

Ruprecht-Karls-Universität Heidelberg
Institut für Informatik
Lehrstuhl Software Engineering

Bachelor-Arbeit

Entwicklung einer Plattform zur Evaluation
(Semi-)automatischer unüberwachter
domänenspezifischer
Terminologieextraktionsansätze aus
natürlichsprachigen Texten

Name: Christopher Jung
Matrikelnummer: 2743615
Betreuer: Prof. Dr. Barbara Paech, Thomas Quirchmayr

Eidesstatliche Erklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Sowohl inhaltlich als auch wörtlich entnommene Inhalte wurden als solche kenntlich gemacht.

Die Arbeit ist in gleicher oder vergleichbarer Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Heidelberg, den 9. Januar 2018

Christopher Jung

Abstract

Der Fokus dieser Arbeit ist die Evaluation bestehender unüberwachter, domänenspezifischer Termextraktionsansätze. Dafür wird eine Evaluationsplattform erstellt, die die relevantesten Ansätze miteinander vergleicht. Der theoretischen Teil dieser Arbeit beschäftigt sich mit der Analyse der existierenden Ansätze und gibt einen Einblick in die domänenspezifische Termextraktion. Eine selbst implementierte Entwicklungsplattform, die die relevantesten Ansätze vergleicht, wird ebenfalls beschrieben. Die Ergebnisse dieser Arbeit werden im Bezug auf Precision, Recall und F1-Score bewertet und diskutiert. Im letzten Kapitel werden die Ergebnisse zusammengefasst und es wird ein kurzer Ausblick gegeben.

This work is focused on an evaluation of existing approaches for unsupervised domain specific term extraction. To this end, an evaluation platform is created to compare the most relevant approaches. The theoretical part deals with the analysis of existing approaches and gives an overview of domain specific term extraction. A self implemented evaluation platform, comparing the most relevant approaches, will also be described. The results will be discussed and evaluated in terms of precision, recall and F1-score. In the last chapter the results will be summarized and a short outlook will be given.

Inhaltsverzeichnis

Abbildungsverzeichnis	1
Abkürzungsverzeichnis	2
1 Einleitung	3
1.1 Motivation	3
1.2 Aufgabenstellung und Zielsetzung	3
1.3 Aufbau der Arbeit	4
2 Grundlagen	5
2.1 HITS-Algorithmus	5
2.2 Part-of-speech-Tagger	7
2.3 Shallow Semantic Relations	8
2.4 n-Gramm	8
2.5 Kontrastkorpus	8
2.6 Stanford Parser	9
2.7 Precision, Recall und F1-Score	9
3 Literaturrecherche	10
3.1 Forschungsfragen	10
3.2 Methodik der Literaturrecherche	10
3.2.1 Suchterm-basierte Literaturrecherche	11
3.2.2 Snowballing	11
3.3 Auswahlkriterien zur Bestimmung relevanter Ansätze	11
3.4 Durchführung der Literaturrecherche	13
3.4.1 Herleitung des Suchterms	13
3.4.2 Durchführung des Forward- und Backward-Snowballing	16
3.5 Ergebnisse der Literaturrecherche	17
3.5.1 Beantwortung der Forschungsfragen	17
3.5.2 Beschreibung relevanter Ansätze	20

4	Implementierung der Evaluationsplattform	24
4.1	Allgemeine Implementierungsaspekte	24
4.1.1	Entwicklungsumgebung	24
4.1.2	Natural Language Parser	24
4.1.3	Architektur Übersicht	25
4.1.4	Bedienung Übersicht	26
4.2	Implementierung des Ansatzes von Balachandran et al.	27
4.2.1	Notwendige Anpassungen	27
4.2.2	Architektur	28
4.2.3	Bedienung	29
4.2.4	Ausgabe	30
4.3	Implementierung des Ansatzes von Venu et al.	31
4.3.1	Notwendige Anpassungen	31
4.3.2	Architektur	31
4.3.3	Bedienung	32
4.3.4	Ausgabe	34
4.4	Implementierung des Evaluations-Programms	35
4.4.1	Architektur	35
4.4.2	Bedienung	35
4.4.3	Ausgabe	36
5	Evaluation	38
5.1	Evaluation von Balachandran et al.	38
5.1.1	Anwendbarkeit und Probleme	38
5.1.2	Ergebnisse der Evaluation	39
5.1.3	Verbesserungsmöglichkeiten	40
5.2	Evaluation von Venu et al.	40
5.2.1	Anwendbarkeit und Probleme	41
5.2.2	Ergebnisse	41
5.2.3	Verbesserungsmöglichkeiten	42
5.3	Allgemeine Verbesserungsmöglichkeiten	42
5.4	Fazit der Evaluation	43
6	Zusammenfassung und Ausblick	46
6.1	Zusammenfassung	46
6.2	Ausblick	46
	Literaturverzeichnis	48

Abbildungsverzeichnis

3.1	Suchbegriffe und Ergebnisse der Literaturrecherche	15
3.2	Ergebnisse des Snowballings	16
4.1	Implementierte Klassen	25
4.2	Übersicht zur Bedienung der Evaluationsplattform	26
4.3	Architektur des Algorithmus von Balachandran et al.	28
4.4	Reihenfolge der Benutzereingaben für die Implementierung von Balachandran et al.	29
4.5	Sortierte Ausgabedatei des Ansatzes von Balachandran et al.	30
4.6	Sortierte Ausgabedatei des Ansatzes von Balachandran et al. inklusive Bewertung	30
4.7	Architektur des Algorithmus von Venu et al.	32
4.8	Reihenfolge der Benutzereingaben für die Implementierung von Venu et al.	33
4.9	Sortierte Ausgabedatei von SSR und Nomen des Ansatzes von Venu et al.	34
4.10	Architektur des Evaluations-Programms	35
4.11	Reihenfolge der Benutzereingaben für die Implementierung der Evaluation	36
4.12	Ausgabedatei der Evaluation	37
5.1	Ergebnisse der Evaluation von Balachandran et al.	40
5.2	Ergebnisse der Evaluation der SSR und Nomen von Venu et al.	42
5.3	Vergleich der Precision von Venu et al. und Balachandran et al.	44

Abkürzungsverzeichnis

DC	Domain Consensus
DP	Domain Pertinence
HITS	Hyperlink-Induced Topic Search
IDF	Inverse Document Frequency
LC	Lexial Cohesion
NLP	Natural Language Processing
ML	Machine Learning
POS	Part-of-Speech
SR	Structural Relevance
SSR	Shallow Semantic Relations
TF	Term Frequency

1 Einleitung

In diesem Kapitel wird die Motivation, Aufgabenstellung und Zielsetzung sowie der Aufbau dieser Bachelor-Arbeit beschrieben.

1.1 Motivation

Die zunehmende Digitalisierung unserer Gesellschaft führt zu einer rapide steigenden Zahl von textbasierten Dokumenten. Es ist kaum noch möglich diese Informationsflut manuell auszuwerten und (kontext-)relevante Informationen in diesen Texten zu identifizieren.

Abhilfe versucht diesbezüglich jedoch das Natural Language Processing (NLP) zu schaffen. Ein Teilbereich des NLP beschäftigt sich dabei mit natürlichsprachigen Texten und deren Auswertung mit Hilfe des Computers, um so Informationen konsolidiert zur Verfügung zu stellen. Dabei werden relevante Informationen durch unterschiedliche Ansätze extrahiert. [2]

Die Ansätze verwenden meist manuell erstellte Trainingsdaten, um relevante Informationen zu identifizieren. Das Erstellen solcher Trainingsdatensätze ist jedoch sehr zeitintensiv. [26]

Daher ist es sinnvoll den Bereich der (semi-)automatisierten Termextraktionsansätze weiter zu verfolgen um den manuelle Aufwand weiter zu minimieren.

1.2 Aufgabenstellung und Zielsetzung

Im Rahmen dieser Bachelor-Arbeit soll eine Plattform zur Evaluation (semi-)automatischer unüberwachter domänenspezifischer Terminologieextraktionsansätze aus natürlichsprachigen Texten erstellt werden.

Hierzu soll eine ausführliche Literaturrecherche vollzogen, vorhandene Extraktionsansätze gefunden und die gegebenen Forschungsfragen beantwortet werden. Geeignete Extraktionsansätze sollen zudem in einer Evaluationsplattform implementiert und ein gegebener Text damit analysiert werden. Bei dem Text handelt es sich um ein Benutzerhandbuch eines CRM Systems, das für ein Unternehmen analysiert werden soll. Die

Ergebnisse sollen anschließend mit Hilfe eines ebenfalls gegebenen Goldstandards bewertet und interpretiert werden. Ziel ist es dadurch dem Industriepartner die Möglichkeit zu geben auf Basis eines existierenden Benutzerhandbuchs feature relevante Informationen aufzudecken.

1.3 Aufbau der Arbeit

Das erste Kapitel stellt eine Einleitung in das Thema der domänenspezifischen Termextraktion dar. Dabei wird die Motivation sowie die Zielsetzung dieser Bachelor-Arbeit beschrieben. Im zweiten Kapitel werden die Grundlagen, die zum Verständnis dieser Bachelor-Arbeit notwendig sind, erläutert. Kapitel drei beschreibt die Literaturrecherche, beantwortet die Forschungsfragen und stellt die verwendeten Ansätze im Detail vor. In Kapitel vier wird die implementierte Evaluationsplattform erläutert. Neben der Architektur wird auch die Bedienung und Ausgabe beschrieben. Das fünfte Kapitel beschäftigt sich mit der Evaluation der zuvor beschriebenen Ausgabe. Hier werden die Ergebnisse bezüglich ihrer Qualität bewertet und verglichen sowie ein Fazit abgegeben. Das letzte Kapitel liefert eine kurze Zusammenfassung und gibt einen Ausblick.

2 Grundlagen

Dieses Kapitel beschäftigt sich mit den Grundlagen zum Thema dieser Bachelor-Arbeit. Es liefert einen ersten Einblick in die domänenspezifische Termextraktion und stellt, die im späteren verwendeten Ansätze, Methoden und Begrifflichkeiten vor.

2.1 HITS-Algorithmus

Der *HITS-Algorithmus* ist ein graphbasiertes Ranking-Ansatz, der seinen Ursprung in der Beurteilung von Webseiten findet. Der Algorithmus wird jedoch, in angepasster Form, auch bei graphbasierten Methoden der Termextraktion verwendet. Dabei dient er der Klassifizierung relevanter Terme. In seiner Grundform bewertet der *HITS-Algorithmus* jede Webseite nach den Kategorien Hubs und Authorities. Hubs sind dabei Seiten, die auf viele wichtige andere Seiten zeigen, wohingegen Authorities Seiten sind, die besonders häufig verlinkt werden. In einem Graph können somit Hubs als Knoten bezeichnet werden, die besonders viele abgehende Verlinkungen aufweisen und Authorities als Knoten, die besonders viele eingehende Verlinkungen besitzen. [12]

Berechnung der Hubs und Authorities Um die Hubs und Authorities eines Graphen zu identifizieren, wird dieser zunächst in eine Adjazenzmatrix A überführt. Aus der Adjazenzmatrix A wird dann in die transponierte Adjazenzmatrix A^T berechnet. Um die Authorities zu bestimmen, wird der Authority-Vektor a_i berechnet. Dieser ergibt sich aus der Multiplikation der transponierten Adjazenzmatrix A^T mit dem Hub-Vektor h_{i-1} .

$$a_i = A^T \cdot h_{i-1}$$

Der Hub-Vektoren h_i wird aus dem berechneten Authority-Vektor und der regulären Adjazenzmatrix berechnet.

$$h_i = A \cdot a_i$$

Zum Start der Berechnungen wird der Hub-Vektor h_0 initial auf 1 gesetzt. Nach der Berechnung der Vektoren, werden diese normiert und der Algorithmus bis zur Konvergenz fortgeführt. Aus den berechneten Vektoren lassen sich die geeignetsten Hubs und Au-

thorities ablesen. Hub-Knoten besitzen einen hohen Wert im Hub-Vektor, Authorities besitzen einen hohen Wert im Authority-Vektor. [12]

Beispiel: HITS-Algorithmus

1. Graph in Adjazenzmatrix A überführen:

$$A = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

2. Transponierte Adjazenzmatrix A^T berechnet:

$$A^T = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix}$$

3. Authority-Vektor a_1 mit initialem Hub-Vektor berechnen:

$$a_1 = A^T \cdot h_0 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix}$$

4. Hub-Vektor h_1 berechnen:

$$h_1 = A \cdot a_1 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \\ 0 \end{pmatrix}$$

5. Vektoren normieren:

$$a_1 = \frac{a_1}{|a_1|} = \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix} \cdot \frac{1}{\sqrt{0^2 + 0^2 + 2^2}} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

$$h_1 = \frac{h_1}{|h_1|} = \begin{pmatrix} 2 \\ 2 \\ 0 \end{pmatrix} \cdot \frac{1}{\sqrt{2^2 + 2^2 + 0^2}} = \begin{pmatrix} 0,707 \\ 0,707 \\ 0 \end{pmatrix}$$

6. Algorithmus iterativ bis zur Konvergenz fortführen

$$a_2 = \left| \begin{pmatrix} 0 \\ 0 \\ 1.414 \end{pmatrix} \right| = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

$$h_2 = \left| \begin{pmatrix} 1.414 \\ 1.414 \\ 0 \end{pmatrix} \right| = \begin{pmatrix} 0,707 \\ 0,707 \\ 0 \end{pmatrix}$$

7. Bereits im Iterationsschritt $i = 2$ ist keine Änderung bei den Vektoren mehr feststellbar. Deshalb sind die Knoten 1 und 2 mit einem Hub-Wert von 0,707 die geeignetsten Hubs in diesem Graphen. Der Knoten 3 ist mit einem Authority-Wert von 1 der einzige Authority-Knoten.

2.2 Part-of-speech-Tagger

Ein Part-of-speech-Tagger oder kurz POS-Tagger ist ein Programm, das jedem Wort in einem gegebenen Text eine Wortart (englisch. *part of speech*) automatisch zuordnet. Die dadurch gewonnenen Informationen werden im Bereich der Termextraktion, vor allem von linguistischen Algorithmen, verwendet. Dabei können beispielsweise sinnvolle Wortartkombinationen erkannt und als mögliche relevante Terme extrahiert werden. [16]

Einer der bekanntesten und zugleich frei verfügbaren POS-Tagger ist der Stanford Parser [16], der an der Stanford University entwickelt wurde. Dieser POS-Tagger wird im Rahmen dieser Bachelor-Arbeit auch bei der Umsetzung der Evaluationsplattform eine wichtige Rolle spielen.

Beispiel: Part-of-speech-Tagger

Folgender Satz wird dem Stanford Parser übergeben:

„In some cases, it is configurable in the Admin Tool if an error or a warning should be displayed.“

Der POS-Tagger liefert folgendes Ergebnis:

„In/IN some/DT cases/NNS ,/, it/PRP is/VBZ configurable/JJ in/IN the/DT Admin/NNP Tool/NNP if/IN an/DT error/NN or/CC a/DT warning/NN should/MD be/VB displayed/VBN ./.“

Die Wortarten werden als Tags direkt hinter dem Wort angegeben. Der Tag *„/NNS“* steht beispielsweise für ein Nomen im Plural, wohingegen der Tag *„/NN“* für ein Nomen

im Singular steht. Mit linguistischen Regeln, die die Kombination bestimmter Wortarten festlegen, könnten nun passende Terme extrahiert werden. Eine Regel die beispielsweise alle Terme extrahiert, die eine Kombination aus zwei hintereinander folgenden Nomen darstellen, würde aus obigem Satz den Term „*Admin Tool*“ extrahieren.

