Universität Leipzig
Wirtschaftswissenschaftliche Fakultät
Institut für Wirtschaftsinformatik
Professur Anwendungssysteme

Masterarbeit

Methoden des Data-Minings zur Plagiatanalyse studentischer Abschlussarbeiten

Betreuender Hochschullehrer:	Prof. Dr. U. Eisenecker
Betreuender Assistent:	M. Sc. D. Baum
Bearbeiter:	Johann-David Märker

Eingereicht am: 24.07.2018

Abstract

Autor
Johann-David Märker

Inhalt
Bestehende Ansätze der automatisierten Plagiatanalyse nutzen umfangreiche und pflegeaufwändige Referenzkorpora oder greifen ausschließlich auf die im Untersuchungsobjekt enthaltenen Informationen zurück. Die Nutzung externer Daten führt in der Regel zu besseren Analyseergebnissen (vgl. [Tschuggnall 2014, 8]). In der vorliegenden Arbeit wurde ein extrinsisches Verfahren zur Plagiatanalyse studentischer Abschlussarbeiten entwickelt und evaluiert, welches einen begrenzten Trainingsdatensatz als Referenzkorpus nutzt. Das genannte Verfahren greift hierbei auf die Methoden der Dokumenttypklassifikation und der Stilometrie zurück. Entspricht ein Abschnitt des Eingabedokuments nicht dem durchschnittlichen Schreibstil einer studentischen Abschlussarbeit, so wird dieser als potentielles Plagiat markiert. Anhand verschiedener Evaluationsschritte konnte gezeigt werden, dass das Verfahren prinzipiell für die Plagiatanalyse studentischer Abschlussarbeiten geeignet ist. Im simulierten Anwendungskontext konnten 71,03 % der Segmente aus Bachelor- und Masterarbeiten sowie 53,62 % der Segmente aus Fachbüchern, Fachartikeln und Wikipediaartikeln korrekt eingeordnet werden. Der erreichte F_1-Wert entspricht der Performanz intrinsischer Verfahren. Der erzielte *Recall*-Wert ist hierbei wesentlich höher. Die aus den Trainingskorpora extrahierten *features* wurden als ARFF-Dateien zur Verfügung gestellt.

Schlüsselwörter
Data-Mining, Dokumenttypklassifikation, Plagiatanalyse, Stilometrie

Datum
24.07.2018

Gliederung

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

8F	8-Features
ARFF	Attribute-Relation File Format
CLPD	cross language plagiarism detection
CRISP-DM	Cross-Industry Standard Process for Data Mining
FN	false negatives
FP	false positives
HTML	Hypertext Markup Language
JGAAP	Java Graphical Authorship Attribution Program
JSON	JavaScript Object Notation
NB	Naive Bayes
NLP	natural language processing
OCR	optical character recognition
PAN-PC-10	PAN Plagiarism Corpus 2010
PAN-PC-11	PAN Plagiarism Corpus 2011
PDF	Portable Document Format
POS	part of speech
SBD	sentence boundary detection
SVM	support vector machine
TF-IDF	term frequency inverse document frequency
TN	true negatives
TP	true positives
WEKA	Waikato Environment for Knowledge Analysis
WP	Writeprints
XML	Extensible Markup Language

1 Einleitung

Studien zeigen, dass ein großer Teil der Studierenden an Hochschulen im Verlauf des Studiums plagiiert (vgl. [Gipp 2014, 336ff]). Die zunehmende Verbreitung von Informations- und Kommunikationstechnologien erleichtert die Anfertigung von Plagiaten (vgl. [Gipp 2014, 1]) und erschwert die Prüfung verdächtiger Dokumente. Insbesondere im Bereich der akademischen Lehre werden daher sogenannte Plagiatscanner eingesetzt.

Im Kontext der automatisierten Plagiatanalyse wird zwischen intrinsischen und extrinsischen Verfahren unterschieden. Erstere arbeiten ohne externe Daten und nutzen ausschließlich die im Untersuchungsobjekt enthaltenen Informationen, während letztere auf einen Referenzkorpus bestehender Literatur zurückgreifen. Durch die Nutzung von Referenzdaten erzielen externe Verfahren in der Regel bessere Ergebnisse bei der Erkennung von Plagiaten (vgl. [Tschuggnall 2014, 8]). Bei bestehenden extrinsischen Verfahren erfolgt die Zuordnung der Quelle eines potentiellen Plagiats auf Basis konkreter Dokumente. Der genutzte Referenzkorpus muss Bezugsdaten für jede in der Ergebnismenge vertretene Ausprägung enthalten. Da die Anzahl existierender Dokumente stetig wächst, ist die Vollständigkeit eines entsprechenden Korpus nur mit hohem Aufwand sicherzustellen.

In der vorliegenden Arbeit wird ein extrinsisches Verfahren entwickelt und evaluiert, welches potentielle Plagiate in studentischen Abschlussarbeiten auf Basis des zugeordneten Dokumenttyps bestimmt. Im Zuge des Entwicklungsprozesses wird überprüft, inwiefern sich Dokumenttypen wie Bachelorarbeiten, Fachartikel und Wikipediaartikel anhand der stilistischen Eigenschaften eines Textausschnitts automatisiert unterscheiden lassen. Der hierfür genutzte Trainingskorpus ist entsprechend der Anzahl möglicher Typzuordnungen begrenzt und bedarf daher keiner regelmäßigen Erweiterung.

In den folgenden Kapiteln werden zunächst grundlegende Konzepte des Data-Minings erläutert. Anschließend werden bestehende Ansätze der automatisierten Plagiatanalyse vorgestellt und im Kontext der Dokumenttypklassifikation beleuchtet. Die Entwicklung und Evaluation des angesprochenen Verfahrens erfolgt nach dem *Cross-Industry Standard Process for Data Mining* (CRISP-DM). Hierbei werden Problemdefinition, Datenexploration, Datenvorverarbeitung, Modellierung, Evaluation und Bereitstellung als einzelne Prozessphasen betrachtet (vgl. [Witten et al. 2017a, 29]).

2 Grundlagen

2.1 Data-Mining

Das in der vorliegenden Arbeit untersuchte Verfahren klassifiziert Segmente eines Eingabedokuments nach Dokumenttypen. Hierbei werden Verfahren des Data-Minings und des maschinellen Lernens genutzt. Letztere werden in den folgenden Kapiteln erläutert.

2.1.1 Data-Mining und maschinelles Lernen

Durch die zunehmende Digitalisierung vieler Geschäfts- und Alltagsprozesse steigt die Menge der durch Computersysteme erfassten Daten (vgl. [Han et al. 2012, 1ff]). Die Auswertung der genannten Datenmenge ist aufgrund ihres Umfangs und der Heterogenität der enthaltenen Daten jedoch schwierig. Han et al. fassen die beschriebene Situation als einen Reichtum an Daten und eine Armut an Informationen zusammen (vgl. [Han et al. 2012, 5]). Die Prozesse des Data-Minings tragen zur Lösung der erläuterten Problematik bei und suchen in großen Datenmengen nach verwertbaren Mustern oder Tendenzen (vgl. [Larose 2014, 2]).

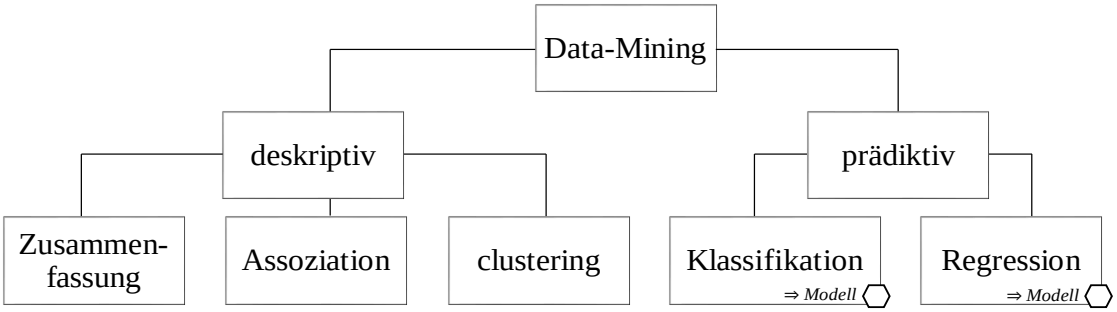


Abbildung 1: Aufgaben des Data-Minings (in Anlehnung an [Kantardzic 2009, 2f; Alpaydin 2010, 5ff])

Abbildung 1 zeigt verschiedene Aufgabenstellungen, die mit jeweils unterschiedlichen Data-Mining-Algorithmen bearbeitet werden. Anhang A ordnet die genannten Algorithmen in einer Taxonomie. Durch deskriptive Verfahren können bestehende Daten in eine für Menschen auswertebare Form überführt werden. Die hierbei aufgezeigten Muster dienen einem besseren Verständnis der Daten. Durch prädiktive Verfahren soll hingegen ein Modell in Form einer Abbildungsvorschrift entwickelt werden, auf dessen Basis unbekannte oder zukünftige Daten beurteilt werden (vgl. [Kantardzic 2009, 2f]).

Durch Algorithmen, welche für die Zusammenfassung von Daten entwickelt wurden, können Informationen in eine kompaktere Darstellung überführt werden. Typische Anwendungsbeispiele sind automatisch generierte Visualisierungen oder Geschäftsberichte

(vgl. [Chandola/Kumar 2007, 355]). Durch Assoziationsanalysen können Attribute aufzeigt werden, die häufig gemeinsam auftreten. Im betrieblichen Kontext werden die genannten Analysen im Rahmen der *market basket analysis* genutzt, um Waren zu identifizieren, die gemeinsam gekauft werden (vgl. [Larose 2014, 14]). *Clustering*-Verfahren gruppieren Dateneinträgen so, dass möglichst homogene Gruppen entstehen, welche untereinander möglichst verschieden sind und werden beispielsweise zur Visualisierung von Strukturen oder Entwicklung von Ontologien genutzt. Im Gegensatz zu Klassifikationsverfahren gibt es hierbei keine Zielwerte beziehungsweise Klassenvorgaben. Die erstellten Gruppen werden ausschließlich auf Grundlage der Attributwerte erstellt (vgl. [Larose 2014, 12]). Klassifikationsverfahren ordnen hingegeben einem Attributwert einen konkreten Zielwert beziehungsweise eine Klasse zu. Die hierfür notwendige Abbildungsvorschrift wird aus vorgegebenen Zuordnungen abgeleitet. Klassifikationsverfahren werden zum Beispiel genutzt, um Betrugsfälle bei Finanztransaktionen zu erkennen (vgl. [Larose 2014, 10]). Kapitel 2.1.3 erläutert die Methoden der Klassifikation im Detail. Die Methoden der Regression ähneln jenen der Klassifikation. Statt konkreter Klassenausprägungen soll jedoch ein stetiger, numerischer Zielwert zu jedem Attributwert zugeordnet werden. Regressionsverfahren werden beispielsweise genutzt, um Aktien- oder Grundstückswerte für zukünftige Zeitpunkte abzuschätzen (vgl. [Kantardzic 2009, 3]).

Wie bereits beschrieben, ist die Zielstellung prädiktiver Verfahren die Erstellung eines Modells. Letzteres kann hierbei beispielsweise eine Abbildungsvorschrift in Form einer Funktion oder eine Abbildungsvorschrift in Form von Programmanweisungen sein (vgl. [Kantardzic 2009, 2f]). Werden Programme automatisiert anhand von Daten erstellt, so entspricht dies dem Prozess des maschinellen Lernens (vgl. [Domingos 2012, 78]). Verfahren des maschinellen Lernens werden insbesondere dann genutzt, wenn vorhandenes, auf Erfahrung basierendes Wissen durch Menschen nur schwer in explizite Regeln übersetzt werden kann (vgl. [Alpaydin 2010, XXXI]). Maschinelles Lernen wird beispielsweise in den Domänen der Sprach- und Bilderkennung eingesetzt und hat in den vergangenen Jahren durch die erhöhte Datenverfügbarkeit und die Entwicklung performanterer Hardware bedeutende Fortschritte gemacht (vgl. [Osoba/Davis 2018, 5ff]).

Algorithmen des maschinellen Lernens können für alle der in Abbildung 1 dargestellten Data-Mining-Aufgabenstellungen genutzt werden. Die hierbei entwickelten Modelle beziehungsweise Programme sind jedoch nur im Rahmen von prädiktiven Analysen als Abbildungsvorschrift zu verstehen (vgl. [Kantardzic 2009, 2f]). Typische Aufgabenstellungen des maschinellen Lernens sind *clustering*, Klassifikation und Regression. Ersteres zählt zu

den Anwendungen des unüberwachten Lernens, welches einen Lernvorgang ohne Zielvorgaben beziehungsweise Zielvariablen bezeichnet (vgl. [Russell/Norvig 2016, 528ff]). Ausschließlich die in den Attributausprägungen vorhandenen Strukturen werden genutzt und von strukturloser Information unterschieden (vgl. [Ghahramani 2004, 74]). Klassifikations- und Regressionsverfahren werden den Methoden des überwachten Lernens zugeordnet. Zielstellung ist hierbei, auf Grundlage gegebener Attribut- und Zielwertzuordnungen eine Abbildungsvorschrift in Form eines Programms zu entwickeln, die jedem Attributwert einen Zielwert zuordnet (vgl. [Russell/Norvig 2016, 528ff]).

Das im Rahmen dieser Arbeit untersuchte Verfahren ordnet Textausschnitte anhand ihrer Eigenschaften vorgegebenen Dokumenttypklassen zu. Das folgende Kapitel beschreibt verschiedene Vorverarbeitungsschritte, die zur Erfassung der Eigenschaften beziehungsweise der Attributwerte textueller Daten notwendig sind.

2.1.2 Text Mining und Natural Language Processing

Text mining ist eine spezialisierte Form des Data-Minings und wird zur Analyse von Mustern in semi- oder unstrukturierten, textuellen Daten genutzt (vgl. [Aggarwal/Zhai 2012, 2f]). Methoden des *text mining* werden beispielsweise zur Erkennung von Spam-E-Mails oder zur Stimmungsanalyse in sozialen Netzwerken verwendet (vgl. [Aggarwal/Zhai 2012, 164f; Pak/Paroubek (2010), 1]). Im Vergleich zu anderen Formen des Data-Minings ist die Vorverarbeitung der Eingabedaten sehr aufwändig. Bevor eine Musteranalyse erfolgen kann, müssen die unstrukturierten Daten durch einen als *Informationsextraktion* bezeichneten Prozess in eine strukturierte Form überführt werden (vgl. [Aggarwal/Zhai 2012, 11]). Die zur Abbildung von Texten genutzten Repräsentationsformen sind durch ihre hohe Dimensionalität geprägt (vgl. [Aggarwal/Zhai 2012, 2f]).

Eine einfache, aber häufig genutzte Form der Repräsentation ist die Speicherung als *bag of words* (vgl. [Aggarwal/Zhai 2012, 167]). Hierbei wird zunächst ein Vokabular definiert, welches die Anzahl möglicher Wortausprägungen bestimmt. Jede mögliche Wortausprägung entspricht einer Dimension im Repräsentationsvektor, der für jedes Textdokument erstellt wird. Die im Textdokument enthaltenen Wörter werden anschließend ihrer Dimension im Repräsentationsvektor zugeordnet, um für diese einen passenden Wert zu hinterlegen. In der Regel wird hierbei der *Term-Frequency-Inverse-Document-Frequency*-(TF-IDF)-Wert genutzt. Letzterer gewichtet die Häufigkeit, mit der ein einzelnes Wort im Dokument auftritt, in Bezug auf die Häufigkeit des Auftretens im Korpus aller erfassten Textdokumente (vgl.

[Gipp 2014, 22ff]). Spezielle Wörter charakterisieren einen Text somit stärker als gewöhnliche Wörter (vgl. [Hariharan et al. 2010, 499f]). Zur Definition des Vokabulars und für die Zuordnung einzelner Wörter zu ihrer Dimension im Repräsentationsvektor werden unter anderem *Stemming*-Verfahren eingesetzt. *Stemming*-Verfahren führen das betrachtete Wort auf seinen Wortstamm zurück (vgl. [Kumar 2012, 17f]). Die beschriebene Repräsentation als *bag of words* wird im Rahmen von *Document-Retrieval*-Systemen als *Vektorraummodell* bezeichnet und findet insbesondere im Kontext von Suchmaschinen Anwendung (vgl. [Gipp 2014, 22ff; Manning et al. 2008, 120]). Das erläuterte Verfahren ist in Bezug auf die Verarbeitungsgeschwindigkeit und die genutzten Hardwareressourcen effizient. Die erzeugte Textrepräsentation ermöglicht jedoch keine Auswertung syntaktischer oder semantischer Eigenschaften des Textes, da die Reihenfolge der Wörter im Textdokument nicht erfasst wird. Moderne *Text-Mining*-Verfahren nutzen daher komplexere Verfahren der Informationsextraktion (vgl. [Aggarwal/Zhai 2012, 9]).

Natural language processing (NLP) wird im Kontext der Informationsextraktion aus Textdokumenten eingesetzt und nutzt Erkenntnisse aus den Forschungsgebieten der Informatik und Linguistik. Das zentrale Thema ist hierbei die Interaktion zwischen Computern und menschlicher Sprache. Man unterscheidet zwischen *natural language generation systems* und *natural language understanding systems*. Erstere konvertieren die durch Computer genutzten Sprachrepräsentationen in menschliche Sprache, während letztere menschliche Sprache in eine computergeeignete Darstellung überführen (vgl. [Kumar 2012, 1]). Im Rahmen der vorliegenden Arbeit ist ein NLP-System als ein *natural language understanding system* definiert. Zur Verarbeitung von Textdokumenten erfolgt zunächst eine morphologische Analyse. Jedes im Dokument enthaltene Wort wird hierbei im Rahmen des bereits genannten *stemming* auf seinen Wortstamm zurückgeführt. Vorhandene Flexionen, sowie Präund Suffixe können jedoch erfasst und für spätere Auswertungen gespeichert werden. In der anschließenden lexikalischen Analyse werden die Wortarten jedes Wortes auf Basis eines hinterlegten Lexikons ermittelt. Der beschriebene Vorgang wird als *part of speech* (POS) *tagging* bezeichnet (vgl. [Mitkov 2009, 219]). Substantive erhalten beispielsweise den *POS tag* „NN“ (vgl. [Schiller et al. 1999, 6ff]). Die anschließende syntaktische Analyse untersucht auf Basis definierter grammatikalischer Regeln die Beziehungen der Wörter untereinander. Hierfür müssen die Grenzen jedes Satzes im Dokument ermittelt werden. Der als *sentence boundary detection* (SBD) bezeichnete Prozess der Satzerkennung hat großen Einfluss auf die Performanz des NLP-Systems (vgl. [Wang/Huang 2003, 1]) und wird in Kapitel 3.5.3 näher erläutert. Eine semantische Analyse ermöglicht die Erfassung der Bedeutung der

Wörter auf inhaltlicher Ebene (vgl. [Kumar 2012, 17f]). Hierzu werden umliegende Strukturen betrachtet und Datenbanken wie *SemRep* genutzt, welche Domänenwissen als semantisches Netz abbilden (vgl. [Rindflesch/Fiszman 2003, 468]).

Die durch NLP-Systeme aus dem Text extrahierten Eigenschaften können anschließend direkt genutzt oder im Rahmen des *feature engineering* zur Erstellung komplexerer *features* verwendet werden. Ein *feature* beschreibt jene Eigenschaften eines Objekts, welche besonders relevant sind, um letzteres in Bezug auf andere Objekte zu beschreiben oder zu erkennen (vgl. [Jain et al. 1995, 460]). *Feature engineering* bezeichnet die Erstellung oder das Finden von *features*, die im Rahmen der Zielstellung hilfreich sind (vgl. [Reese et al. 2017, 363]). *Feature engineering* ist ein komplexer Prozess, der umfangreiches Domänenwissen erfordert (vgl. [Culotta/Sorensen (2004), 2]). Im Kontext von *Text-Mining*-Vorhaben werden die durch NLP-Systeme extrahierten Texteigenschaften weiterverarbeitet und verknüpft, um komplexere *features* des Textes abzubilden (vgl. [Liu/Motoda 1998, 4]). Im Rahmen der vorliegenden Arbeit werden *feature sets* aus dem Forschungsgebiet der Stilometrie genutzt. Ein *feature set* bezeichnet eine Sammlung definierter *features*. Der Prozess der *feature selection* wird hingegen als das Auswählen besonders relevanter *features* innerhalb einer vordefinierten Auswahl beschrieben. Hierfür werden beispielsweise Entropieberechnungen oder Korrelationsmaße wie *Chi-Quadrat* genutzt (vgl. [Aggarwal/Zhai 2012, 167ff]).

Das folgende Kapitel erläutert die Relevanz von *features* im Kontext von Klassifikationsverfahren.

2.1.3 Klassifikation und Dokumenttypklassifikation

Wie bereits beschrieben wurde, wird im Rahmen von Klassifikationsverfahren ein Modell entwickelt, welches Attributwerten einen Zielwert beziehungsweise eine Klasse zuweist (vgl. [Larose 2014, 10]). Abbildung 2 zeigt das hierbei angewendete Vorgehen. Auf Basis eines Trainingsdatensatzes mit vorhandenen Klassenzuordnungen, im Beispiel die Klassen *No* und *Yes*, erstellt ein Klassifikationsalgorithmus ein Klassifikationsmodell. Hierfür werden die *features*, also die Eigenschaften der gegebenen Instanzen, in Verbindung mit ihrer Klassenzuordnung betrachtet und durch einen Vorgang des maschinellen Lernens zur Entwicklung einer Abbildungsvorschrift genutzt. Durch Anwendung des erstellten Modells können anschließend Instanzen ohne Klassenzuordnungen klassifiziert werden (vgl. [Aggarwal 2015, 2]).

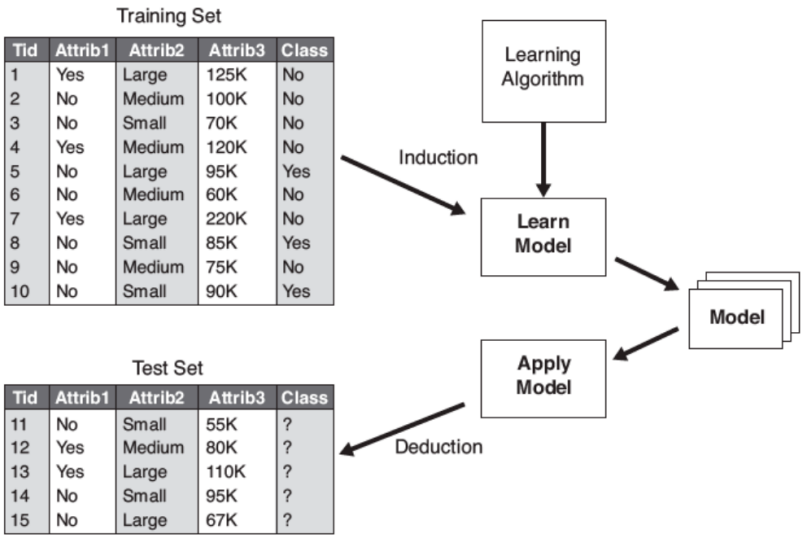


Abbildung 2: Umsetzung eines Klassifikationsvorhabens [Tan et al. 2009, 148]

Für die Entwicklung des Modells können verschiedene Klassifikationsalgorithmen genutzt werden. Anhang A listet und ordnet eine Auswahl letzterer in einer Taxonomie. In der Regel wird zwischen entscheidungsbaumbasierten, regelbasierten, probabilistischen, funktionsbasierten und entfernungsbasierten Klassifikationsalgorithmen unterschieden (vgl. [Aggarwal/Zhai 2012, 176ff]). Kapitel 3.6.6 erläutert die im Kontext dieser Arbeit genutzten Algorithmen im Detail.

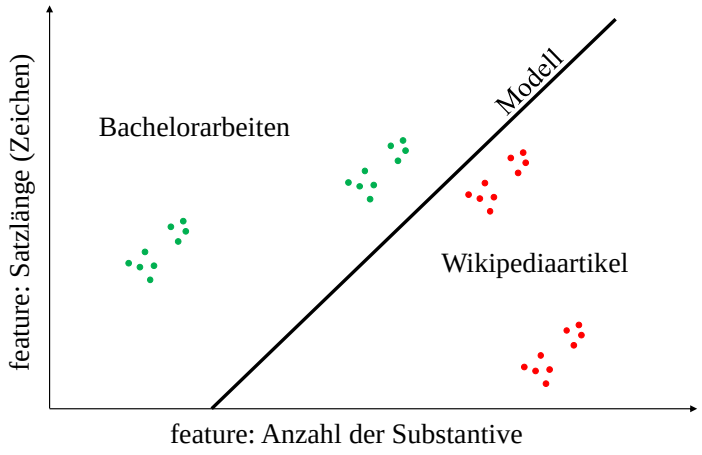


Abbildung 3: Vorgehen eines Klassifikationsalgorithmus (in Anlehnung an [Kantardzic 2009, 107])

Abbildung 3 zeigt das allgemeine Vorgehen eines Klassifikationsalgorithmus an zwei Beispielklassen. Jeder dargestellte Punkt repräsentiert ein Textsegment beziehungsweise eine Eingabeinstanz. Auf Grundlage der farbig dargestellten Klassenzuordnungen und der erfassten *features*, im Beispiel die Satzlänge und die enthaltene Anzahl an Substantiven, versucht der Algorithmus zwei Mengen an Instanzen zu bilden. Funktionsbasierte Algorithmen ermitteln hierfür die in der Grafik als Linie dargestellte Unterteilung. Andere Algorithmen

verwenden anderen Methoden zur Klassentrennung. Im Beispiel werden zwei Dimensionen beziehungsweise zwei *features* für die Klassifikation berücksichtigt. Das stilometrische *feature set* „Writeprints“ erfasst hingegen mindestens 327 *features* (vgl. [Abbasi/Chen 2008, 14]). Kapitel 3.6.5 erläutert die verwendeten *feature sets* im Detail.

Die Dokumenttypklassifikation wird als Teilgebiet der Textklassifikation dem *text mining* zugeordnet (vgl. [Aggarwal/Zhai 2015, 288f]). Sie wird beispielsweise genutzt, um Dokumenttypen des betrieblichen Umfelds wie E-Mails, Notizen oder Berichte automatisiert zu unterscheiden (vgl. [Taghva/Vergara (2008), 180f]). Taghva und Vergara analysieren hierbei neben lexikalischen Eigenschaften auch die verwendeten Schriftarten sowie strukturelle Eigenschaften der Dokumente. Caragea et al. entwickeln ein Verfahren zur Dokumenttypklassifikation, welches durch Bibliotheken genutzt werden kann. Dokumenttypen wie Bücher oder Fachzeitschriften werden hierbei automatisiert anhand von *Portable-Document-Format*-(PDF)-Dateien klassifiziert (vgl. [Caragea et al. 2016, 3998f]). Anhang B zeigt einen Ausschnitt der Onlinesuche der Universitätsbibliothek Leipzig. Suchergebnisse können durch die dargestellte Auswahl nach Dokumenttypen gefiltert werden.

Im Rahmen der vorliegenden Arbeit werden verschiedene Dokumenttypen, die im Kontext studentischer Abschlussarbeiten relevant sind, automatisiert unterschieden. Die hierbei verwendeten *feature sets* nutzen zur Erfassung stilistischer Eigenschaften Erkenntnisse aus dem Forschungsgebiet der Stilometrie.

2.2 Stilometrie

Stilometrie ist dem Forschungsgebiet der quantitativen Linguistik zuzuordnen (vgl. [Köhler et al. 2005, V]) und beschäftigt sich mit der Quantifizierung von Schreibstilen durch statistische Methoden (vgl. [Gipp 2014, 30; Holmes 1998, 111]). Fobbe definiert Stil in Anlehnung an Sandig (vgl. [Sandig 2006, 53]) als „eine Folge der bewussten und unbewussten sprachlichen Entscheidungen des Emittenten und damit eine ‚virtuelle Eigenschaft von Texten‘, die stets vom Rezipienten ‚rekonstruiert‘ werden muss“ [Fobbe 2011, 107]. Erste statistische Untersuchungen von Schreibstilen erfolgten bereits Mitte des 19. Jahrhunderts. Moderne stilometrische Forschung wurde durch die Veröffentlichungen von Yule und der durch ihn entwickelten *Characteristic K* (vgl. [Yule 1938]) geprägt (vgl. [Tuldava 2005, 370f]). Letztere ist ein Maß zur Erfassung der lexikalischen Diversität.

Zur quantitativen Erfassung von Stil werden verschiedene Eigenschaften eines Textes untersucht. Rudman schätzt, dass mehrere Tausend Eigenschaften und deren Kombinationen einen Schreibstil definieren (vgl. [Rudman 1997, 358]). Lexikalische Analysen erfassen den

Wortschatz eines Autors und untersuchen die lexikalische Diversität. Grammatikalische Analysen überprüfen die Häufigkeiten von Wortarten, sowie deren Korrelationen und Verteilungen. Semantische Analysen berücksichtigen den Kontext, in welchem Texte oder Wörter auftreten, um inhaltliche Aspekte zu erfassen (vgl. [Tuldava 2005, 372ff]). Die genannten Beispielmetriken können anhand der durch NLP-Systeme bereitgestellten Daten automatisiert ermittelt werden. Je nach Anwendungsgebiet werden darüber hinaus externe Informationen, wie zum Beispiel psychometrische oder soziologische Daten genutzt (vgl. [Tuldava 2005, 372ff]). