dar. Bei der Satzfigur des *Vergleichs* kann der Toponym Analyzer ähnlich wie bei den Verwendungsarten Aufzählungen nicht korrekt verarbeiten. Ein Beispiel hierfür ist der folgende Satz aus dem Text *wolgograd*:

In manchen Provinzen WIE **Tschetschenien** und **Inguschetien**
haben sie an Einfluss verloren.

Hier wird ausschließlich die Provinz **Tschetschenien** korrekt als Vergleichsobjekt erkannt. **Inguschetien** wird nicht mit dem Feature `comparison` versehen.

Die Gründe für die falsch oder nicht erkannten Tropen sind zudem deutlich vielfältiger als bei den Verwendungsarten. Ein gewisser Teil der Fehler ergibt sich aus vielen verschiedenen Einzelfällen, die sich nur schwer in Kategorien einsortieren lassen. So treten vereinzelt echte *Metonymien* auf, welche vom Toponym Analyzer nicht erkannt werden können. Ein Beispiel hierfür ist eine Aussage des diesjährigen Bundespräsidenten der Schweiz, Didier Burkhalter, im Text *gauck-schweiz*:

Er wünsche sich vielmehr „etwas mehr **Schweiz** in **Europa**“.

Die **Schweiz** ist hierbei nur mehr ein Begriff für die Eigenschaften, die dem Schweizer Staat, seiner Regierung, seinem politischen System oder seinen Bürgern zugemessen werden, was vom Toponym Analyzer nicht erkannt wird. Der Großteil der Fehler bei den Stilmitteln ergibt sich allerdings aus falsch oder nicht zugeordneten Vorkommnissen der *Synekdoché*. Häufig war die Ursache hierfür, dass in einigen Fällen bei der morphologischen Analyse der Kasus des Toponyms nicht korrekt als Nominativ identifiziert wurde, woraufhin die entsprechende Regel nicht angewendet werden konnte. So wurde im folgenden Satz aus dem Text *figuren* das erste **Deutschland** als Akkusativ erkannt und das eigentliche Akkusativobjekt *dieses* zum Subjekt des Satzteils.

Amerika macht nichts,
Deutschland macht dieses, **Deutschland** macht jenes,
Europa macht alles.

Ein weiteres Beispiel für eine nicht erkannte *Synekdoché* findet sich im Text *nato*. Hier wird das Toponym **Russland** zusammen mit einer Präposition verwendet. Zudem kann der Toponym Analyzer aus dem Prädikat des Satzes keine explizite Handlung ableiten, was sonst auf dieses Stilmittel hindeuten würde:

Eine NATO-Mitgliedschaft der **Ukraine** und **Georgiens** ist für **Russland**
erklärtermaßen eine rote Linie.

6 Zusammenfassung und Ausblick

6.1 Ergebnisse der Arbeit

Wie die ermittelten Zahlen im Kapitel 5.2.2 verdeutlichen, ist es bereits mit dem im Rahmen dieser Arbeit entstandenen Toponym Analyzer möglich, Toponymverwendungen in deutschen Texten zu charakterisieren. Speziell bei denen in Kapitel 2.1.1 definierten Verwendungsarten treten nur für Einzelfälle wie Aufzählungen oder spezielle Attribute wie *US*- oder *EU*- noch Fehler auf, die in einer zukünftigen Version der Software noch korrigiert werden können. Die in Kapitel 4.3.1 angesprochene fehlende Implementierung der Erkennung von Postpositionen machte sich hier nicht weiter negativ bemerkbar, da diese im Vergleich zu den Prä- und Zirkumpositionen nur äußert selten verwendet werden.

Auch bei den Stilmitteln konnte der überwiegende Teil der Stellen, welche im Referenzkorpus entsprechend markiert waren, korrekt identifiziert werden. Allerdings ist die Fehlerquote hier signifikant höher als noch bei den Verwendungsarten. Zum einen liegt das auch hier bei der fehlenden Erkennung von Aufzählungen, zum anderen allerdings sowohl an der mangelnden Unterstützung von Metonymien als auch an verschiedenen Problemen mit Vorkommnissen von Synekdochen. Wie bereits in Kapitel 2.1.2 erwähnt, ist es algorithmisch schwierig, echte Metonymien als solche zu erkennen und entsprechend zu annotieren. Die konkreten Stellen in den untersuchten Texten bestätigten diese These nochmals. Hingegen könnten die Fehlerraten bei einigen der in den Texten vorkommenden Synekdochen durch eine Erweiterung der Algorithmen des Toponym Analyzers entsprechend vermindert werden.

Da die Toponymchakterisierung immer auf einer Toponymerkennung aufbaut, können die Resultate der Identifikation der entsprechenden Verwendungsarten und Stilmittel maximal so gut sein wie die genutzten Plugins. So ist ein gewisser Teil der Fehler auf falsche Informationen dieser Komponenten zurückzuführen, die in der Anwendungspipeline vor dem Toponym Analyzer ausgeführt werden. Der POS-Tagger TreeTagger und der für die Ermittlung von Abhängigkeitsinformationen eingesetzte ParZu waren nur in Einzelfällen für Probleme verantwortlich, die dann auch auf die *Charakterisierung von Toponymverwendungen* Einfluss hatten. Anders verhielt es sich allerdings bei dem Plugin zur Ermittlung morphologischer Informationen. Wahrscheinlich bedingt durch die Komplexität der deutschen Grammatik wurden hier in einigen Fällen falsche Kasus für Satzteile mit Toponymen bestimmt, was zu falschen oder übersehenen Treffern führte.

Einen großen Einfluss hatten allerdings die Komponenten, die selbst für die Toponymerkennung zuständig waren. Wie die Ergebnisse in Kapitel 5.2.1 veranschaulichen, war es für eine gute Toponymcharakterisierung notwendig, die mit GATE ausgelieferte Technologie zu verbessern und mit anderen Komponenten zu erweitern. Die Verbesserungen durch die Integration der OpenGeoDB-Daten in einen eigenen GATE-Gazetteer sind zwar mit den verwendeten Texten

des Referenzkorpus messbar, fallen allerdings nicht so stark aus wie die bloßen Zahlen in Kapitel 3.2.3 nahelegen. Dies dürfte im Wesentlichen daran liegen, dass einige Texte absichtlich dahingehend ausgewählt wurden, dass auch Toponyme von außerhalb des deutschen Sprachraums Verwendung fanden. Den größten Einfluss auf die Erkennungsraten hatte allerdings quer über alle Texte die Integration des Stanford NER-Parsers mit einem von Faruqui und Padó bereitgestellten Klassifizierer für die deutsche Sprache. Die Kombination aus dieser Software und dem erweitertem GATE konnte so für die Mehrzahl der Texte das beste Ergebnis erzielen. Eine Toponymerkennung unter Zuhilfenahme der Daten von geonames.org war hingegen nur für ausgewählte Texte sinnvoll, da sie in vielen Fällen zu viele falsche Treffer lieferte. Dennoch kann der Toponym Analyzer speziell in Texten mit weniger gebräuchlichen Ortsnamen, wie sie etwa in Reiseführern vorkommen, von dieser Komponente profitieren. Insgesamt haben die Erweiterungen jedoch eindeutig einen positiven Beitrag zur Toponymerkennung geleistet. Somit konnte eine bessere Grundlage zur anschließenden Toponymcharakterisierung gelegt werden.

Eine bessere Interpretation der Zahlen für die Toponymcharakterisierung wäre ggf. dann möglich gewesen, wenn in der Evaluation nicht nur die von der Anwendungspipeline ermittelten Toponyme verwendet worden wären, sondern der Toponym Analyzer seine Arbeit auf Basis von manuellen Referenzannotationen hätte durchführen können. Hierzu hätte allerdings nicht nur ein Referenzkorpus mit korrekten `Location`-Annotationen, sondern auch mit allen anderen notwendigen Informationen, etwa aus der morphologischen Analyse, bereitgestellt werden müssen, was sich am Ende als zu aufwendig für diese Masterarbeit herausgestellt hat.

Die Entscheidung, GATE als Grundlage für diese Masterarbeit zu verwenden, erwies sich am Ende als passend. Die bereits hierfür existierenden Komponenten und der modulare Aufbau ermöglichten es, einerseits mit einer guten Grundlage die ersten Schritte bei der Ermittlung der Verwendungsarten und Stilmittel gehen zu können und andererseits diese im Laufe der Bearbeitungszeit immer wieder mit notwendig gewordenen Technologien wie ParZu oder dem Stanford NER-Tagger zu erweitern. Die Integration neuer Komponenten wurde allerdings auch durch die im Toponym Analyzer genutzte Blackboard-Architektur stark erleichtert.

Jedoch gab es bei der Arbeit mit GATE speziell bei der manuellen Annotation des Referenzkorpus und der anschließenden Evaluation der Ergebnisse verschiedene Probleme. So wechselte der Fokus des Cursors auf der Benutzeroberfläche beim Editieren der Annotationen oft scheinbar willkürlich vom amodalen Editor-Popup auf den Texteditor, was das Arbeiten massiv erschwerte. So musste zu jedem Zeitpunkt darauf geachtet werden, dass die Annotation und nicht der Text selbst verändert wurde. Weiterhin sind einige Komponenten darauf angewiesen, dass die Textkodierung korrekt über einen Laufzeitparameter definiert wird. Standardmäßig war hier oft ISO-8859-1 voreingestellt. Da in dieser Arbeit jedoch viele Texte mit Unicode-Zeichen verwendet wurden, musste in jeder Komponente sichergestellt sein, dass diese auch im UTF-8-Format prozessiert wurden, um Fehler bei verschiedenen Annotationen und der Analyse zu vermeiden.

Insgesamt lieferte die genutzte Software zusammen mit den Neuentwicklungen als Komponenten von GATE jedoch eine gute Grundlage, um das eingangs in Kapitel 1.1 gesetzte Ziel der Erkennung der in Texten verwendeten Stilmittel im Zusammenhang mit Toponymverwendungen zu erreichen. Zusammenfassend bietet diese Arbeit somit die nach meinem Kenntnisstand erste systematische *Charakterisierung von Toponymverwendungen* für Texte in deutscher Sprache an. Das kann eine gute Basis dafür sein, mit weiterer Software die zu Beginn dieser Arbeit ebenfalls

angesprochenen Informationsbedürfnisse der Nutzer besser zu befriedigen, indem die Ergebnisse etwa dazu herangezogen werden, um das Genre des Textes zu erkennen. Weiterhin bietet die Arbeit allerdings auch unabhängig von der verwendeten Sprache einen neuen Ansatz zur Kategorisierung der Stilmittel im Rahmen von Toponymverwendungen an. Dieser unterscheidet sich deutlich von dem von Markert und anderen, weil er sich eng an die klassischen Definitionen der Stilmittel anlehnt und keine eigenen Klassen wie *place-for-event* oder *place-for-people* verwendet.

Darüber hinaus sind auch Anwendungen im Bereich der *Toponymauflösung* denkbar, obwohl diese grundsätzlich unabhängig von der *Toponymcharakterisierung* durchgeführt werden kann. Dies ist besonders dann interessant, wenn im Text Grenzverschiebungs- bzw. Sprungtropen zum Einsatz kommen und sich die *Location*, also die Ortszuordnung eines Absatzes oder eines ganzen Textes ohne das Wissen über die Stilmittel nicht korrekt ermitteln lässt.

6.2 Ausblick

Die in der Evaluation und im vorherigen Abschnitt angesprochenen Fehler sowohl bei der Erkennung der Toponymverwendungen als auch ihrer Charakterisierung lassen viel Spielraum für weitere, auf dieser Arbeit aufbauende Untersuchungen und Lösungen. So könnte an der Infrastruktur der geonames.org-Anbindung zur Vermeidung vieler falscher Treffer beispielsweise eine wort- und regelbasierte Ausnahmeliste nach dem Vorbild der GATE-eigenen Toponymerkennung implementiert werden. Weiterhin sollten die in der vorgestellten Variante generisch eingebundenen Tagger ParZu und Stanford NER besser mit einem eigenen Plugin in GATE integriert werden, da die hiermit gefundenen Ergebnisse die Erkennungsraten stark verbessern bzw. einzelne Charakterisierungen erst möglich machen. Eine native Integration böte zudem die Chance, deren Ergebnisse besser mit denen der anderen, bereits existierenden GATE-Komponenten abzumischen. So unterstützt ParZu ähnlich wie der Durm German Lemmatizer unter anderem eine morphologische Satzanalyse und gibt zudem wie das TreeTagger-Plugin die POS-Informationen zurück. Eine bessere Integration könnte so vermeiden, dass verschiedene Features der gleichen `Token`-Annotation redundante oder sogar widersprüchliche Informationen zurückliefern.

Speziell beim ParZu-Tagger wäre zudem eine bessere Rückmeldung über den aktuellen Status der Analyse nötig. Der Grund ist, dass dieser selbst bei kleinen Texten zur Analyse teilweise mehrere Sekunden und bei größeren mehr als eine Minute benötigt. Darüber hinaus könnte der Nutzer ohne entsprechende Rückmeldung vermuten, dass die Prozessierung nicht mehr reagiert und deshalb abgebrochen werden muss. Ganz grundsätzlich wäre hier auch zu untersuchen, inwiefern sich die Performance des Taggers allgemein verbessern lässt.

Auch bei der Figurbestimmung lassen sich bereits mit existierenden Mitteln verschiedene Verbesserungen erreichen. So könnten beispielsweise Vorkommen der Sinnfigur *Epitheton Ornans* über eine bessere Integration der GermaNet-Datenbank erkannt werden, wenn das schmückende Beiwort als Relation zum Toponym in der Datenbank gepflegt ist. Allerdings besteht bei der Nutzung von GermaNet in der aktuell freigegebenen Version ähnlich wie bei ParZu ein großer Verbesserungsbedarf bei der Performance. Dadurch, dass vor jeder Nutzung zunächst eine große Menge an XML-Dateien eingelesen werden müssen, verzögert sich die weitere Prozessierung um mehrere Sekunden. Hier müssten die Daten in einer effizienteren Persistenz abgelegt werden, um

sie schneller einlesen zu können. Alternativ müssten die Autoren eine API freigeben, welche sich zum Auslesen der Daten wie der Editor GernEdiT direkt gegen eine bereits fertig eingerichtete PostgreSQL-Datenbank verbindet.

Weiterhin ließe sich gegebenenfalls bei der morphologischen Analyse durch die Integration des Stanford NLP eine Verbesserung bei der Erkennung der Kasus der einzelnen Satzteile erreichen. Wie das kurze Beispiel in Kapitel 3.1.3 zeigt, unterstützt dieser die Erkennung von deutlich mehr Phrasentypen als die von der Anwendungspipeline aktuell genutzten Komponenten und könnte so beispielsweise den MuNPEx ersetzen sowie darüber hinaus zusätzliche Informationen zur Analyse liefern.

Der Toponym Analyzer nutzt aktuell zudem nur eine kleine, beispielhafte Liste an Toponymen, die als Antonomasie, Allegorie oder Metapher verwendet werden können. Diese Liste müsste mit weiteren Elementen entsprechend komplettiert werden, um die Erkennungsraten dieser Stilmittel erhöhen zu können. Insgesamt wäre zudem zu prüfen, ob der Ansatz, bestimmte Stilmittel vorrangig mit Hilfe von Gazetteers zu finden, auf Dauer trägt, oder ob hier andere Technologien erfolgversprechender sind.

Weiterhin müsste untersucht werden, inwiefern sich die verschiedenen Herangehensweisen und Ergebnisse der Arbeiten zur Toponymcharakterisierung in Texten englischer Sprache, die in Kapitel 2.3.3 vorgestellt wurden, auf die Analyse von deutschen Texten übertragen lassen. In diesem Fall könnte die existierende Blackboard-Architektur des Toponym Analyzers beispielsweise um weitere Wissensquellen erweitert werden, die sich diese Daten zunutze machen.

Abseits der Verbesserungen an der aktuellen Infrastruktur könnte die *Charakterisierung von Toponymverwendungen* und darauf aufbauende Anwendungen von maschinellen Lernalgorithmen profitieren, wie sie auch bereits bei einigen verwandten Arbeiten zum Einsatz gekommen sind. Ist zum Beispiel einmal eine ausreichend große Anzahl an Texten mit Toponymverwendungen und den genutzten Stilmitteln annotiert, könnten diese Informationen dazu verwendet werden, andere Dokumente zu klassifizieren bzw. in Cluster aufzuteilen. GATE bietet hier mit einer eigenen Processing Resource bereits die entsprechende Infrastruktur[1] an. Die Autoren der Dokumentation weisen bereits in der Erklärung darauf hin, dass beispielsweise die Annotationen vom Typ `Token` mit ihren Features als Instanzen für den maschinellen Lernprozess etwa beim POS-Tagging verwendet werden können. Zur Toponymerkennung und -charakterisierung müssten dann analog die Annotationen vom Typ `Location` bzw. `Toponym` aus einem Referenzkorpus genutzt werden.

Wie bereits in der Motivation in Kapitel 1.1 erwähnt, spricht zudem vieles dafür, dass verschiedene Charakteristika unterschiedlich stark in bestimmten Textgenres eingesetzt werden. Wenn diese Vermutung zutrifft, könnte zum Beispiel anhand der Vorkommenshäufigkeiten bestimmter Verwendungsarten oder Stilmittel ein passendes Genre abgeleitet werden. Eine solche Herangehensweise wurde unter anderem bereits ähnlich bei der Untersuchung von Filmen gewählt [BOS13]. Hier wurden Rollentypen wie etwa der Hauptdarsteller oder der Antagonist nur anhand der Aktivitäten abgeleitet, die von den entsprechenden Schauspielern dargestellt werden. Dabei sortierte das zugrundeliegende Modell beispielsweise die Hauptpersonen der Filme *Iron Man*, *Die Bourne Identität* oder *The Dark Knight* auf Basis von Attributen wie *besser*, *schießen*,

[1] `http://gate.ac.uk/sale/tao/splitch18.html#chap:ml` - abgerufen am 22. April 2014

bestrafen oder *verhaften* in eine gemeinsame Gruppe ein. Übertragen auf die angestrebte Genreerkennung könnte ein passendes Modell eventuell viele Nachrichtentexte aufgrund des gehäuften Auftretens von Toponymen als Synekdochen zusammen in eine Gruppe einsortieren, während in einer anderen Gruppe mehrheitlich Reiseberichte wegen vieler Referenzen und vergleichsweise weniger Stilmittel vorkommen. Bei einer solchen Untersuchung könnte zudem ermittelt werden, ob wiederum andere Stilmittel wie beispielsweise die Alliteration überhaupt verstärkt in einem bestimmten Genre auftreten.

Literaturverzeichnis

[ABN12] Andogah, G. ; Bouma, G. ; Nerbonne, J.: Every document has a geographical scope. In: *Data & Knowledge Engineering* 81-82 (2012), November, 1–20. http://dx.doi.org/10.1016/j.datak.2012.07.002. – DOI 10.1016/j.datak.2012.07.002. – ISSN 0169023X

[AHSS04] Amitay, Einat ; Har'El, N ; Sivan, Ron ; Soffer, Aya: Web-a-where: geotagging web content. In: *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval* (2004), 273–280. http://dl.acm.org/citation.cfm?id=1009040. ISBN 1581138814

[Arb10] Arbeitskreis Terminologie der Sachverständigengruppe der Vereinten Nationen für geographische Namen (UNGEGN): *Deutsches Glossar zur toponymischen Terminologie*. http://141.74.33.52/stagn/Portals/0/100419_Glossar_3.pdf. Version: 2010. – abgerufen am 13. April 2014

[BGP12] Bordogna, Gloria ; Ghisalberti, Giorgio ; Psaila, Giuseppe: Geographic information retrieval: Modeling uncertainty of user's context. In: *Fuzzy Sets and Systems* 196 (2012), Juni, 105–124. http://dx.doi.org/10.1016/j.fss.2011.04.005. – DOI 10.1016/j.fss.2011.04.005. – ISSN 01650114

[BOB82] Belkin, N.J. ; Oddy, R.N. ; Brooks, H.M.: Ask for Information Retrieval: Part I. Background and Theory. In: *Journal of Documentation* 38 (1982), Nr. 2, 61–71. http://dx.doi.org/10.1108/eb026722. – DOI 10.1108/eb026722. – ISSN 0022–0418

[BOS13] Bamman, David ; O'Connor, Brendan ; Smith, Noah A.: Learning Latent Personas of Film Characters. In: *Proceedings pf the 51st Annual Meeting of the Association for Computational Linguistics* (2013). http://www.cs.cmu.edu/~nasmith/papers/bamman+oconnor+smith.acl13.pdf

[BR08] Buscaldi, Davide ; Rosso, Paulo: A conceptual density-based approach for the disambiguation of toponyms. In: *International Journal of Geographical Information Science* 22 (2008), März, Nr. 3, 301–313. http://dx.doi.org/10.1080/13658810701626251. – DOI 10.1080/13658810701626251. – ISSN 1365–8816

[Bru08] Brunner, Tobias: *Geographic Information Retrieval: Identifikation der geographischen Lage von Zeitungsartikeln'*, Universität Zürich, Diss., 2008. http://www.geo.uzh.ch/fileadmin/files/content/abteilungen/gis/research/msc_thesis/msc_tobias_brunner_2008.pdf. – abgerufen am 13. April 2014

[Bus11] Buscaldi, Davide: Approaches to disambiguating toponyms. In: *SIGSPATIAL Special* 3 (2011), Juli, Nr. 2, 16–19. http://dx.doi.org/10.1145/2047296.2047300. – DOI 10.1145/2047296.2047300. – ISSN 19467729

[CM02] Cunningham, Hamish ; Maynard, Diana: GATE: an architecture for development of robust HLT applications. In: *ACL '02 Proceedings of the 40th Annual Meeting on Association for Computational Linguistics* (2002). http://dl.acm.org/citation.cfm?id=1073112

[CMS10] Croft, W. B. ; Metzler, Donald ; Strohman, Trevor: *Search Engines - Information Retrieval in Practice*. Pearson Education, 2010. – 524 S. – ISBN 978–0–13–136489–9

[Coh60] Cohen, J.: A Coefficient of Agreement for Nominal Scales. In: *Educational and Psychological Measurement* 20 (1960), April, Nr. 1, 37–46. http://dx.doi.org/10.1177/001316446002000104. – DOI 10.1177/001316446002000104. – ISSN 0013–1644

[Ebe14] Eberhard Karls Universität Tübingen: *GermaNet - An Introduction*. http://www.sfs.uni-tuebingen.de/lsd/index.shtml. Version: 2014. – abgerufen am 13. April 2014

[Fot04] Foth, Killian A.: *Eine umfassende Constraint-Dependenz-Grammatik des Deutschen*. http://nats-www.informatik.uni-hamburg.de/pub/CDG/DeutschGrammar/doc.ps. Version: 2004. – abgerufen am 13. April 2014

[FP10] Faruqui, Manaal ; Padó, Sebastian: Training and evaluating a German named entity recognizer with semantic generalization. In: *Proc. of KONVENS* (2010). http://www.ims.uni-stuttgart.de/institut/mitarbeiter/schulte/publications/proceedings/konvens-10.pdf#page=130

[Gaw09] Gawryjolek, Jakub J.: *Automated Annotation and Visualization of Rhetorical Figures*, University of Waterloo, Ontario, Kanada, Diss., 2009. https://uwspace.uwaterloo.ca/bitstream/handle/10012/4426/Gawryjolek_Jakub.pdf. – abgerufen am 13. April 2014

[GHJV12] Gamma, Erich ; Helm, Richard ; Johnson, Ralph ; Vlissides, John: *Entwurfsmuster: Elemente wiederverwendbarer objektorientierter Software*. 6. Auflage. Addison-Wesley Verlag, 2012. – 504 S.

[HF97] Hamp, Birgit ; Feldweg, Helmut: GermaNet - A Lexical-Semantic Net for German. In: *Proceedings of ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications* (1997). http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.44.9354&rep=rep1&type=pdf

[HL07] Henrich, Andreas ; Luedecke, Volker: Characteristics of geographic information needs. In: *Proceedings of the 4th ACM workshop on Geographical information retrieval - GIR '07* (2007), 1. http://dx.doi.org/10.1145/1316948.1316950. – DOI 10.1145/1316948.1316950. ISBN 9781595938282

[HL08] Henrich, Andreas ; Lüdecke, Volker: Determining geographic representations for arbitrary concepts at query time. In: *Proceedings of the first international workshop on Location and the web - LOCWEB '08* (2008), 17–24. http://dx.doi.org/10.1145/1367798.1367802. – DOI 10.1145/1367798.1367802. ISBN 9781605581606

[JZR08] Jones, Rosie ; Zhang, WV ; Rey, Benjamin: Geographic intention and modification in web search. In: *International Journal of Geographical Information Science* 22 (2008), Nr. 3, 1–20. http://dx.doi.org/10.1080/13658810701626186. – DOI 10.1080/13658810701626186

[Kö5] Kürschner, Wilfried: *Grammatisches Kompendium.* 5. Auflage. Tübingen : Narr Francke Attempto Verlag, 2005. – 293 S.

[KAH+10] Kelly, Ashley R. ; Abbott, Nike a. ; Harris, Randy A. ; DiMarco, Chrysanne ; Cheriton, David R.: Toward an ontology of rhetorical figures. In: *Proceedings of the 28th ACM International Conference on Design of Communication - SIGDOC '10* (2010), 123. http://dx.doi.org/10.1145/1878450.1878471. – DOI 10.1145/1878450.1878471. ISBN 9781450304030

[KRS02] Kolmer, Lothar ; Rob-Santer, Carmen: *Studienbuch Rhetorik.* Verlag Ferdinand Schöningh, Paderborn, 2002. – 237 S. – ISBN 3–506–97017–8

[Lï0] Lüdecke, Volker: *Geographisches Information-Retrieval für deutsche Webseiten: Problemstellungen und Lösungsansätze im Rahmen des Szenarios einer Deutschland-Suchmaschine.*, Otto-Friedrich-Universität Bamberg, Deutschland, Diss., 2010

[Lei06] Leidner, Jochen L.: An evaluation dataset for the toponym resolution task. In: *Computers, Environment and Urban Systems* 30 (2006), Juli, Nr. 4, 400–417. http://dx.doi.org/10.1016/j.compenvurbsys.2005.07.003. – DOI 10.1016/j.compenvurbsys.2005.07.003. – ISSN 01989715

[Lei07] Leidner, Jochen L.: *Toponym resolution in text*, University of Edinburgh, Großbritannien, Diss., Dezember 2007. https://www.era.lib.ed.ac.uk/handle/1842/1849. – 124 S. – abgerufen am 13. April 2014

[LH08] Leveling, Johannes ; Hartrumpf, Sven: On metonymy recognition for geographic information retrieval. In: *International Journal of Geographical Information Science* 22 (2008), März, Nr. 3, 289–299. http://dx.doi.org/10.1080/13658810701626244. – DOI 10.1080/13658810701626244. – ISSN 1365–8816

[LL11] Leidner, JL ; Lieberman, MD: Detecting geographical references in the form of place names and associated spatial natural language. In: *SIGSPATIAL Special* (2011), 5–11. http://dl.acm.org/citation.cfm?id=2047298

[Mar97] Marcu, Daniel: The rhetorical parsing of natural language texts. In: *ACL '98 Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics* (1997), 96–103. http://dl.acm.org/citation.cfm?id=979630

[MH02] Markert, Katja ; Hahn, Udo: Understanding metonymies in discourse. In: *Artificial Intelligence* 135 (2002), Februar, Nr. 1-2, 145–198. http://dx.doi.org/10.1016/S0004-3702(01)00150-3. – DOI 10.1016/S0004–3702(01)00150–3. – ISSN 00043702

[Mic01] Michel, Georg: *Stilistische Textanalyse - Eine Einführung.* Frankfurt am Main : Europäischer Verlag der Wissenschaften, 2001. – 221 S.

[MN09] Markert, Katja ; Nissim, Malvina: Data and models for metonymy resolution. In: *Language Resources and Evaluation* 43 (2009), Februar, Nr. 2, 123–138. http://dx.doi.org/10.1007/s10579-009-9087-y. – DOI 10.1007/s10579–009–9087–y. – ISBN 1057900990

[Nii86] Nii, H P.: Blackboard Systems. Version: 1986. http://i.stanford.edu/pub/cstr/reports/cs/tr/86/1123/CS-TR-86-1123.pdf. 1986. – Forschungsbericht. – abgerufen am 13. April 2014

[NJMS12] Nastase, Vivi ; Judea, Alex ; Markert, Katja ; Strube, Michael: Local and global context for supervised and unsupervised metonymy resolution. In: *EMNLP-CoNLL '12 Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (2012), Nr. July, 183–193. http://dl.acm.org/citation.cfm?id=2390971

[NM03] Nissim, Malvina ; Markert, Katja: Syntactic features and word similarity for supervised metonymy resolution. In: *ACL '03 Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1* 2 (2003), Nr. July, 56–63. http://dl.acm.org/citation.cfm?id=1075104

[OR08] Overell, Simon ; Rüger, Stefan: Using co-occurrence models for placename disambiguation. In: *International Journal of Geographical Information Science* 22 (2008), März, Nr. 3, 265–287. http://dx.doi.org/10.1080/13658810701626236. – DOI 10.1080/13658810701626236. – ISSN 1365–8816

[PBP13] Presser, Brandon ; Bain, Carolyn ; Parnell, Fran: *Lonely Planet Reiseführer Island.* 3. deutsch. MAIRDUMONT, Ostfildern, 2013. – 404 S.

[PW05] Perera, Praharshana ; Witte, René: A Self-Learning Context-Aware Lemmatizer for German. In: *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing* (2005), Nr. October, 636–643. http://dl.acm.org/citation.cfm?id=1220655

[RM08] Rafferty, Anna N. ; Manning, Christopher D.: Parsing Three German Treebanks: Lexicalized and Unlexicalized Baselines. (2008)

[RW10] Radding, Lisa ; Western, John: What's in a Name? Linguistics, Geography, and Toponyms. In: *Geographical Review* 100 (2010), Juni, Nr. 3, 394–412. http://dx.doi.org/10.1111/j.1931-0846.2010.00043.x. – DOI 10.1111/j.1931–0846.2010.00043.x. – ISSN 00167428

[Sal68] Salton, Gerard: *Automatic Information Organization and Retrieval.* McGraw Hill Text, 1968. – ISBN 0070544859

[Sch94] SCHMID, Helmut: Probabilistic part-of-speech tagging using decision trees. In: *Proceedings of the International Conference on New Methods in Language Processing* (1994). http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/data/tree-tagger1.pdf. – abgerufen am 13. April 2014

[Sch95] SCHMID, Helmut: Improvements in part-of-speech tagging with an application to German. In: *In Proceedings of the ACL SIGDAT-Workshop* (1995). http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.52.2255

[Sch13] SCHÄTZLE, Christin: *Eine computerlinguistische Untersuchung des Genitivschwundes*, Universität Konstanz, Deutschland, Diss., 2013. http://ling.uni-konstanz.de/pages/home/schaetzle/publications/MA_Christin.pdf. – abgerufen am 13. April 2014

[SFH04] SCHMID, Helmut ; FITSCHEN, Arne ; HEID, Ulrich: SMOR: A German Computational Morphology Covering Derivation, Composition and Inflection. In: *LREC* (2004), Nr. 3, 4–7. http://lrec-conf.org/proceedings/lrec2004/pdf/468.pdf. – abgerufen am 13. April 2014

[SH11] SCHLIEDER, Christoph ; HENRICH, Andreas: Spatial grounding with vague place models. In: *SIGSPATIAL Special* 3 (2011), Juli, Nr. 2, 20–23. http://dx.doi.org/10.1145/2047296.2047301. – DOI 10.1145/2047296.2047301. – ISSN 19467729

[SK04] SANDERSON, Mark ; KOHLER, Janet: *Analyzing geographic queries*. http://www.geo.uzh.ch/~rsp/gir/abstracts/sanderson.pdf. Version: 2004. – abgerufen am 13. April 2014

[SLMR08] STOKES, Nicola ; LI, Yi ; MOFFAT, Alistair ; RONG, Jiawen: An empirical study of the effects of NLP components on Geographic IR performance. In: *International Journal of Geographical Information Science* 22 (2008), März, Nr. 3, 247–264. http://dx.doi.org/10.1080/13658810701626210. – DOI 10.1080/13658810701626210. – ISSN 1365–8816

[SSVW09] SENNRICH, Rico ; SCHNEIDER, Gerold ; VOLK, Martin ; WARIN, Martin: A new hybrid dependency parser for German. In: *Proc. of the German Society for Computational Linguistics and Language Technology 2009* (2009). https://files.ifi.uzh.ch/CL/volk/papers/Sennrich_Schneider_Volk_Warin_Pro3Gres_GSCL.pdf. – abgerufen am 13. April 2014

[STT99] SCHILLER, Anne ; TEUFEL, Simone ; THIELEN, C: *Guidelines für das Tagging deutscher Textcorpora mit STTS*. http://www.sfs.uni-tuebingen.de/resources/stts-1999.pdf. Version: 1999. – abgerufen am 13. April 2014

[WC08] WELCH, Michael J. ; CHO, Junghoo: Automatically identifying localizable queries. In: *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '08* (2008), 507. http://dx.doi.org/10.1145/1390334.1390421. – DOI 10.1145/1390334.1390421. ISBN 9781605581644

[YRL09] YI, Xing ; RAGHAVAN, Hema ; LEGGETTER, Chris: Discovering users' specific geo intention in web search. In: *Proceedings of the 18th international conference on*

World wide web - WWW '09 (2009), 481. http://dx.doi.org/10.1145/1526709.1526774. – DOI 10.1145/1526709.1526774. ISBN 9781605584874

Anhang

Texte des Referenzkorpus

Name	Quelle
atomkraftwerk	Internet - SPIEGEL ONLINE: http://www.spiegel.de/politik/ausland/ungarn-vergibt-milliardenschweren-atomauftrag-an-russland-a-943568.html - abgerufen am 13. April 2014
deutschlandlied	Erste Strophe des „Lieds der Deutschen“ von August Heinrich Hoffmann von Fallersleben
down-under	Internet - afp - Wolfsburger Allgemeine: http://www.waz-online.de/Nachrichten/Panorama/Uebersicht/Prinz-William-begeistert-in-Down-Under - abgerufen am 13. April 2014
figuren	Zu Testzwecken selbst zusammengestellter Text
fraenkisches-rom	Internet - Wikipedia: http://de.wikipedia.org/wiki/Fränkisches_Rom - abgerufen am 13. April 2014
gauck-schweiz	Internet - SPIEGEL ONLINE: http://www.spiegel.de/politik/ausland/zuwanderung-gauck-warnt-schweiz-vor-distanzierung-von-europa-a-962032.html - abgerufen am 13. April 2014
island	Lonely Planet Reiseführer Island, Seite 27
nato	Internet - tagesschau.de: http://www.tagesschau.de/ausland/nato-ukraine-georgien100.html - abgerufen am 13. April 2014
norwegen	Internet - MERIAN: http://www.merian.de/magazin/norwegen-bahnfahrt-von-oslo-nach-bergen.html - abgerufen am 13. April 2014
putin	Internet - DIE WELT: http://www.welt.de/politik/ausland/article126543003/Macht-Putin-Ernst-hat-die-Ukraine-keine-Chance.html - abgerufen am 13. April 2014

Name	**Quelle**
rumaenien-auto	Internet - DIE WELT: `http://www.welt.de/wirtschaft/article124211568/In-Rumaenien-liegt-die-Zukunft-des-deutschen-Autos.html` - abgerufen am 13. April 2014
tropen	Zu Testzwecken selbst zusammengestellter Text
uncle-sam	Internet - Neue Züricher Zeitung: `http://www.nzz.ch/aktuell/startseite/article8NM14-1.208282` - abgerufen am 13. April 2014
wiege-der-menschheit	Internet - DIE WELT: `http://www.welt.de/wissenschaft/article10571917/Wiege-der-Menschheit-womoeglich-nicht-Afrika.html` - abgerufen am 13. April 2014
wolgograd	Internet - SPIEGEL ONLINE: `http://www.spiegel.de/politik/ausland/russische-stadt-wolgograd-anschlaege-ueberschatten-olympia-a-941242.html` - abgerufen am 13. April 2014

Eidesstattliche Erklärung

Ich erkläre hiermit gemäß §16 der Prüfungsordnung für den Masterstudiengang VAWi, dass ich die vorliegende Masterarbeit mit dem Titel „Charakterisierung von Toponymverwendungen in Texten“ selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel und Quellen benutzt, Zitate kenntlich gemacht und die Arbeit noch keiner anderen Stelle zu Prüfungszwecken vorgelegt habe.

Walldorf, 05.05.2014

Ort, Datum | Sebastian Johannes Wolf