2.3 Shallow Semantic Relations

Als Shallow Semantic Relations (SSR) werden einfache semantische Beziehungen zwischen zwei oder mehreren Termen innerhalb eines Satzes bezeichnet. SSR spielen im Natural Language Parsing eine bedeutende Rolle um Informationen über den Text, wie z.B. Abhängigkeiten oder Synonyme zu erlangen. Mit einem Natural Language Parser können SSR identifiziert werden. [10]

Beispiel: SSR Parser

Ein SSR Parser liefert aus dem „Samsung has a battery“ die Beziehung „nnMod:samsungt _battery“. Dabei steht „nnMod“ für die Art der SSR, hier Nomen-Modifizierer und „samsung“ und „battery“ für die verknüpften Konzepte. [19]

2.4 n-Gramm

Als n-Gramm werden Terme bezeichnet, die aus n Einzeltermen bestehen. Für n können beliebige ganze positive Zahlen eingesetzt werden. [4]

Die für diese Bachelor-Arbeit notwendigen n-Gramme, sind das Monogramm (1-Gramm), das Bigramm (2-Gramm) und das Trigramm (3-Gramm).

Beispiel: n-Gramm

Der Term *admin* besteht aus einem Term und wird daher als Monogramm bezeichnet. Der Term *admin tool* besteht aus zwei Einzeltermen und wird daher als Bigramm bezeichnet.

Der Term *admin tool box* besteht aus drei Einzeltermen und wird daher als Trigramm bezeichnet.

2.5 Kontrastkorpus

Eine Ansammlung von Texten jeglicher Form wird auch Korpus genannt [23]. Stammen alle Texte eines Korpus aus der selben Domäne, so lässt sich dieser auch als domänen-

spezifischer Korpus bezeichnen [5]. Als *Kontrastkorpora* werden nun domänenspezifische Korpora bezeichnet, deren Domäne ungleich der behandelten Domäne ist. [3].

Beispiel:Kontrastkorpus

Ist die aktuell behandelte Domäne die Informatik, werden domänenspezifische Korpora aus den Domänen Bio-Medizin, Cricket, Wirtschaft und so weiter als *Kontrastkorpora* bezeichnet.

2.6 Stanford Parser

Der *Stanford Parser* ist ein, an der Stanford Universität entwickelter, Natural Language Parser. Er ist in der Lage als POS-Tagger zu fungieren, Worte zu lemmatisieren und Termabhängigkeiten zu identifizieren, die für die Extraktion von SSR notwendig sind. Der Parser ist neben einer JAVA-Bibliothek auch in vielen anderen Programmiersprachen frei verfügbar. [16]

Zudem ist der *Stanford Parser* ausführlich dokumentiert [6]. Dadurch ist der *Stanford Parser* der ideale Natural Language Parser für diese Bachelor-Arbeit.

2.7 Precision, Recall und F1-Score

Precision, Recall und F1-Score sind Klassifikatoren zur Bewertung der Treffermenge einer Recherche. Im Natural Language Processing werden diese Metriken zur Bewertung der identifizierten Texte genutzt. Die Berechnung der Klassifikatoren werden aus den tatsächlich korrekt erkannten Treffern (tp = true positives), den tatsächlich negativ erkannten Treffern (tn = true negatives), den falsch positiv erkannten Treffern (fp = false positive) und den falsch negativ erkannten Treffern (fn = false negativ) berechnet. [20]

$$Precision = \frac{tp}{tp + fp} \quad (2.1)$$

$$Recall = \frac{tp}{tp + fn} \quad (2.2)$$

$$F1 - Score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (2.3)$$

3 Literaturrecherche

In diesem Kapitel wird die Literaturrecherche dieser Bachelor-Arbeit beschrieben. Zunächst werden dabei die Forschungsfragen erläutert, die im Rahmen der Literaturrecherche beantwortet werden sollen. Darauf folgend wird die Methodik und die Durchführung der Literaturrecherche erläutert. Die daraus gewonnen Ergebnisse werden anschließend zur Beantwortung der Forschungsfragen verwendet sowie für die Evaluationsplattform relevante Ansätze im Detail vorgestellt.

3.1 Forschungsfragen

Im Rahmen der Literaturrecherche im Kontext (semi-) automatischer, unüberwachter, domänenspezifischer Terminologieextraktion sollen die folgenden Forschungsfragen beantwortet werden:

RQ1 Welche (semi-)automatisierten, unüberwachten, domänenspezifischen Terminologieextraktionsansätze gibt es?

RQ1.1 In welcher Hinsicht unterscheiden sich die Algorithmen bezüglich verwendeter NLP und ML-Methoden sowie Anwendbarkeit in unterschiedlichen Domänen?

RQ1.2 Welche Vor- beziehungsweise Nachteile haben die Ansätze?

Diese Forschungsfragen werden im Abschnitt 3.5.1 dieses Literaturkapitels beantwortet.

3.2 Methodik der Literaturrecherche

Die Literaturrecherche dieser Bachelor-Arbeit verwendet sowohl Suchterme, als auch das sog. Snowballing zur Auswahl relevanter Publikationen. Beide Methoden werden im Nachfolgenden erläutert.

3.2.1 Suchterm-basierte Literaturrecherche

Bei der Suchterm-basierten Literaturrecherche werden relevante Publikationen mittels Suchterm in wissenschaftlichen Online-Literaturdatenbanken gesucht. Der Suchterm ergibt sich dabei aus der Aufgabenstellung der Bachelor-Arbeit sowie den Forschungsfragen und wird iterativ auf die Ergebnisse angepasst. Zur Klassifizierung in relevant und nicht relevant werden die gefundenen Werke mit zuvor bestimmten Auswahlkriterien abgeglichen. Die Auswahlkriterien werden, wie der Suchterm, aus der Aufgabenstellung dieser Bachelor-Arbeit abgeleitet. Mit Synonymen oder Akronymen kann der Suchterm zudem ergänzt werden, sodass zusätzliche relevante Publikationen gefunden werden. Um eine hohe Trefferquote mit dem Suchterm zu erreichen, können weitere Filterkriterien genutzt werden. Die Online-Literaturdatenbanken IEEE und ACM bieten z.B. die Möglichkeit nur den Abstract einer Publikation zu durchsuchen. Dadurch wird der zu durchsuchende Bereich eingegrenzt und die Ergebnisse liefern eine höhere Relevanz zum Suchterm. Die Herleitung des Suchterms dieser Bachelor-Arbeit wird im Abschnitt 3.4.1 detailliert beschrieben.

3.2.2 Snowballing

Basierend auf einer Startmenge relevanter Publikationen wird das sogenannte Forward- und Backward-Snowballing angewandt. Die relevanten Publikationen stammen dabei aus einer Suchterm-basierten Literaturrecherche.

Beim Forward-Snowballing werden Publikationen untersucht, die die relevante Ausgangspublikation referenzieren. Das Backward-Snowballing funktioniert genau umgekehrt. Es werden jene Publikationen auf Relevanz hin untersucht, die von einer relevanten Publikation aus referenziert werden. Zeitlich betrachtet sind damit durch Forward-Snowballing gefundene Publikationen aktueller als die Ausgangspublikation, wohingegen Publikationen, die mittels Backward-Snowballing gefunden werden, älter sind als die Ausgangspublikation. [25] Gerade deshalb ist bei sehr aktuellen Publikationen ein Forward-Snowballing daher meist nicht zielführend.

3.3 Auswahlkriterien zur Bestimmung relevanter Ansätze

Um aus der Vielzahl der existierenden Termextraktionsansätze geeignete Ansätze zu ermitteln, werden eindeutige Kriterien festgelegt. Nur Termextraktionsansätze, die diese Kriterien erfüllen, werden für die Evaluationsplattform dieser Arbeit in Betracht gezo-

gen.

Kriterium 1: Automatische domänenspezifische Termextraktion

Um das Kernforschungsgebiet dieser Arbeit im Fokus zu behalten, müssen die Termextraktionsansätze ohne manuellen Aufwand domänenspezifische Terme extrahieren können. Dies bedeutet, dass die Termextraktion unüberwacht und ohne manuelle Hilfe automatisch ausgeführt wird.

Kriterium 2: Anwendbarkeit auf Ziel-Domäne

Es muss gewährleistet sein, dass die Ansätze mit geringem Anpassungsaufwand in der Ziel-Domäne der Evaluationsdaten dieser Bachelor-Arbeit verwendet werden kann. Demnach muss die Termextraktion in der Domäne der Informatik anwendbar sein. Ansätze die Terme dieser Domäne nicht extrahieren oder nicht darauf angepasst werden können, werden daher nicht weiter berücksichtigt.

Kriterium 3: Precision, Recall, F1-Score & Vergleich

Die Term-Extraktionsansätze müssen sich in einem wissenschaftlichen Test gegenüber den zum Vergleich stehenden Ansätzen bezüglich ihrer Qualität durchsetzen. Als Grundlage der Bewertung dient dabei die in der jeweiligen Publikation vorgestellten Evaluation. Die Evaluation muss zudem einen ausreichenden Vergleich zu bestehenden Ansätzen liefern. Um einen eindeutigen Vergleich zu ermöglichen werden Precision, Recall und F1-Score verwendet.

Kriterium 4: Umsetzbarkeit & Verfügbarkeit

Nur Ansätze, deren programmiertechnische Umsetzung in der Evaluationsplattform als möglich erscheinen, werden in Betracht gezogen. Ansätze, die auf nicht zurückgreifbare Datenbanken zur Bestimmung der domänenspezifischen Terme zurückgreifen, werden ebenfalls ignoriert.

Kriterium 5: Unterschiedliche Ansätze

Um ein breites Spektrum und einen sinnvollen Vergleich der existierenden domänenspezifischen Extraktionsansätze zu liefern, sollen in dieser Bachelor-Arbeit nur sich unterscheidende Ansätze ausgewählt werden. Existieren zwei oder mehrer Ansätze die den Kriterien eins bis vier genügen, sich aber in ihrer Ansatz nicht ausreichend unterscheiden, wird der Ansatz mit der besseren Bewertung bezüglich Precision, Recall und F1-Score gewählt.

3.4 Durchführung der Literaturrecherche

Die Literaturrecherche wird in zwei Schritten ausgeführt. Im ersten Schritt werden relevante Publikationen über einen Suchterm ermittelt. Anschließend wird auf Basis der relevanten Publikationen durch Forward- und Backward-Snowballing nach weiteren relevanten Werken gesucht.

3.4.1 Herleitung des Suchterms

Für die Literaturrecherche stehen die Plattformen IEEE, ACM und Google Scholar zur Verfügung. Da Google Scholar kein eigenes Literatur-Archiv besitzt, sondern die Suche lediglich auf anderen Archiven (z.B: IEE, ACM, Researchgate, etc) ausführt, führen Suchergebnisse häufig auf nicht verfügbare oder zum Teil kostenpflichtige Inhalte. Google Scholar wird daher lediglich zur Einarbeitung in das Thema verwendet. Die weiterführende Recherche wird auf den Plattformen IEEE und ACM ausgeführt. Um Qualität und vor allem Korrektheit zu gewährleisten, werden ausschließlich vertrauenswürdige Literaturquellen verwendet. Die Ergebnisse der Literaturrecherche sind in Abbildung 3.1 zu finden. Darin wird der Suchort, die Suchanfrage, die Anzahl der vom Suchterm gefundenen Ergebnisse (# Erg.), die für diese Bachelor-Arbeit relevanten Ergebnisse (#relevante Erg.), die relevanten Ergebnisse die bisher noch nicht bekannt waren (# neue Erg.) und die daraus verwendeten Treffern (verwendete Treffer) aufgelistet. Um die Übersichtlichkeit der Abbildung zu wahren wird bei einer größeren Anzahl an verwendeten Treffern auf den Namen der Autoren verzichtet. Bei einer zu hohen Trefferanzahl wird die Auswertung der Ergebnisse übersprungen und der Suchterm weiter angepasst. Die Herleitung des Suchterms wird in Nachfolgendem beschrieben.

Um sich einen Überblick zu dem Thema Termextraktion sowie die notwendigen Grundlagen zu verschaffen, werden zunächst mit Hilfe von Google Scholar und dem Suchterm „term extraction“ entsprechende Informationen erlangt. Eine ausführliche und übersichtliche Erklärung zu dieser Thematik liefert die Publikation „NLP Techniques for Term Extraction and Ontology Population“ von Maynard et al. [17]. Durch den weit gefassten Suchterm werden jedoch sowohl Methoden des unüberwachten als auch des überwachten Lernens gefunden. Die Anzahl der relevanten Treffer ist mit dem Suchterm „term extraction“ nicht auswertbar, es wird lediglich die zum Überblick dienliche Publikation von Maynard et al. in den ersten 30 Treffern gefunden. Der Suchterm dahingehend angepasst, dass nur Begriffe innerhalb des Abstracts beachtet werden. So wird sichergestellt, dass die Ergebnisse Relevanz zu dem Thema aufwiesen.

Der Fokus dieser Arbeit liegt auf (semi-)automatischen domänenspezifischen Termextraktionsansätzen, der Suchterm wird deshalb um den Term „automatic“ erweitert. Der

domänenspezifischen Zusammenhang wird in den gefunden Publikationen mit der Formulierung „domain specific“ als auch „from domain corpora“ ausgedrückt. Daher wird der Suchterm um den Begriff „domain“ erweitert. Somit werden beide oben genannten Beispiele in der Suche eingeschlossen.

Um überwachte, beziehungsweise nicht automatische Methoden auszuschließen, muss der Suchterm weiter spezifiziert werden. Der Begriff „automatic“ wird zu dem Suchterm hinzugefügt, sodass der Suchterm „automatic domain term extraction“ entsteht. Dieser führt zu einer deutlichen Steigerung der relevanten Suchergebnisse. Mit diesem Suchterm können Quellen gefunden werden, die als Basis dieser Bachelor-Arbeit dienen. (Siehe Abbildung 3.1)

Die Ergebnisse dieses Suchterms sind noch stark von der Wortwahl des Autors abhängig. Werden zur Formulierung des gleichen Sachverhaltes andere Worte benutzt, werden diese durch den Suchterm nicht gefunden. Es wird daher im Folgenden sichergestellt, dass Synonyme und andere Formulierungsmöglichkeiten im Suchterm inkludiert sind. Nur dadurch kann sichergestellt werden, dass der Suchterm die Mehrzahl der relevanten Dokumente findet. Im ersten Schritt werden dafür die Begriffe des Suchterms, falls sinnvoll, auf den Wortstamm zurückgeführt. Aus „extraction“ wird „extract“, da dadurch die möglichen Flexionsendungen „extracting“, „extracted“ und „extracts“ gefunden werden. Der Begriff „automatic“ lässt sich nicht sinnvoll auf einen Wortstamm zurückführen, die Flexionsendung „automated“ soll jedoch beachtet werden. Der Begriff „automated“ wird daher zum Suchterm hinzugefügt, sodass sowohl „automatic“ als auch „automated“ gefunden wird. Der Suchterm sieht daher wie folgt aus: „(automated automatic) domain term extract“ Eine Klammer um zwei Suchbegriffe bedeutet, dass die Ergebnisse einen der beiden Suchbegriffe beinhalten müssen.

Im nächsten Schritt werden Synonyme und mögliche Formulierungen einbezogen. Im Gebiet des Natural Language Processing werden automatische („automatic“) Extraktionsmethoden auch den unüberwachten („unsupervised“) Extraktionsmethoden als Übergriff zugeordnet. Der Suchterm wird daher so erweitert, dass Ergebnisse eine der beiden Formulierungen beinhalten können. Anstelle von „term“ wird im Text der gefundenen relevanten Werke häufig der Begriff „concept“ und dahingehend auch die „concept extraction“ verwendet, weshalb der Begriff „concept“ als Synonym im Suchterm aufgenommen wird. Die Literaturrecherche zeigt, dass die Methode der Termextraktion die Grundlage des weit verbreiteten „Ontology learning“ ist. Da die Termextraktion eine kleinere Rolle in dieser Thematik spielt, jedoch immer notwendig ist, wird der Suchterm um den Begriff „ontology learning“ und „ontology extract“ erweitert. Die durch die Literaturrecherche gefundenen Werke werden im Abschnitt 3.5 vorgestellt und auf die Auswahlkriterien aus 3.3 geprüft.

Suchort	Suchanfrage	# Erg.	#relevante Erg.	#neue Erg.	verwendete Treffer
Google Scholar	term extraction	2860000	* (1)	(1)	Maynard et al. [17]
ACM	term AND extraction	2463	*	0	
ACM	domain AND term AND extraction	526	*	0	
ACM	term AND extraction ¹	1925	*	0	
ACM	domain AND term AND extraction ¹	309	*	0	
ACM	automatic AND domain AND term AND extraction ¹	102	5	5	[9], [14], [15], [18], [22]
IEEE	automatic AND domain AND term AND extraction ¹	146	3	1	Balachandran et al. [3]
ACM	automatic AND domain AND term AND extract ¹	102	5	0	
IEEE	automatic AND domain AND term AND extract ¹	167	3	0	
ACM	(automatic OR automated) AND domain AND term AND extract ¹	123	6	1	Venu et al. [24]
ACM	(automatic OR automated OR unsupervised) AND domain AND (term OR concept) AND extract ¹	217	6	0	
ACM	(automatic OR automated OR unsupervised) AND domain AND (term OR concept OR ontology) AND extract ¹	263	7	1	Drymonas et al. [7]
ACM	(automatic OR automated OR unsupervised) AND domain AND (((term OR concept OR ontology) and AND extract) OR „ontology learning”)) ¹	272	8	1	Wong et al. [26]

¹Suchbereich auf Abstract beschränkt. *Nicht auswertbar, Anzahl an Treffer zu hoch.

Suchdatum: September 2017

Abbildung 3.1: Suchbegriffe und Ergebnisse der Literaturrecherche

3.4.2 Durchführung des Forward- und Backward-Snowballing

Basierend auf den Ergebnissen aus 3.4.1 wird zunächst mittels Backward-Snowballing nach weiteren relevanten Publikationen gesucht. Bei der Suche werden relevante Werke gefunden, diese sind jedoch schon im entsprechenden Ausgangswerk vorgestellt. Zudem werden alle durch das Backward-Snowballing gefunden Ansätze in der Evaluation des Ausgangswerks mit einer geringen Precision als der vorgestellte Ansatz klassifiziert und verstoßen damit gegen das dritte Kriterium. Damit werden die durch Backward-Snowballing gefundenen Werke nicht in der Evaluationsplattform berücksichtigt, da sie mindestens einem in 3.3 genannten Auswahlkriterien nicht genügen.

Im zweiten Schritt wird mittels Forward-Snowballing gezielt nach neueren relevanten Ansätzen gesucht. Die Werke Venu et al., 2017 [24] und Balachandran et al., 2016 [3] sind zum Stand der Suche so neu, dass keine Werke durch das Forward-Snowballing gefunden werden. Forward Snowballing auf den Werken Wong et al. [26] und Frantzi et al. [9] liefern keine neuen Ergebnisse. Alle zitierenden Werke sind bereits durch das Backward-Snowballing gefunden oder besitzen keine Relevanz zum Thema.

Ausgangs- werk	Snowballing- typ	# Erg.	#relevante Erg.	#neue Erg.	verwendete Treffer
Venu et al.	backward	22	9	0	
Balachandran et al.	backward	13	7	0	
Wong et al.	backward	143	4	0	
Frantzi et al.	backward	91	3	0	
Venu et al.	forward	0	0	0	
Balachandran et al.	forward	0	0	0	
Wong et al.	forward	250	7	0	
Frantzi et al.	forward	184	4	0	

Suchdatum: September 2017

Abbildung 3.2: Ergebnisse des Snowballings

In Abbildung 3.2 werden die Ergebnisse des Snowballings aufgezeigt. Darin enthalten sind die insgesamt durch das entsprechende Snowballing gefunden Ergebnisse (# Erg.), die Anzahl an Ergebnissen die für die Bachelor-Arbeit relevant sind (#relevante Erg.), die relevanten Ergebnisse die bisher noch nicht bekannt waren (# neue Erg.) und die daraus verwendeten Treffern (verwendete Treffer).

3.5 Ergebnisse der Literaturrecherche

In diesem Abschnitt werden die Ergebnisse der Literaturrecherche vorgestellt. Zunächst werden dazu die Forschungsfragen beantwortet. Anschließend werden die Ansätze der relevanten Publikationen im Detail vorgestellt.

3.5.1 Beantwortung der Forschungsfragen

Die Forschungsfragen RQ1, RQ1.1 und RQ1.2 werden im Nachfolgenden Abschnitt beantwortet. Der Bereich der domänenspezifischen Terminologieextraktion wurde in den letzten Jahren immer bedeutender, weshalb mittlerweile eine Vielzahl von Ansätzen zur Extraktion domänenspezifischer Terme existieren. Die Extraktionsansätze verwenden dabei verschiedene, häufig auch mehrere kombinierte Maße und Methoden um einen wichtigen Term zu identifizieren. Die Maße und Methoden lassen sich dabei in vier Kategorien einteilen:

- Statistische Methoden
- Linguistische Methoden
- Maschinelles Lernen
- Graphbasierte Methoden
- Hybride Methoden

Statistische Methoden

Statistische Maße bedienen sich lediglich quantitativer Informationen. Es handelt sich dabei also um rein mathematisch berechnete Werte. [26]

Eines der bekanntesten statistischen Maße, die im Rahmen der Termextraktion genutzt werden, ist die Vorkommenshäufigkeit, kurz TF genannt (engl. Term frequency). Dabei wird die Häufigkeit eines Terms innerhalb eines Dokuments oder einer gegebenen Dokumentensammlung ermittelt. Es wird dabei davon ausgegangen, dass ein Wort, das häufiger innerhalb eines Dokuments auftaucht, wichtiger ist als ein Wort, das weniger häufig auftaucht. Die Vorkommenshäufigkeit ist eine der einfachsten statistischen Maße zur Termextraktion und wurde in vielen Extraktionsansätzen weiterentwickelt. [1]

Eine einfache Anpassung der Vorkommenshäufigkeit durch die inverse Dokumentenhäufigkeit wurde von Salton et al. [22] verwendet. Linden et al. [14] verwendet mit dem Ähnlichkeitsmaß ein weiteres Statistisches Maß. Dies dient der Zuordnung von Termen in unterschiedliche Gruppen, wobei die Gruppen als jeweilige Domäne betrachtet

werden können. Meijer et al. [18] stellt mit Domain Pertinence (DP) , Lexial Cohesion (LC), Domain Consensus (DC) und Structural Relevance (SR) eine Kombination von vier weitere statistischen Methoden vor, die Terme aufgrund verschiedener Wahrscheinlichkeitsberechnungen extrahieren. Die bisher genannten Ansätze sind nur für einfache Terme geeignet und können keine Termkombinationen extrahieren. Die Bestimmung komplexer domänenspezifischer Begriffe wurde von Dymonas et al. [7] durch eine Erweiterung des C-/NC-values realisiert. Der C-Value fand erstmals 1998 bei Frantzi et al. [9] in seiner Grundform Verwendung. Der C/NC-Value erweitert die üblichen statistischen Methoden, nutzt Teile der Linguistik, kann aber lediglich Terme, die aus mehreren Einzeltermen bestehen domänenunabhängig extrahieren. Es existieren weitere statistische Methoden für die domänenspezifische Termextraktion, wie unter anderem von Wong et al. [26] vorgestellt. Die Menge ist jedoch so groß, dass im Rahmen dieser Bachelor-Arbeit nur die den Auswahlkriterien entsprechenden Ansätze berücksichtigt werden.

Großer Vorteil von statistischen Methoden ist der nicht vorhandene Aufwand zur Erstellung von Trainingsdaten. Da die statistischen Methoden auf mathematischer Berechnung basieren, ist meist kein manuelles Eingreifen oder Bereitstellen von Trainingsdaten notwendig. Ein Nachteil statistischer Methoden ist bei der Klassifizierung von selten vorkommenden Termen gegeben. Diese werden, trotz möglicher domänenspezifischer Relevanz, häufig nicht beachtet. [24]

Die gennanten statisitischen Methoden extrahieren domänenspezifische Terme automatisch, sind auf die Ziel-Domäne anwendbar und mit auf der Evaluationsplattform umsetzbar. Allerdings haben weiterführende wissenschaftlichen Arbeiten gezeigt, dass die einfachen statistischen Ansätze eine geringere Precision als beispielsweise Graphbasierte [24] und Hybrid [3] Ansätze besitzen. Einfache statische Methoden werden deshalb in der Evaluationsplattform nicht berücksichtigt.

Linguistische Methoden

Linguistische Maße setzen bei der Extraktion relevanter Terme auf syntaktische Funktionen und Regeln. Die Regeln bestimmten dabei, welche Kombinationen von Wortarten innerhalb eines Satzes zugelassen sind. Zur Anwendung dieser Technik müssen die Sätze daher durch einen sogenannten Part-of-speech-Tagger verarbeitet werden. Dieser ordnet jedem Term innerhalb des Satzes die entsprechende Wortart zu. Nur so lassen sich die durch die Regeln festgelegten Wortartkombinationen filtern. [3]

Das Erstellen geeigneter Regeln ist sehr zeitaufwendig und entscheidet über die Qualität der Extraktion. Linguistische Methoden sind zudem nur geeignet um mögliche Terme zu extrahieren. Die Höhe der Relevanz zu einer Domäne muss mit anderen Mitteln bestimmt werden. [24]

Maschinelles Lernen

Im Bereich des maschinellen Lernens lassen sich grundlegend in überwachte und unüberwachte Methoden unterscheiden. Bei überwachten Methoden wird ein Algorithmus mittels Trainingsdaten angelernt. Die Trainingsdaten bestehen dabei aus manuell gebildeten Paaren von Ein- und Ausgabewerten. Der Algorithmus kann nach entsprechendem Training neue Eingaben auswerten und die korrekte Antwort liefern. Das manuelle Training ist ein aufwändiger Prozess und wird mangels nicht vorhandener Trainingsdaten für diese Bachelor-Arbeit nicht weiter beachtet. [26]

Unüberwachte Methoden erzeugen ohne manuelle Hilfe aus den Eingaben ein geeignetes Vorhersagemodell. Der Algorithmus trainiert sich aus einem entsprechend großen Satz an Daten selbst und nutzt dies zur besseren Vorhersage. [26]

Auch die Verwendung von Kontrastkorpora zur besseren Klassifizierung ist Teil des maschinellen Lernens. Im Bereich der domänenspezifischen Termextraktion sind Kontrastkorpora Texte aus anderen Domänen. Diese werden verwendet um Terme einer Domäne eindeutig zuordnen zu können. Taucht ein Term häufig in unterschiedlichen Domänen auf, ist er höchstwahrscheinlich nicht so relevant wie ein Term, der in einer Domäne häufig, in anderen Domänen aber selten vorkommt. [3]

Kontrastkorpora werden zum Beispiel von Balachandran et al. [3] verwendet. Methoden des maschinellen Lernens erzeugen durch die Notwendigkeit von Testdaten einen hohen Mehraufwand. Trainingsdaten sind nicht immer vorhanden und deren Erstellung ist zeitaufwändig. [24]

Graphbasierte Methoden

Graphbasierte Methoden werden häufig zur Abbildung von Beziehungen zwischen Termen und deren Wichtigkeit genutzt. Diese sogenannten Ontologien basieren meist auf linguistischen Informationen, wie z.B. den entsprechenden Wortarten, um die Beziehung zwischen einzelnen Termen zu modellieren. [26]

Ventura et al [15] stellt mit TeRGraph ein Graph basierten Ansatz für die Termextraktion vor. Die Bewertung wichtiger Terme wird dabei aus den Graphen-basierten Informationen abgeleitet.

Venu et al. [24] verwendet den graphebasierten HITS-Algorithmus zur Auswertung von relevanten Termen. Dabei werden sowohl einfache als auch komplexe Terme berücksichtigt. Die Termkandidaten wurden zuvor durch linguistische Methoden ermittelt. Im direkten Vergleich zu TeRGraph liefert der von Venu et al. vorgestellte Ansatz eine höhere Precision. [24]

Recall und F1-Score werden in der Publikation nicht genannt, weshalb die Bewertung lediglich auf der Precision liegt. Deshalb und durch die Erfüllung aller anderen Kriterien

wird der Algorithmus im Späteren detailliert vorgestellt und Teil der Evaluationsplattform.

Hybrid Methoden

Zur Bestimmung domänenspezifischer Terme bedienen sich moderne Ansätze oft Methoden aus verschiedenen Kategorien [3]. Diese Ansätze werden im Rahmen dieser Bachelor-Arbeit als Hybride Methoden bezeichnet. Sie kombinieren die effektivsten Methoden aus unterschiedlichsten Bereichen und sind so einfachen Methoden meist überlegen. Dies zeigt unter anderem auch die Evaluation von Balachandran et al. [3].

Balachandran et al. nutzt zur Extraktion von möglichen Termen linguistische Methoden und kombiniert diese bei der Auswertung relevanter Terme mit der Wahrscheinlichkeitsberechnung - eine Methode der Statistik. Durch die Verwendung von Kontrastkorpora, zur besseren Zuordnung von Termen zu einer Domäne, nutzt Balachandran et al. zudem auch Methoden des maschinellen Lernens. [3]

Der Ansatz erfüllt alle definierten Kriterien und wird deshalb im späteren Verlauf dieser Bachelor-Arbeit im Detail vorgestellt.

3.5.2 Beschreibung relevanter Ansätze

In diesem Abschnitt werden die relevanten Ansätze für diese Bachelor-Arbeit im Detail beschrieben. Des Weiteren wird dargestellt, weshalb sie den Auswahlkriterien aus Abschnitt 3.3 genügen. Die hier vorgestellten Ansätze wurden durch die Literaturrecherche gefunden und werden in der nachfolgenden Evaluationsplattform dieser Bachelor-Arbeit verwendet.

3.5.2.1 Balachandran et al.

Balachandran et al. [3] stellt einen Ansatz zur Extraktion von einfachen und komplexen domänenspezifischen Termen vor. Als einfache Terme werden in der Publikation Monogramme, also Einzelterme, bezeichnet. Besteht ein Term aus mehreren Einzeltermen (n-Gramm), wird er als komplexer Term bezeichnet. (Beispiel: *admin* ist ein einfacher Term, wohingegen *admin tool* ein komplexer Term ist).

Die Publikation beschreibt sieben bisher existierende Termextraktionsansätze und nennt die Vor- und Nachteile dieses Ansatzes. Das Filtern von domänenspezifischen Termen durch die Verwendung von Termen anderer Domänen liegt bei Balachandran et al. im Fokus. Die Basis des von Balachandran et al. vorgestellten Ansatzes liegt zunächst in der Auswahl des Vergleichskorpora. Da nicht jeder domänenspezifische Text über eine

geeignete Struktur verfügt, wurden nur Texte mit einer hohen lexikalischen Vielfalt (lexical richness) ausgewählt. Ein Text, in dem viele Begriffe häufig vorkommen, verfügt über eine hohe lexikalische Vielfalt. In diesem Rahmen werden vier frei verfügbare sowie zwei über RSS-Feeds aufgebaute domänenspezifische Korpora bezüglich ihrer lexikalischen Vielfalt überprüft. Es eignen sich Korpora aus dem Bereich „Informatik“, „Bio-Medizin“ und „Cricket“. Für die Auswertung von domänenspezifischen Termen werden neben der Zieldomäne auch die entsprechenden Kontrastkorpora gewählt. [3]

Sollen zum Beispiel Terme aus der Ziel-Domäne „Informatik“ ausgewählt werden, besteht der Vergleichskorpora aus den Domänen „Bio-Medizin“ und „Cricket“.

Extraktion möglicher Terme

Die eigentliche Termextraktion basiert auf linguistischen Regeln. Dabei wird ein Part-of-speech-Tagger verwendet und jeder Satz in seine Wortarten zerlegt. Der getaggte Satz wird anschließend auf fünf verschiedene linguistische Regeln geprüft. Die linguistischen Regeln geben dabei vor, welche Kombinationen von Wortarten erlaubt sind. Beachtet werden dabei lediglich Kombinationen, die aus einem bis maximal drei Einzeltermen bestehen. Enthält ein Satz eine regelkonforme Kombination, wird diese zu einer Liste der möglichen relevanten Terme hinzugefügt. [3]

Bestimmung domänenspezifischer Terme

Aus der Liste der möglichen relevanten Terme werden die tatsächlich relevanten Terme über ihre statistische Verteilung innerhalb der Zieldomäne bestimmt. Dabei wird die Wahrscheinlichkeit berechnet mit der ein Term zu der Zieldomäne gehört. Bei komplexen Termen wird neben der Wahrscheinlichkeit des komplexen Terms zur Zieldomäne auch die einzelnen Wahrscheinlichkeiten der enthaltenen Einzeltermen bestimmt. Die jeweiligen Wahrscheinlichkeiten werden miteinander multipliziert, um die Gesamtwahrscheinlichkeit des komplexen Terms zu erhalten. Um die Eindeutigkeit zu der Zieldomäne zu festigen, wird dieser Wahrscheinlichkeitswert durch den Wahrscheinlichkeitswert geteilt, mit dem der Term zu einer anderen Domäne gehört. Da es mehrere Kontrast-Domänen gibt, wird die Domäne mit dem höchsten Wahrscheinlichkeitswert für diesen Term gewählt. [3]

Precision im Vergleich

Die abschließende Evaluation der Publikation vergleicht den vorgestellten Ansatz mit DC und C-Value. Bei einfachen Termen liefert der vorgestellte Ansatz im Vergleich zu DC einen um 8% bis 25% höheren positiven Vorhersagewert (Precision). Komplexe Terme werden im Vergleich zu C-Value sogar mit einer knapp 25% bis 30% höheren

Genauigkeit gefunden. Abhängig sind diese Ergebnisse von der gewählten Domäne und damit dem Kontrastkorpora. [3]

Evaluation

Der Ansatz von Balachandran et al. extrahiert domänenspezifische Terme ohne manuelle Unterstützung [3]. Die in der Publikation verwendete Domäne entspricht der Domäne dieser Evaluationsplattform, eine Anpassung an die Domäne muss daher nicht vorgenommen werden. Die Auswahl des Kontrastkorpora wird auf die frei verfügbaren Korpora angepasst. Die Evaluation von Balachandran et al. hat gezeigt, dass der vorgestellte Ansatz eine hohe Precision im Vergleich zu anderen renommierten Ansätzen aufweist. Recall und F1-Score werden in dieser Publikation nicht verwendet, weshalb die Precision als Klassifikator ausreichen muss. Die Umsetzung des Algorithmus kann auf der Evaluationsplattform ohne Einschränkung vorgenommen werden. Zudem existiert kein Ansatz, der dem Vorgehen von Balachandran et al. ähnlich ist. Der von Balachandran et al. vorgestellte Extraktionsansatz genügt damit allen unter Kapitel 3.3 vorgestellten Kriterien und wird in der Evaluationsplattform berücksichtigt.

3.5.2.2 Venu et al.

Venu et al. [24] stellt in der Publikation einen Ansatz zur unüberwachten Erstellung von Ontologien aus Texten vor. Die Termextraktion ist jedoch ein wichtiger Teil zum Erstellen von Ontologien, weshalb dieser Teil der Publikation ihre Relevanz zu diesem Forschungsthema behält. Es wird daher lediglich auf den verschwendeten Termextraktionsansatz eingegangen, da die weiterführende Verarbeitung außerhalb des Rahmens dieser Bachelor-Arbeit liegt.

Der vorgestellte Ansatz zur Identifizierung domänenspezifischer Terme basiert auf dem HITS-Algorithmus. Ein graphbasierter Algorithmus, der die wichtigsten Terme über Hubs und Authorities bestimmt. [24]

Extraktion möglicher Terme

Der Ansatz unterscheidet bei der Extraktion der möglichen Terme zwischen einfachen und komplexen Termen. Einfache Terme sind dabei einzelne Worte, wohingegen komplexe Terme als Kombination mehrerer Einzelterme gesehen werden. Zur Bestimmung komplexer Terme wird der Text satzweise durch den Stanford Dependency Parser analysiert. Dieser identifiziert die Shallow Semantic Relations (SSR) zwischen den einzelnen Worten. Alle Nomen stellen einfache Terme dar.

Bestimmung domänenspezifischer Terme

Zur Bestimmung domänenspezifischer Terme wird der HITS-Algorithmus verwendet. Dabei werden die Hubs durch SSR und die Authorities durch Nomen dargestellt. Die Verbindung zwischen Hubs und Authorities ist durch die Häufigkeit bestimmt, mit der SSR und Nomen gemeinsam innerhalb des Korpora vorkommen. Die Hubs-Gewichte werden aus der Summe der Authority-Gewichte berechnet und umgekehrt. Die Berechnung wird iterativ bis zur Konvergenz fortgeführt. Die SSR mit dem höchsten Hubs-Gewichten werden als komplexe domänenspezifische Terme ausgewählt. Nomen mit den höchsten Authority-Gewichten stellen hingegen einfache domänenspezifische Terme dar. [24]

Precision im Vergleich

In einer kurzen Evaluation wird der verwendete HITS-Algorithmus gegen die Ansätze LIDF, TeRGraph und eine Kombination aus DP, DC, LC und SR verglichen. Als Maßstab dient lediglich der positive Vorhersagewert (Precision), der in Abhängigkeit zur Anzahl der extrahierten Terme angegeben wird. Mit steigender Anzahl der gewählten Terme sinkt die Precision. Dennoch liefert die Termextraktion mittels des HITS-Algorithmus im Durchschnitt eine um knapp 8% genauere Identifizierung als die anderen verglichenen Ansätze. [24]

Evaluation

Der von Venu et al. vorgestellte graphbasierte Extraktionsansatz kann ohne manuelle Hilfe domänenspezifische Terme extrahieren [24]. Er ist unabhängig von Domänen und ist daher mit den zur Verfügung stehenden Testdaten kompatibel. In der Evaluation hat Venu et al. gezeigt, dass der vorgestellte Ansatz eine höhere Precision als andere bekannte Termextraktionsansätze besitzt. Da der genutzte Stanford Dependency Parser frei verfügbar ist, lässt sich der Ansatz auch auf der späteren Evaluationsplattform implementieren. Der Ansatz unterscheidet sich zudem gravierend von dem bereits ausgewählten Ansatz. Damit genügt der von Venu et al. vorgestellte Algorithmus den in Kapitel 3.3 definierten Kriterien.

4 Implementierung der Evaluationsplattform

In diesem Kapitel wird die Architektur sowie die Funktionsweise der Evaluationsplattform erläutert. Dabei wird zunächst eine Übersicht geliefert und die allgemeine Struktur der Evaluationsplattform beschrieben. Im Anschluss wird das implementierte Programm im Detail vorgestellt. Insbesondere die notwendigen Anpassungen, Architektur, Bedienung und Ausgabe werden hier beleuchtet.

4.1 Allgemeine Implementierungsaspekte

In diesem Abschnitt wird eine Übersicht der Evaluationsplattform gegeben. Des Weiteren werden die allgemeingültigen Aspekte der Evaluationsplattform erläutert. Gemeinsam verwendete Hilfsmittel und Funktionen werden ebenfalls geschildert.

4.1.1 Entwicklungsumgebung

Die Evaluationsplattform und alle dazugehörigen Funktionen sind vollständig in der Programmiersprache JAVA umgesetzt. Diese Programmiersprache bietet die notwendigen Eigenschaften, um die Anforderungen an die Evaluationsplattform umzusetzen. Zudem ist JAVA betriebssystemunabhängig und kann damit auf allen Systemen verwendet werden. Die Versionsverwaltung wird über GitHub umgesetzt. So kann auf vorherige Versionen zurückgegriffen und der Status der Veränderungen jederzeit nachvollzogen werden. Eine spätere Weiterentwicklung ist zudem ohne weitere Einschränkung möglich. Als Entwicklungsumgebung wird Eclipse genutzt. Dieses Programmierwerkzeug bietet neben vielen automatisierten Programmierhilfen auch eine reibungslose Integration der Versionsverwaltung über Git.

4.1.2 Natural Language Parser

Alle in der Evaluationsplattform implementierten Ansätze benötigen zur Bestimmung möglicher Terme Natural Language Parser. Balachandran et al. benötigt einen Part-Of-

Speech-Tagger um bestimmte Wortarten zu finden. Des Weiteren werden Worte lemmatisiert (auf ihre Grundform gebracht), auch hierfür ist ein geeigneter Natural Language Parser notwendig. [3]

Venu et al. verwendet zunächst einen Dependency Parser, um Shallow Semantic Relations aus dem Text zu extrahieren. Für die anschließende Extraktion von Nomen wird ebenfalls ein Part-Of-Speech-Tagger benötigt. [24]

Für die Evaluationsplattform wird daher der Stanford Parser als Natural Language Parser verwendet. Der Stanford Parser liefert alle benötigten Funktionen, um beide Ansätze mit den notwendigen Informationen zu versorgen. Zudem ist er in einer frei verfügbaren Java-Bibliothek nutzbar und ist ausführlich dokumentiert. [16]

4.1.3 Architektur Übersicht

Die Evaluationsplattform ist vollständig objektorientiert umgesetzt und besteht aus insgesamt sieben Klassen, die in der Abbildung 4.1 zu sehen sind.

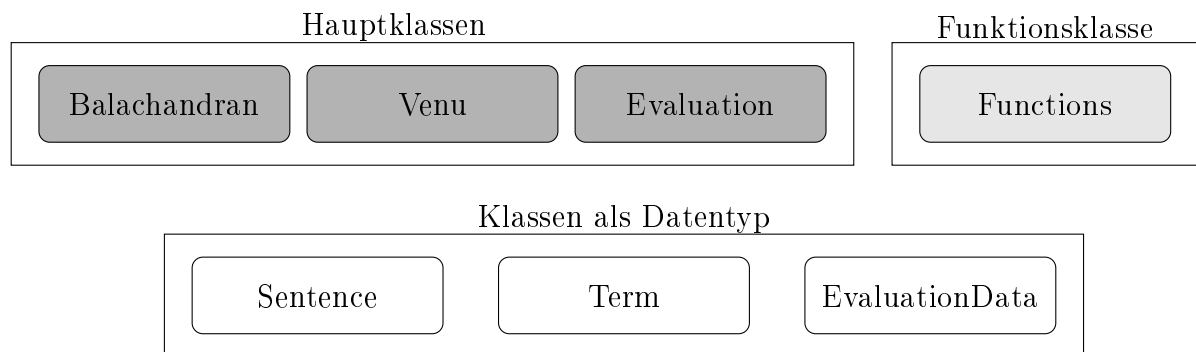


Abbildung 4.1: Implementierte Klassen

Jede Klasse ist dabei eine separate JAVA-Datei. Die Ansätze von Balachandran et al. und Venu et al. sowie die Evaluation sind jeweils in einer separaten Klasse implementiert. Diese stellen dabei den Kern dieser Evaluationsplattform dar und können damit als Hauptklassen bezeichnet werden. Allgemeine Funktionen, wie z.B. das Einlesen oder Schreiben von Dateien, die in mehreren Klassen Verwendung finden, sind in eine alleinstehende Funktionsklasse ausgelagert. Die weiteren drei Klassen (**Sentence**, **Term**, & **EvaluationData**) werden für das Zwischenspeichern, der zum Teil komplexen Datenstrukturen, verwendet. Durch das Instanzieren dieser Klassen werden die Klassenvariablen des entsprechenden Objekts gefüllt. Jedes Objekt stellt dabei einen Datensatz dar. Die Daten bleiben damit immer im Hauptspeicher und können dadurch im Gegensatz zur lokalen Zwischenspeicherung schneller weiterverarbeitet werden. Dieser Klassentyp wird auch als Datentyp-Klasse bezeichnet. Datentyp-Klassen besitzen keine eigene

Funktionalität und werden lediglich für das Ablegen und Verarbeiten von Informationen verwendet. [8]

4.1.4 Bedienung Übersicht

Die Bedienung der Evaluationsplattform erfolgt in zwei Schritten. Im ersten Schritt wird das Programm zum Ansatz von Balachandran et al. oder Venu et al. ausgeführt. Die dadurch entstandenen Dateien werden im zweiten Schritt durch die Evaluation ausgewertet und die Ergebnisse in einer weiteren Datei abgelegt (siehe Abbildung 4.2).

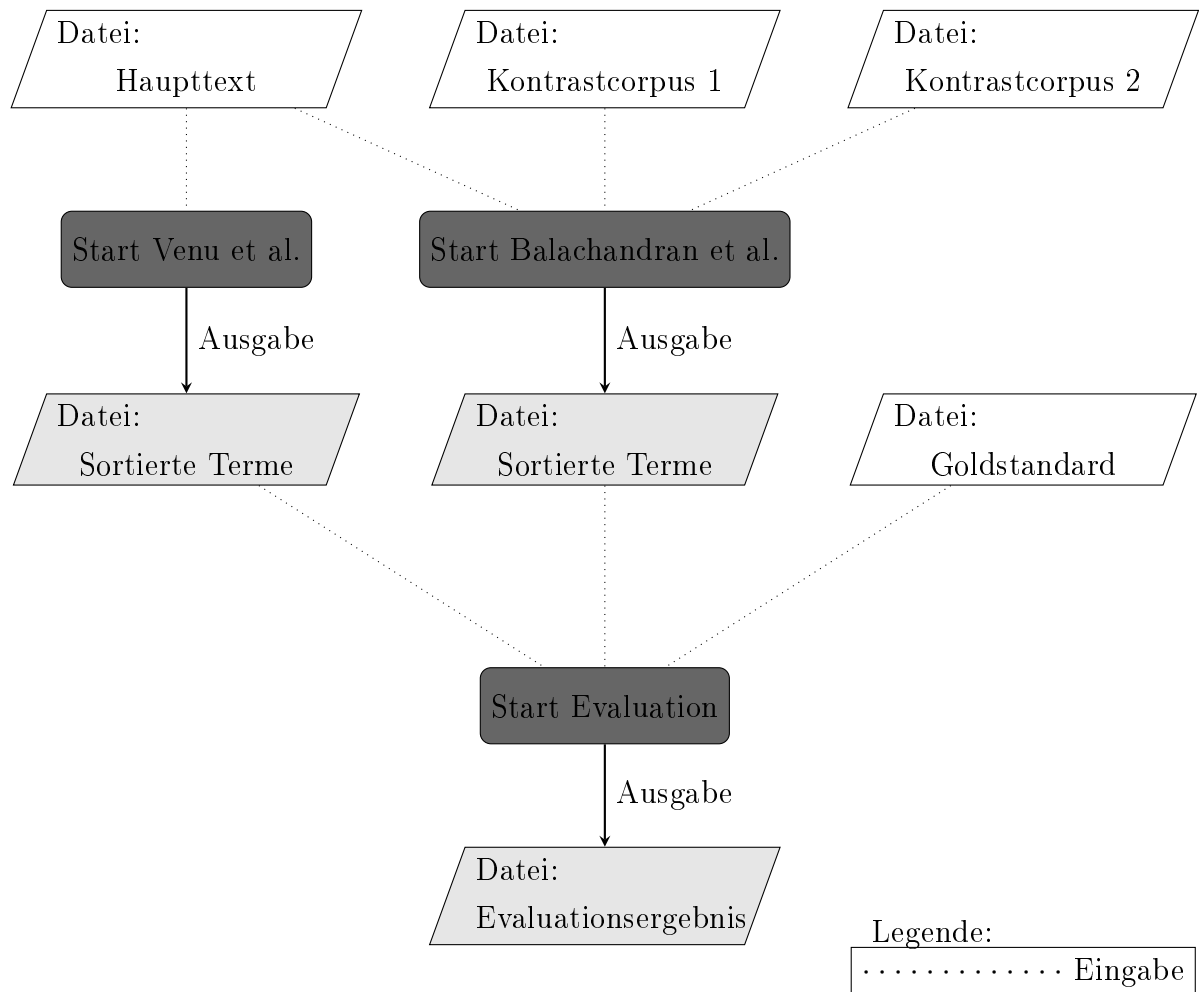


Abbildung 4.2: Übersicht zur Bedienung der Evaluationsplattform

Die Programme in Schritt eins benötigen unterschiedliche Eingaben. Das Programm zu Venu et al. benötigt lediglich den Haupttext. Balachandran et al. benötigt neben dem Haupttext auch noch die Eingabe von beiden Kontrastkorpora. Beide Programme liefern als Ausgabe Dateien, in der die gefundenen Terme nach Relevanz absteigend

sortiert sind. Für die Ausführung der Evaluation in Schritt zwei ist eine in Schritt eins erstellte Datei sowie der Goldstandard notwendig. Die notwendigen Eingaben sind in der Abbildung 4.2 in weiß gefüllt, die Ausgabedateien in hellem grau. Das Ergebnis der Evaluation ist eine Datei, in der die gefundenen Terme bezüglich der Metriken Precision, Recall und F1-Score bewertet werden. Beide Schritte können unabhängig voneinander, auch zeitlich versetzt, gestartet werden. Zu beachten ist hierbei lediglich, dass zur Ausführung der Evaluation bereits eine Datei aus Schritt eins vorhanden sein muss.

4.2 Implementierung des Ansatzes von Balachandran et al.

Eine Implementierung des von Balachandran et al. verwendeten Ansatzes, zur Bestimmung der domänenspezifischen Terme, ist nicht öffentlich verfügbar. Der Ansatz muss daher selbst in einem Programm implementiert werden. Als Vorlage für die Umsetzung dient dabei die Publikation von Balachandran et al [3].

4.2.1 Notwendige Anpassungen

Der Ansatz von Balachandran et al. extrahiert mittels fünf unterschiedlichen Regeln Monogramme, Bigramme und Trigramme. Der Goldstandard enthält jedoch nur Monogramme und Bigramm. Deshalb wird das Programm so angepasst, dass die Länge der Terme, die extrahiert werden sollen, frei wählbar ist. In der späteren Evaluation können so lediglich Monogramme und Bigramme extrahiert werden, um den Goldstandard besser abzubilden.

Im Ansatz von Balachandran et al. werden des Weiteren Terme aus verschiedenen Domänen extrahiert. Hierfür wird der Haupttext mehrmals mit unterschiedlichen Kontrastkorpora ausgewertet. Dadurch werden beispielsweise zunächst die im Haupttext enthaltenen Terme für die Domäne Informatik extrahiert, anschließend die Terme für die Domäne Bio-Medzin. Für diese Bachelor-Arbeit ist lediglich die Extraktion aus einer Domäne relevant, weshalb das Programm dahingehend angepasst wird. Durch eine entsprechende Auswahl des Kontrastkorpora lassen sich jedoch, durch mehrere Ausführungen des Programms, weiterhin verschiedene domänenspezifische Terme extrahieren.

4.2.2 Architektur

Die Implementierung des Ansatzes von Balachandran et al. lässt sich grob in acht Arbeitsschritte aufteilen. Die Arbeitsschritte entsprechen im Programm jeweils einer Methode. Die Methoden rufen sich teilweise gegenseitig auf oder werden durch eine Steuermethode aufgerufen. Für das bessere Verständnis des Programms sind unwichtige Methoden ausgelassen und das Modell vereinfacht dargestellt. Die vereinfachte Architektur lässt sich der Abbildung 4.3 entnehmen. Im Nachfolgenden wird diese Architektur erläutert.

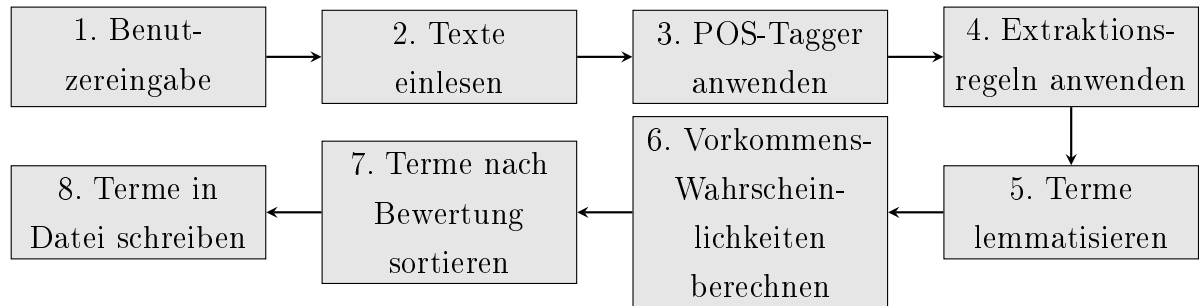


Abbildung 4.3: Architektur des Algorithmus von Balachandran et al.

Im ersten Schritt (1.) werden verschiedene Benutzereingaben eingelesen. Hier werden neben dem Pfad zum Haupttext und den Kontrastcorpora auch andere Informationen, wie z.B. die Größe der Termlänge, der Ausgabepfad und anderes abgefragt. Im zweiten Schritt (2.) werden die Texte aus den angegebenen Pfaden eingelesen und für die weitere Verarbeitung vorbereitet. Anschließend wird der Haupttext durch einen POS-Tagger, in diesem Fall dem Stanford Parser, analysiert und die Terme durch ihre entsprechende Wortart identifiziert (3.). Im vierten Schritt (4.) werden die Terme anhand der gegebenen Regeln extrahiert. Basierend auf den Einstellungen aus (1.) können Monogramme, Bigramme und Trigramme berücksichtigt werden. Die gefundenen Terme werden daraufhin lemmatisiert (5.), also auf ihre Grundform gebracht. Zur Bewertung der Terme wird die Vorkommens-Wahrscheinlichkeit im Haupttext durch die größte Vorkommens-Wahrscheinlichkeit der Kontrastcorpora geteilt (6.). Auf Basis der in Schritt (6.) berechneten Werte wird die Termliste in Schritt (7.) absteigend sortiert. Die sortierte Termliste wird im letzten Schritt (7.) als Text-Datei unter dem in (1.) angegebenen Pfad gespeichert. Diese Text-Datei wird für die spätere Evaluation benötigt.

4.2.3 Bedienung

Nach dem Start des Programms werden insgesamt sechs Benutzereingaben gefordert. Mit den Eingaben wird die Ein- und Ausgabe sowie das zu erwartende Ergebnis gesteuert. In Abbildung 4.4 werden die Eingaben in der tatsächlichen Reihenfolge angezeigt. Erst wenn die Eingabe des jeweiligen Schrittes korrekt ist, kann die nächste Eingabe getätigt werden. Im Nachfolgenden werden die Schritte detailliert erklärt.

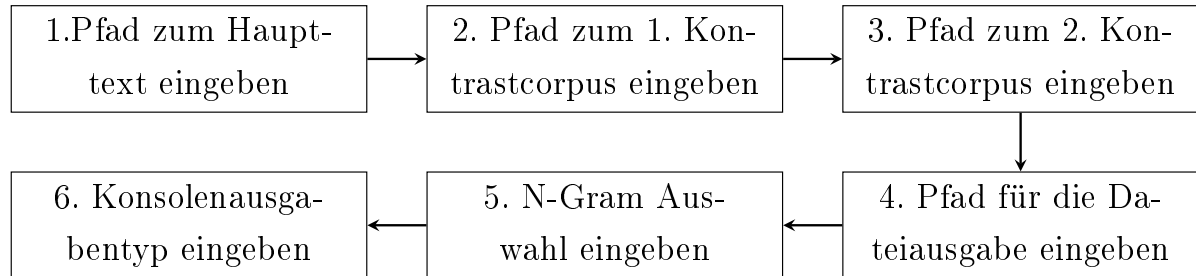


Abbildung 4.4: Reihenfolge der Benutzereingaben für die Implementierung von Balachandran et al.

In den ersten drei Schritten wird der exakte Pfad zum Haupttext, zum ersten Kontrastkorpus und zum zweiten Kontrastkorpus angegeben. Es sind nur existierende Dateipfade mit der Endung „.txt“ erlaubt. Pfade zu nicht existierenden Dateien oder Dateien mit anderem Dateiformat werden nicht angenommen und führen zu einer erneuten Eingabeaufforderung. Wird eine leere Eingabe bestätigt, so wählt das Programm automatisch die im Klassenpfad hinterlegten Texte aus. Dabei handelt es sich um den, in der späteren Evaluation verwendeten Haupttext sowie den Kontrastkorpora von GENIA (Bio-Medizin) und Reuters (Wirtschaft).

Im vierten Schritt wird die Eingabe des Pfades für die spätere Ausgabe verlangt. Auch hier werden nur existierende Pfade angenommen. Pfade zu nicht existierenden Ordnern oder Dateien jeglicher Art sind nicht gestattet und führen zu einer erneuten Eingabeaufforderung. Wird eine leere Eingabe bestätigt, werden die Dateien im Klassenpfad gespeichert.

Mit der fünften geforderten Eingabe wird klassifiziert, welche Termlänge bei der Termextraktion berücksichtigt wird. Es ist möglich nur Monogramme (Eingabe „1“), Monogramme und Bigramme (Eingabe „2“) oder Monogramme, Bigramme und Trigramme (Eingabe „3“) zu extrahieren. Die Eingabe ist verpflichtend und alle anderen als die oben genannten Eingaben werden nicht akzeptiert und führen zu einer erneuten Eingabeaufforderung.

Die letzte Eingabe bestimmt, ob die Ergebnisse zusätzlich in der Konsole angezeigt

werden sollen. Es sind lediglich die Eingaben „0“ (Keine Ausgabe) und „1“ (Mit Konsolenausgabe) möglich.

Nach Bestätigung der letzten Eingabe wird die Berechnung gestartet und liefert die den Eingaben entsprechende Ausgabe. Die Struktur der Ausgabe kann in Abschnitt 4.2.4 betrachtet werden.

4.2.4 Ausgabe

Das implementierte Programm zum Ansatz von Balachandran et al. liefert zwei Ausgabedateien. Dabei handelt es sich um Text-Dateien (*.txt), die dem Pfad gespeichert werden, der in der Eingabe ausgewählt wird. Die erste Textdatei beinhaltet alle gefundenen Terme, die absteigend ihrer Bewertung aufgelistet sind (siehe Abbildung 4.5).

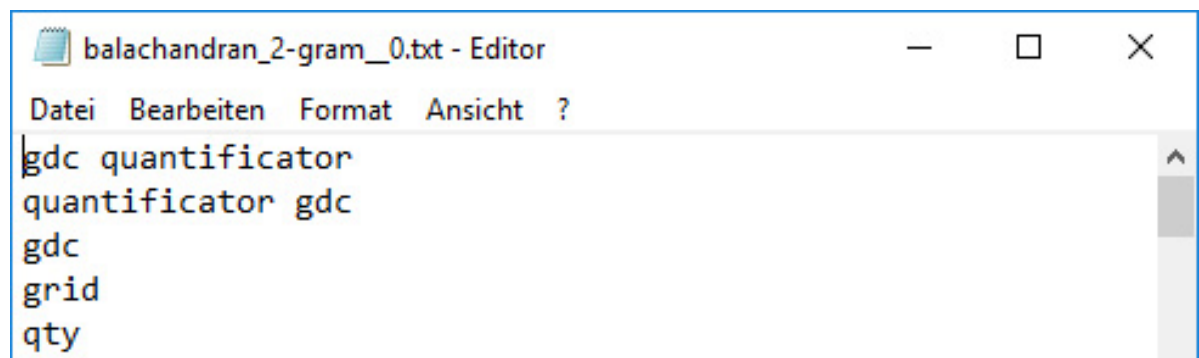


Abbildung 4.5: Sortierte Ausgabedatei des Ansatzes von Balachandran et al.

Die zweite Textdatei beinhaltet, neben den absteigend aufgelisteten Termen, auch noch die berechnete Bewertung. Diese ist mit einem Unterstrich („_“) an den Term angehängt (siehe Abbildung 4.6).

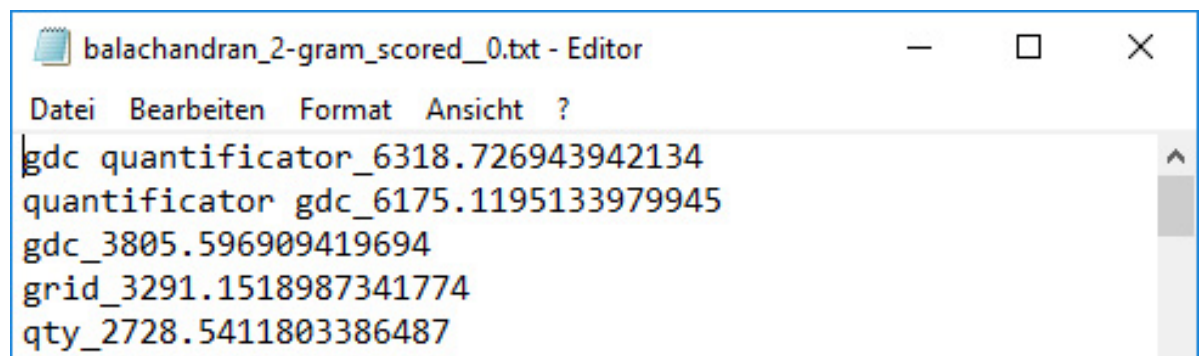


Abbildung 4.6: Sortierte Ausgabedatei des Ansatzes von Balachandran et al. inklusive Bewertung

Jede Zeile in den Textdateien entspricht einem gefundenen Term. Aus den Dateinamen lässt sich der jeweilige Ansatz (hier „balachandran“), die ausgewählte Termlänge (hier „2-gram“) und die Art des Ausgabeformats („scored“ = mit Bewertung) ablesen. Die Dateien sind zudem durchnummeriert, sodass bei einer erneuten Ausführung die ursprüngliche Datei nicht überschrieben wird.

Für das Evaluations-Programm können beide Dateien verwendet werden. Das Evaluations-Programm berücksichtigt derzeit jedoch lediglich die Reihenfolge und nicht die Höhe der Bewertung. Für spätere Erweiterung oder die manuelle Kontrolle kann die Ausgabedatei mit den Bewertungen dennoch hilfreich sein.

Zusätzlich zu den Ausgabedateien kann das Ergebnis auch in der Konsole ausgegeben werden. Diese Ausgabe ist optional und wird durch eine Benutzereingabe beim Start des Programms aktiviert (siehe Abschnitt 4.2.3).

4.3 Implementierung des Ansatzes von Venu et al.

Eine Implementierung des Ansatzes von Venu et al. ist nicht frei verfügbar. Der Ansatz muss daher selbst in einem Programm implementiert werden. Die Beschreibung in der Publikation von Venu et al. [24] und der darin referenzierten Publikation von Mukherjee et al. [19] dienen dabei als Vorlage für die Programmierung.

4.3.1 Notwendige Anpassungen

Die für diese Bachelor-Arbeit relevante Termextraktion stellt in der Publikation von Venu et al. nur einen kleinen Teil des vorgestellten Konzepts dar. Die weiterführende Erstellung einer Ontologie ist nicht Teil dieser Bachelor-Arbeit und wird daher nicht berücksichtigt. Es wird deswegen lediglich der Ansatz zur Bestimmung domänenspezifischer Terme umgesetzt.

4.3.2 Architektur

Das Programm zum Ansatz von Venu et al. lässt sich grob in sieben Arbeitsschritte aufteilen. Die Arbeitsschritte entsprechen im Programm jeweils einer Methode. Die Methoden rufen sich teilweise gegenseitig auf oder werden durch eine Steuermethode aufgerufen. Für das bessere Verständnis des Programms sind unwichtige Methoden ausgelassen und das Modell vereinfacht dargestellt.

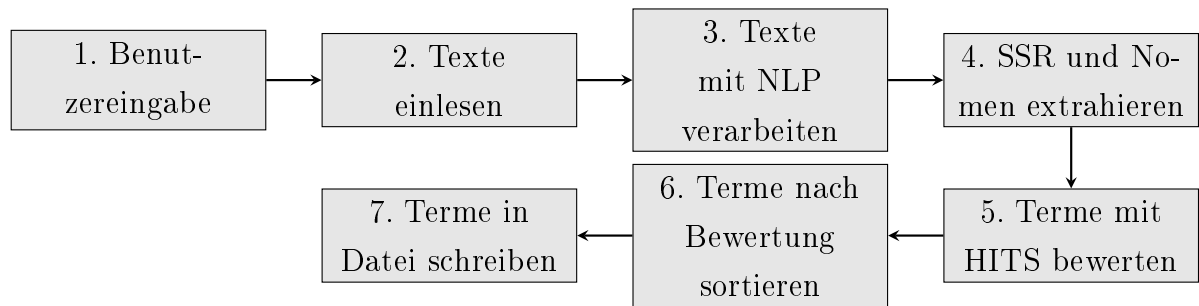


Abbildung 4.7: Architektur des Algorithmus von Venu et al.

Die grobe Architektur lässt sich der Abbildung 4.7 entnehmen. Im Nachfolgenden wird diese Architektur erläutert.

Im ersten Schritt (1.) werden verschiedene Benutzereingaben eingelesen. Neben dem Pfad zum Haupttext und dem der Ausgabe wird auch die gewünschte Termlänge (Monogramme, Bigramme, Trigramme, etc.) abgefragt. Im zweiten Schritt (2.) werden die Texte aus den angegebenen Pfaden eingelesen und für die weitere Verarbeitung vorbereitet. Anschließend wird der Haupttext durch einen umfangreichen Natural Language Parser analysiert (3.). In diesem Fall liefert der Stanford Parser alle, für die späteren Zwecke, notwendigen Informationen, wie z.B. die im Text enthaltenen Shallow Semantic Relations und Nomen. Die SSR werden aus den vom Stanford Parser gelieferten „Universal dependencies“ extrahiert. Für die Nomen wird der „POS-Tagger“ verwendet. Die durch den Parser erlangten Informationen, werden in Schritt (4.) zur Extraktion der SSR und Nomen verwendet. Mit dem HITS-Algorithmus werden die Terme bewertet (5.). SSR stellen dabei die Hubs, Nomen die Authorities dar. Die bewerteten Terme werden in Schritt (6.) absteigend sortiert. Im letzten Schritt werden die bewerteten Terme in eine Datei geschrieben. Es werden Insgesamt drei Dateien erzeugt: Eine Datei für die bewerteten Nomen, eine Datei für die bewerteten SSR und eine Datei in der Nomen und SSR zusammen enthalten sind. Diese Text-Dateien werden für die spätere Evaluation benötigt.

4.3.3 Bedienung

Nach dem Start des Programms werden insgesamt fünf Benutzereingaben gefordert. Mit den Eingaben wird die Ein- und Ausgabe sowie das zu erwartende Ergebnis gesteuert. In Abbildung 4.8 werden die Eingaben in der tatsächlichen Reihenfolge angezeigt. Erst, wenn die Eingabe des jeweiligen Schrittes korrekt ist, kann die nächste Eingabe getätigt werden. Im Nachfolgenden werden die Schritte detailliert erklärt.

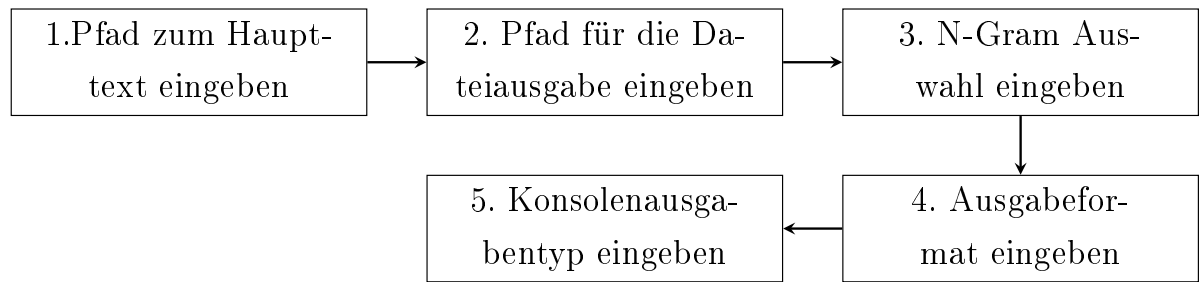


Abbildung 4.8: Reihenfolge der Benutzereingaben für die Implementierung von Venu et al.

Im ersten Schritt wird der exakte Pfad zum Haupttext angegeben. Es sind nur existierende Dateipfade mit der Endung „*.txt“ erlaubt. Pfade zu nicht existierenden Dateien oder Dateien mit anderem Dateiformat werden nicht angenommen und führen zu einer erneuten Eingabeaufforderung. Wird eine leere Eingabe bestätigt, so wählt das Programm automatisch den im Klassenpfad hinterlegten Haupttext aus.

Im zweiten Schritt wird die Eingabe des Pfades für die spätere Ausgabe gefordert. Auch hier werden nur existierende Pfade angenommen. Pfade zu nicht existierenden Ordnern oder Dateien jeglicher Art sind nicht gestattet und führen zu einer erneuten Eingabeaufforderung. Wird eine leere Eingabe bestätigt, werden die Dateien im Klassenpfad gespeichert.

Mit der dritten geforderten Eingabe wird klassifiziert, welche Termlänge bei der Termextraktion der SSR berücksichtigt wird. Es ist möglich SSRs mit maximal zwei (Eingabe „2“), drei (Eingabe „3“) oder vier (Eingabe „4“) Einzeltermen zu extrahieren. Die Eingabe ist verpflichtend und alle anderen als die oben genannten Eingaben werden nicht akzeptiert und führen zu einer erneuten Eingabeaufforderung.

Im vierten Schritt wird das Ausgabeformat gewählt. Es ist möglich die Terme ohne der berechneten Bewertung (Eingabe „0“) oder mit berechneter Bewertung zu speichern (Eingabe „1“). Die Darstellung und Auswirkung wird in Abschnitt 4.3.4 beschrieben.

Die letzte Eingabe bestimmt, ob die Ergebnisse zusätzlich in der Konsole angezeigt werden sollen. Es sind lediglich die Eingaben „0“ (Keine Ausgabe) und „1“ (Mit Konsolenausgabe) möglich.

Nach Bestätigung der letzten Eingabe wird die Berechnung gestartet und liefert die den Eingaben entsprechende Ausgabe. Die Struktur der Ausgabe wird in Abschnitt 4.3.4 beschrieben.

4.3.4 Ausgabe

Das implementierte Programm zum Ansatz von Venu et al. liefert drei Ausgabedateien. Dabei handelt es sich um Text-Dateien (*.txt), die in dem Pfad gespeichert werden, der in der Eingabe ausgewählt wird. Die Dateien beinhalten die gefundenen Terme, die absteigend ihrer Bewertung zeilenweise aufgelistet sind. Für die jeweiligen Extraktionsarten existiert jeweils eine Datei. Die erste Datei enthält die gefundenen Nomen, die zweite Datei die gefundenen SSR und die dritte Datei die Kombination aus SSR und Nomen (siehe Abbildung 4.9). Abhängig von der Benutzereingabe (siehe Abbildung 4.3.3) beinhalten die Dateien Terme mit oder ohne Bewertung.

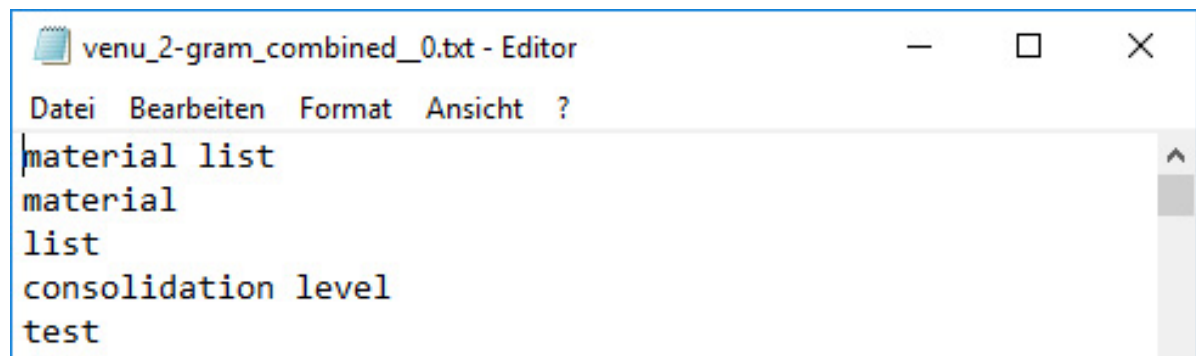


Abbildung 4.9: Sortierte Ausgabedatei von SSR und Nomen des Ansatzes von Venu et al.

Aus dem Dateinamen lässt sich der jeweilige Ansatz (hier „venu“), die ausgewählte Termlänge der SSR (hier „2-gram“), die Extraktionsarten (hier „combined“) und die Art des Ausgabeformats („leer“ = ohne Bewertung) ablesen. Die Dateien sind zudem durchnummeriert, sodass bei einer erneuten Ausführung die ursprüngliche Datei nicht überschrieben wird.

Für das Evaluations-Programm können alle Dateien verwendet werden. Das Evaluations-Programm berücksichtigt derzeit jedoch lediglich die Reihenfolge und nicht die Höhe der Bewertung. Für spätere Erweiterung oder die manuelle Kontrolle kann die Ausgabedatei mit den Bewertungen dennoch hilfreich sein.

Zusätzlich zu den Ausgabedateien kann das Ergebnis auch in der Konsole ausgegeben werden. Diese Ausgabe ist optional und wird durch eine Benutzereingabe beim Start des Programms aktiviert (siehe Abschnitt 4.3.3).

4.4 Implementierung des Evaluations-Programms

Das Evaluations-Programm dient zur Auswertung der extrahierten Terme bezüglich Precision, Recall und F1-Score. Die Architektur, Bedienung sowie Ausgabe werden in diesem Abschnitt beschrieben.

4.4.1 Architektur

Das Programm zur Evaluation lässt sich grob in vier Arbeitsschritte aufteilen. Die Arbeitsschritte entsprechen im Programm jeweils einer Methode. Die Methoden rufen sich teilweise gegenseitig auf oder werden durch eine Steuermethode aufgerufen. Für das bessere Verständnis des Programms sind unwichtige Methoden ausgelassen und das Modell vereinfacht dargestellt.

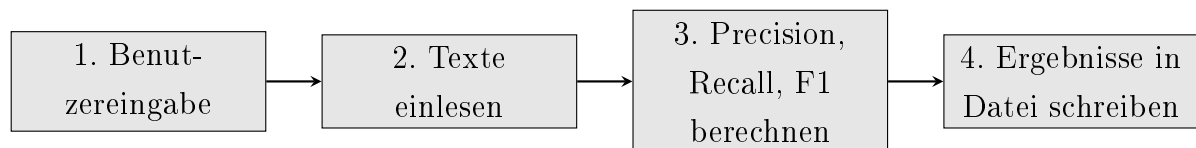


Abbildung 4.10: Architektur des Evaluations-Programms

Die grobe Architektur lässt sich der Abbildung 4.10 entnehmen. Im Nachfolgenden wird diese Architektur erläutert.

Im ersten Schritt (1.) werden verschiedene Benutzereingaben eingelesen. Hier wird die auszuwertende Termliste, der Goldstandard und der Ausgabepfad abgefragt. Im zweiten Schritt (2.) werden die angegebenen Dateien eingelesen und für die weitere Verarbeitung vorbereitet. Anschließend werden die Metriken Precision, Recall und F1-Score für die Top N Terme berechnet. Im letzten Schritt werden die berechneten Ergebnisse in eine Text-Datei geschrieben. Die Datei wird unter dem in der Eingabe spezifizierten Pfad gespeichert. Der Inhalt der Text-Datei wird im Abschnitt 4.4.3 genauer beschrieben.

4.4.2 Bedienung

Nach dem Start des Programms werden insgesamt drei Benutzereingaben gefordert. Mit den Eingaben wird die Ein- und Ausgabe gesteuert. In Abbildung 4.11 werden die Eingaben in der tatsächlichen Reihenfolge angezeigt. Erst, wenn die Eingabe des jeweiligen Schrittes korrekt ist, kann die nächste Eingabe getätigt werden. Im Nachfolgenden werden die Schritte detailliert erklärt.

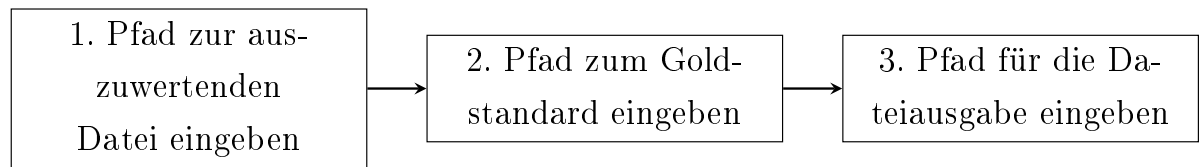


Abbildung 4.11: Reihenfolge der Benutzereingaben für die Implementierung der Evaluation

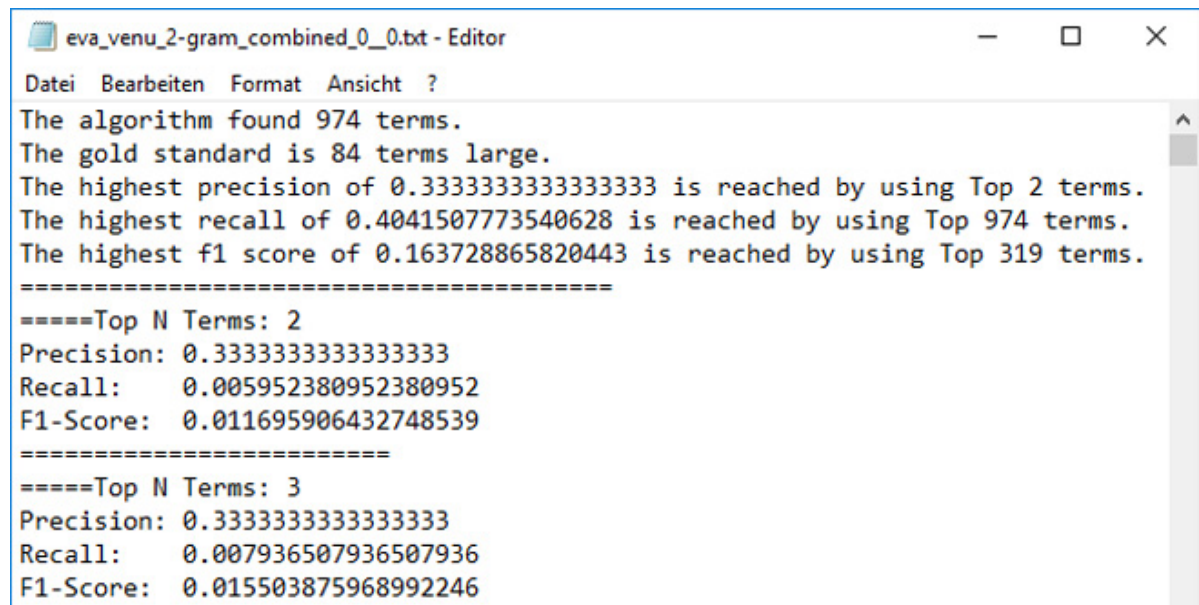
In den ersten beiden Schritten wird der Dateipfad zur auszuwertenden Termliste und dem Goldstandard angegeben. Es sind nur existierende Dateipfade mit der Endung „*.txt“ erlaubt. Pfade zu nicht existierenden Dateien oder Dateien mit anderem Dateiformat werden nicht angenommen und führen zu einer erneuten Eingabeaufforderung. Die Eingabe der auszuwertenden Datei ist verpflichtend. Durch Bestätigen einer leeren Eingabe im zweiten Schritt kann der im Klassenpfad hinterlegte Goldstandard verwendet werden.

Im letzten Schritt wird die Eingabe des Pfades für die spätere Ausgabe gefordert. Auch hier werden nur existierende Pfade angenommen. Pfade zu nicht existierenden Ordnern oder Dateien jeglicher Art sind nicht gestattet und führen zu einer erneuten Eingabeaufforderung. Wird eine leere Eingabe bestätigt, werden die Dateien im Klassenpfad gespeichert.

Nach Bestätigung der letzten Eingabe wird die Berechnung gestartet und die Ausgabe erzeugt. Die Struktur der Ausgabe wird in Abschnitt 4.4.3 beschrieben.

4.4.3 Ausgabe

Das Programm zur Evaluation liefert für jede Ausführung eine Textdatei. Die Textdatei wird in dem zu Beginn definierten Pfad gespeichert. Der Dateiname richtet sich nach der ausgewerteten Eingabedatei und wird um das Präfix „eva_“ und einer durchlaufenden Nummer als Suffix erweitert. Durch das Suffix werden vorhandene Dateien bei erneuter Ausführung des Programms mit gleichen Einstellungen nicht überschrieben. Bei der Auswertung werden immer alle möglichen Top N Terme betrachtet. So werden die Metriken zuerst auf die Top 1 Terme, dann die Top 2 Terme und so weiter bis hin zur Maximalanzahl an gefundenen Termen berechnet und in die Ausgabedatei geschrieben. Ein Beispiel für den Inhalt und die Form der ausgegebenen Textdatei ist in Abbildung 4.12 zu finden. Die Werte zu Top 1 Termen wurden manuell aus der Datei entfernt, da alle berechneten Werte „0“ ergaben. Der vorhandene Inhalt wird im Nachfolgenden beschrieben.



```

eva_venu_2-gram_combined_0_0.txt - Editor
Datei Bearbeiten Format Ansicht ?
The algorithm found 974 terms.
The gold standard is 84 terms large.
The highest precision of 0.3333333333333333 is reached by using Top 2 terms.
The highest recall of 0.4041507773540628 is reached by using Top 974 terms.
The highest f1 score of 0.163728865820443 is reached by using Top 319 terms.
=====
====Top N Terms: 2
Precision: 0.3333333333333333
Recall:    0.005952380952380952
F1-Score:  0.011695906432748539
=====
====Top N Terms: 3
Precision: 0.3333333333333333
Recall:    0.007936507936507936
F1-Score:  0.015503875968992246
  
```

Abbildung 4.12: Ausgabedatei der Evaluation

Die erste Zeile gibt an wie viele Terme in der auszuwertenden Eingabedatei vorhanden sind. Darauf folgend werden die Anzahl der Terme aus dem Goldstandard gelistet. Die Zeilen drei bis fünf geben an, bei welcher Top N Termauswahl der Maximalwert für Precision, Recall und F1-Score erreicht wurde. Die folgenden Zeilen geben die berechneten Metriken für alle Top N Terme in aufsteigender Reihenfolge an.

Auf Basis dieser Datei lassen sich die implementierten Ansätze evaluieren.

5 Evaluation

In diesem Kapitel werden die Ergebnisse der implementierten Evaluationsplattform ausgewertet. Für jeden implementierten Ansatz werden dafür zunächst die Metriken Precision, Recall und F1-Score mit einem gegebenen manuell erstellten Goldstandard untersucht. Die Werte hierfür liefert das implementierte Evaluations-Programm. Anschließend wird die Anwendbarkeit im Bezug auf den Implementierungsaufwand sowie die Benutzerfreundlichkeit bewertet. Des Weiteren wird eine Begründungen für den Ausgang der Ergebnisse gegeben und mögliche Ursachen sowie Verbesserungen diskutiert. Im Anschluss dazu werden beide Ansätze direkt miteinander verglichen und ein Fazit gezogen.

5.1 Evaluation von Balachandran et al.

Hier wird die Implementierung des Ansatzes von Balachandran et al. [3] evaluiert. Neben der Anwendbarkeit und aufgetauchten Problemen bei der Umsetzung, werden die Ergebnisse im Detail erläutert und Verbesserungsmöglichkeiten aufgezeigt.

5.1.1 Anwendbarkeit und Probleme

Implementierungsaufwand und Benutzerfreundlichkeit

Der Implementierungsaufwand des von Balachandran et al. vorgestellten Ansatzes ist gering.

Der Ansatz beinhaltet keine komplexen Berechnungen und bedient sich für die Klassifizierung der domänenspezifischen Terme lediglich einem einfachen statistischen Ansatz. Dabei wird geprüft wie häufig ein Term im Haupttext sowie den zwei Kontrasttexten vorkommt. [3]

Durch die ausführliche Dokumentation des verwendeten Stanford Parsers stellt die Implementierung des notwendigen POS-Taggers kein Hindernis dar.

Der Ansatz benötigt zur Ausführung Kontrastkorpora, der nur schwer zu beschaffen ist. Je nach gewünschter Domäne muss dieser neu erstellt werden. Durch die Einbindung des Kontrastkorpora sind bei der Ausführung des Programms zudem viele Benutzereingaben

notwendig. Die Lemmatisierung jedes einzelnen Wortes kostet zudem viel Zeit, was sich in einer höheren Laufzeit bemerkbar macht.

Beschaffung des Kontrastkorpus

Im Gegensatz zur vergleichsweise einfachen Programmierung steht die Beschaffung von geeignetem Kontrastkorpora. Die Anzahl an geeigneten und frei verfügbaren domänen-spezifischen Kontrastkorpora ist gering [3]. Um eine hohe Qualität der Ergebnisse zu liefern, müssen die Kontrastkorpora zudem aus einer anderen Domäne als dem Haupttext stammen und eine hohe lexikalische Vielfalt besitzen [3]. Korpora, in denen viele Begriffe nur wenige male vorkommen, besitzen keine hohe lexikalische Vielfalt und sind somit nicht für diesen Algorithmus geeignet [27]. Der in dieser Bachelor-Arbeit auszuwertende Haupttext ist der Domäne der Informatik zuzuordnen. Die in der Publikation von Balachandran et al. [3] vorgestellten Kontrastkorpora, die nicht aus der Informatik stammen, sind, bis auf den GENIA Korpus (Bio-Medizin) [11] und Reuters (Wirtschaft) [21], nicht ohne Weiteres verfügbar. Der GENIA Korpus wird von Balachandran et al., im Gegensatz zu Reuters, durch eine hohe lexikalische Vielfalt als geeignet klassifiziert. Es kann jedoch kein Korpus mit einer höheren lexikalischen Vielfalt gefunden werden, weshalb der Reuter Korpus in dieser Bachelor-Arbeit weiterhin verwendet wird.

5.1.2 Ergebnisse der Evaluation

Für die Evaluation wird das Programm zum Ansatz von Balachandran et al. mit dem auszuwertenden Haupttext und den Kontrastkorpora GENIA (Bio-Medizin) und Reuters (Wirtschaft) ausgeführt. Da der gegebene Goldstandard nur maximal Bigramme (2-Gramme) beinhaltet, wird das Programm entsprechend eingestellt. Das Programm extrahiert insgesamt 542 Terme aus dem Haupttext. Die höchste Precision wird bei der Betrachtung der Top 35 Terme mit ca. 19,5% erreicht. Bei der Betrachtung aller 542 Terme wird der höchste Recall mit ca. 33,8% erreicht. Der höchste F1-Score wird bei den Top 379 Termen mit ca. 16,4% erreicht. In Abbildung 5.1 sind die Ergebnisse im Detail aufgelistet. Es werden die Werte Precision, Recall und F1-Score für die jeweiligen Top N Terme angegeben. Der Höchstwert ist jeweils fett gedruckt.

Top N Terme	Precision	Recall	F1-Score
10	0,073	0,005	0,009
20	0,133	0,017	0,030
30	0,189	0,035	0,059
40	0,193	0,047	0,079
50	0,185	0,056	0,086
100	0,172	0,103	0,129
200	0,142	0,170	0,155
300	0,126	0,225	0,161
400	0,116	0,277	0,164
500	0,107	0,320	0,161

Abbildung 5.1: Ergebnisse der Evaluation von Balachandran et al.

Die Ergebnisse werden in Abschnitt 5.4 mit den Ergebnissen von Venu et al. verglichen und bewertet.

5.1.3 Verbesserungsmöglichkeiten

Da die Qualität der Ergebnisse stark vom gewählten Kontrastkorpus abhängig ist, kann mit einer geeigneten Auswahl des Kontrastkorpus diese womöglich verbessert werden. Folgendes Beispiel aus den Evaluationsergebnissen stützt diese These:

In den Top 40 der extrahierten Terme ist der Term „quantity“ insgesamt 14 mal vertreten. Beispiele hierfür sind „quantity“, „input quantity“, „material quantity“, „editable quantity“ und viele mehr. Der Term „quantity“ ist kein domänenspezifischer Term des Haupttextes, jedoch kommt er zu selten in den Kontrastkorpora vor und wird dadurch nicht ausreichend abgewertet. Mit einem Kontrastkorpus, der den Term häufig genug beinhaltet, würde der Term stärker abgewertet werden, sodass bessere Ergebnisse erzielt werden könnten. Weitere Verbesserungsmöglichkeiten, die sowohl für dieses als auch das Programm zum Ansatz von Venu et al. gelten, werden im Abschnitt 5.3 diskutiert.

5.2 Evaluation von Venu et al.

Hier wird die Implementierung des Ansatzes von Venu et al. [24] evaluiert. Neben der Anwendbarkeit und den aufgetauchten Problemen bei der Umsetzung, werden die Ergebnisse im Detail erläutert und mögliche Verbesserungsmöglichkeiten aufgezeigt.

5.2.1 Anwendbarkeit und Probleme

Implementierungsaufwand und Benutzerfreundlichkeit

Der Implementierungsaufwand des von Venu et al. vorgestellten Ansatzes ist vergleichsweise hoch. Mit dem HITS-Algorithmus nutzt der Ansatz eine komplexe Berechnung für die Klassifizierung domänenspezifischer Terme [24]. Der Algorithmus beinhaltet Matrix- und Vektormultiplikationen [24], die mit den in JAVA gegebenen Möglichkeiten umgesetzt werden müssen. Zudem stellt die Extraktion von SSR eine weitere Schwierigkeit dar. Diese wird im Anschluss diskutiert.

Nach der Implementierung des Ansatzes ist die Verwendung sehr einfach. Es müssen keine Kontrastkorpora oder ähnliches manuell erstellt werden, um unterschiedliche Domänen zu bedienen. Bei der Ausführung des Programms werden zudem wenige Benutzereingaben benötigt.

Extraktion von Shallow Semantic Relations

Die Beschreibung der Extraktion von SSR ist in der Publikation von Venu et al. und den referenzierten Werken nicht ausreichend beschrieben. Zwar wird in der Publikation von Mukherjee et al. [19] die Extraktion der SSR mit dem *ESG Parser* beschrieben, allerdings unterscheidet sich die Ausgabe des *ESG Parser* deutlich zu der vom *Stanford Parser* gelieferten.

Alle vom *ESG Parser* mit „rel“ (rel = Relations) getaggten Terme werden als SSR identifiziert. Einen vergleichbaren Tag liefert der in dieser Bachelor-Arbeit verwendete *Stanford Parser* nicht. Deshalb werden mithilfe der Beschreibung und den gegebenen Beispielen aus der Publikation von Mukherjee et al. die relevanten Tags des *Stanford Parsers* ermittelt. Die relevanten Tags sind „nmod“, „compound“ und „dobj“ [6]. Aufgrund der mangelhaften Beschreibung ist die Liste womöglich nicht vollständig.

5.2.2 Ergebnisse

Für die Evaluation wird das Programm zum Ansatz von Venu et al. mit dem auszuwertenden Haupttext ausgeführt. Da der gegebene Goldstandard nur maximal Bigramme (2-Gramme) beinhaltet wird das Programm entsprechend eingestellt.

Das Programm extrahiert insgesamt 323 Monogramme (Nomen) und 650 Bigramme (SSR) aus dem Haupttext. Für die weitere Evaluation wird die Kombination aus den extrahierten Monogrammen und Bigrammen verwendet, da der Goldstandard ebenfalls beide Typen beinhaltet. Des Weiteren ist hiermit eine bessere Vergleichbarkeit mit dem Ansatz von Balachandran et al. möglich.

Die höchste Precision wird bei der Betrachtung der Top 2 Terme mit ca. 33,3% erreicht. Bei der Betrachtung aller 973 Terme wird der höchste Recall mit ca 40,4% erreicht. Der höchste F1-Score wird bei den Top 319 Termen mit ca. 16,4% erreicht. In Abbildung 5.2 sind die Ergebnisse im Detail aufgelistet. Es werden die Werte Precision, Recall und F1-Score für die jeweiligen Top N Terme angegeben. Der Höchstwert ist jeweils fett gedruckt.

Top N Terme (SSR & Nomen)	Precision	Recall	F1-Score
10	0,273	0,018	0,034
20	0,224	0,028	0,050
30	0,187	0,035	0,058
40	0,172	0,042	0,067
50	0,175	0,053	0,081
100	0,163	0,098	0,122
200	0,146	0,175	0,159
300	0,127	0,228	0,164
400	0,113	0,270	0,160
500	0,102	0,303	0,152

Abbildung 5.2: Ergebnisse der Evaluation der SSR und Nomen von Venu et al.

Die Ergebnisse werden in Abschnitt 5.4 mit den Ergebnissen von Balachandran et al. verglichen und bewertet.

5.2.3 Verbesserungsmöglichkeiten

Die Extraktion der SSR könnte mit dem in der Publikation von Mukherjee et al. [19] *ESG Parser* verglichen werden, um möglicherweise fehlende Tags zu identifizieren und diese anzupassen. Weitere Verbesserungsmöglichkeiten, die sowohl für dieses als auch das Programm zum Ansatz von Balachandran et al., gelten werden im Abschnitt 5.3 diskutiert.

5.3 Allgemeine Verbesserungsmöglichkeiten

Eine ausführliche Analyse des Haupttextes und des Goldstandards haben Verbesserungsmöglichkeiten aufgezeigt, die die Ergebnisse beider implementierter Ansätze aufwerten

würden.

Der zu analysierende Haupttext ist zum Teil falsch formatiert. Insbesondere fehlende Leerzeichen sorgen beim Ausführen des *Stanford Parsers* zu nicht gewünschten Ergebnissen. Ein Beispiel hierfür ist der nachfolgende Teil eines Satzes: „the material list,RESULTEach row“. Zwischen dem Term „RESULT“ und dem Term „Each“ fehlt ein Leerzeichen. Der Satz kann durch den *Stanford Parser* nicht richtig aufgelöst werden. An anderer Stelle macht sich das Fehlen von Leerzeichen stärker bemerkbar und hat direkten Einfluss auf die Qualität der Ergebnisse. So sind beispielsweise die Terme „PMR“ und „PMT“ als wichtige Terme im Goldstandard genannt, im Text jedoch nur in folgendem Format vorhanden: „PMT/PMR“. Der *Stanford Parser* erkennt diese Zeichenfolge nicht als die Nomen „PMT“ und „PMR“, sondern als Adjektiv „PMT/PMR“. Dadurch können diese Terme von keinem der implementierten Ansätze gefunden werden. Deshalb wird der Haupttext manuell angepasst, um die Auswirkungen zu evaluieren. Beide Ansätze finden dadurch neun Terme mehr. Die durchschnittlichen Werte für Precision, Recall und F1-Score erhöhen sich dadurch um weniger als 0,5%.

Des Weiteren sind insgesamt 12 der 84 Terme aus dem Goldstandard nicht im Haupttext vorhanden. Beispiele hierfür sind „buying group“, „contract model“, „disposal frequency“ und „frozen handling“ um nur ein paar zu nennen. Diese Terme können von den implementierten Ansätzen nicht gefunden werden und verschlechtern damit die Werte Precision, Recall und F1-Score. Zudem scheint der Goldstandard nicht vollständig zu sein. Einige von den Ansätzen gefunden Terme, die nicht im Goldstandard stehen, würde man der Domäne der Informatik zuordnen. Eine Anpassung des Goldstandards würde daher zu besseren Ergebnissen führen.

5.4 Fazit der Evaluation

Beide implementierten Ansätze können nicht mit der, in den jeweiligen Publikation erreichten, Precision mithalten. Venu et al. bewertet den Ansatz in der Publikation mit durchschnittlich ca. 72,7%, wohingegen die Implementierung dieser Bachelor-Arbeit im Durchschnitt eine Precision von 16,8% erreicht. Auch in der Publikation von Balachandran et al. wird der Ansatz mit einer Precision von ca. 62,4% deutlich höher als die hier erreichten ca. 14,4%, ausgewiesen. Die gravierenden Unterschiede sind jedoch auch dadurch zu erklären, dass bei den Publikationen für jeden Term entschieden wird ob er relevant ist oder nicht und kein zuvor gegebener Goldstandard getroffen werden muss. Die unter Abschnitt 5.3 genannten Verbesserungsmöglichkeiten können die große Differenz der Werte zudem verkleinern.

Die Tendenz, die aus beiden Publikationen bereits zu erkennen war, wurde durch diese

Evaluation bestätigt. Der Ansatz von Venu et al. liefert im Durchschnitt die minimal besseren Werte bei Precision, Recall und F1-Score als Balachandran et al.. Ab den Top 50 Termen unterscheidet sich die Precision beider Ansätze um weniger als 1% (siehe Abbildung 5.3).

Top N Terme	Venu	Balachachandran	Differenz
10	0,273	0,073	0,200
20	0,224	0,133	0,091
30	0,187	0,189	0,002
40	0,172	0,193	0,021
50	0,175	0,185	0,010
100	0,163	0,172	0,009
200	0,146	0,142	0,004
300	0,127	0,126	0,001
400	0,113	0,116	0,003
500	0,102	0,107	0,005
∅	0,168	0,144	

Abbildung 5.3: Vergleich der Precision von Venu et al. und Balachandran et al.

Die größeren Werte sind jeweils fett gedruckt. Der höhere Durchschnittswert bei Venu et al. kommt vor allem durch die hohe Precision innerhalb der ersten Top 20 Terme zustande. Die Differenz zeigt den Unterschied beider Ansätze als Absolutbetrag.

Durch die Unabhängigkeit zu anderen Korpora ist der Ansatz von Venu et al. stabiler und kann auf jede Domäne ohne Anpassung angewendet werden. Das implementierte Programm ist in der Ausführung mit 71,3 Sekunden zu 153,3 Sekunden (Balachandran et al.) zudem um den Faktor zwei schneller.

Um die geringen Werte bei Precision, Recall und F1 letztendlich zu verifizieren wird zusätzlich mit Programm NaCTeM TerMine [13] ein Online verfügbarer Ansatz evaluiert. Hierzu wird der Haupttext durch die TerMine Web Demonstration analysiert und die Ergebnisse durch das Evaluations-Programm ausgewertet. TerMine verwendet die C-Value Methode, die auch in der Publikation von Balachandran et al. [3] zum Vergleich verwendet wird. Diese Methode liefert nur komplexe Terme (keine Monogramme), dies wird bei der Evaluation bedacht und soll lediglich als Anhaltspunkt dienen. Balachandran et al. gibt den C-Value Ansatz mit einer durchschnittlichen Precision von 30% an. Mit dem gegebenen Haupttext und Goldstandard liefert die Online-Demo von TerMine jedoch lediglich eine Precision von ca. 8%. Damit ist gezeigt, dass die Implementierungen

legitime Ausgaben liefern und die Höhe der evaluierten Werte am Haupttext und dem Goldstandard liegen muss.

6 Zusammenfassung und Ausblick

6.1 Zusammenfassung

Für diese Bachelor-Arbeit sollten die existierenden Ansätze zur domänenspezifischen Termextraktion identifiziert und die geeignetsten davon verglichen werden. Die Literaturrecherche führte zu einer hohen Anzahl von Ansätzen, die zunächst durch festgelegte Kriterien minimiert wurden. Es blieben die Ansätze von Balachandran et al. und Venu et al. übrig. Im Rahmen der Literaturrecherche wurden die Forschungsfragen beantwortet (3.5.1) und die ausgewählten Ansätze erläutert (3.5.2).

Im Anschluss wurde die Evaluationsplattform beschrieben und die Implementierung der Algorithmen erläutert (4). Diesbezüglich wurden auch notwendige Anpassungen der Ansätze, die Bedienung und Ausgabe beschrieben. Die darauffolgende Evaluation der implementierten Ansätze (5) stellte den Kern dieser Bachelor-Arbeit dar. Es zeigte sich, dass die Ergebnisse nicht mit der, in den jeweiligen Publikationen geschilderten Qualität mithalten konnten. Hierfür wurden neben der Anpassung des Goldstandards weitere Verbesserungsmöglichkeiten genannt.

Das Fazit der Evaluation (5.4) stellte die Ergebnisse beider Ansätze in Vergleich. Zudem wurde mit Hilfe eines bereits bestehenden Ansatzes gezeigt, dass die niedrigen Werte bei Precision, Recall und F1-Score legitim sind. Der Ansatz von Venu et al. wurde mit leichten Vorteilen in Precision, Recall und F1, durch die höhere Benutzerfreundlichkeit und kürzere Laufzeit zum besseren domänenspezifischen Termextraktor ausgewählt.

6.2 Ausblick

Das Gebiet der domänenspezifischen Termextraktion ist ein sehr aktuelles Forschungsgebiet. Bereits während der Erstellung dieser Bachelor-Arbeit könnten neue Ansätze entwickelt worden sein, die ein besseres Ergebniss und einfachere Benutzung ermöglichen. Zumindest aber im Laufe der Zeit werden neue Ansätze vorgestellt, die es zu vergleichen gilt. Diese Evaluationsplattform kann durch diese Ansätze erweitert werden und dabei helfen eine Vergleichsmöglichkeit zu schaffen. Zudem können nach Umsetzung

der in Abschnitt 5.3 vorgestellten Verbesserungsmöglichkeiten der Haupttext neu analysiert werden um bessere Ergebnisse zu erzielen. Diese Bachelor-Arbeit soll zudem im Rahmen einer Master-Arbeit zu diesem Thema aufgegriffen werden.

Literaturverzeichnis

- [1] Akiko Aizawa. An information-theoretic perspective of tf-idf measures. *Information Processing & Management*, 39(1):45–65, 2003.
- [2] James F Allen. Natural language processing. 2003.
- [3] Kiruparan Balachandran and Surangika Ranathunga. Domain-specific term extraction for concept identification in ontology construction. In *Web Intelligence (WI), 2016 IEEE/WIC/ACM International Conference on*, pages 34–41. IEEE, 2016.
- [4] Noah Bubenhofer. 4. kollokationen, n-gramme, mehrworteinheiten. *Handbuch Sprache in Politik und Gesellschaft*, 19:69, 2017.
- [5] Niladri Sekhar Dash. *Corpus linguistics: An introduction*. Pearson Education India, 2008.
- [6] Marie-Catherine De Marneffe and Christopher D Manning. Stanford typed dependencies manual. Technical report, Technical report, Stanford University, 2008.
- [7] Euthymios Drymonas, Kalliopi Zervanou, and Euripides GM Petrakis. Unsupervised ontology acquisition from plain texts: The ontogain system. In *NLDB*, pages 277–287. Springer, 2010.
- [8] Hans-Dieter Ehrich, Martin Gogolla, and Udo Walter Lipeck. *Algebraische Spezifikation abstrakter Datentypen: eine Einführung in die Theorie*. Springer-Verlag, 2013.
- [9] Katerina Frantzi, Sophia Ananiadou, and Junichi Tsujii. The c-value/nc-value method of automatic recognition for multi-word terms. *Research and advanced technology for digital libraries*, pages 520–520, 1998.
- [10] Sanda Harabagiu, Cosmin Adrian Bejan, and Paul Morarescu. Shallow semantics for relation extraction. In *Proceedings of the 19th international joint conference on Artificial intelligence*, pages 1061–1066. Morgan Kaufmann Publishers Inc., 2005.

- [11] J-D Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. Genia corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl_1):i180–i182, 2003.
- [12] Jon M Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.
- [13] V Kosa, D Chaves-Fraga, D Naumenko, E Yuschenko, C Badenes-Olmedo, V Ermolayev, and A Birukou. Cross-evaluation of automated term extraction tools. Technical report, Technical Report TS-RTDC-TR-2017-1, 30.09. 2017, Dept of Computer Science, Zaporizhzhia National University, Ukraine, 2017.
- [14] Krister Lindén, Jussi Olavi Piitulainen, et al. Discovering synonyms and other related words. In *Proceedings of COLING 2004 CompuTerm 2004: 3rd International Workshop on Computational Terminology*, 2004.
- [15] Juan Antonio Lossio-Ventura, Clement Jonquet, Mathieu Roche, and Maguelonne Teisseire. Yet another ranking function for automatic multiword term extraction. In *International Conference on Natural Language Processing*, pages 52–64. Springer, 2014.
- [16] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*, pages 55–60, 2014.
- [17] Diana Maynard, Yaoyong Li, and Wim Peters. Nlp techniques for term extraction and ontology population., 2008.
- [18] Kevin Meijer, Flavius Frasincar, and Frederik Hogenboom. A semantic approach for extracting domain taxonomies from text. *Decision Support Systems*, 62:78–93, 2014.
- [19] Subhabrata Mukherjee, Jitendra Ajmera, and Sachindra Joshi. Domain cartridge: Unsupervised framework for shallow domain ontology construction from corpus. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 929–938. ACM, 2014.
- [20] David Martin Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. 2011.

- [21] Tony Rose, Mark Stevenson, and Miles Whitehead. The reuters corpus volume 1-from yesterday's news to tomorrow's language resources. In *LREC*, volume 2, pages 827–832, 2002.
- [22] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.
- [23] Carmen Scherer. *Korpuslinguistik*. Winter Heidelberg, 2006.
- [24] Sree Harissh Venu, Vignesh Mohan, Kodaikkaavirinaadan Urkalan, and TV Geetha. Unsupervised domain ontology learning from text. In *International Conference on Mining Intelligence and Knowledge Exploration*, pages 132–143. Springer, 2016.
- [25] Claes Wohlin. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, page 38. ACM, 2014.
- [26] Wilson Wong, Wei Liu, and Mohammed Bennamoun. Ontology learning from text: A look back and into the future. *ACM Computing Surveys (CSUR)*, 44(4):20, 2012.
- [27] Martin Wynne. *Developing linguistic corpora: A guide to good practice*, volume 92. Oxbow Books Oxford, 2005.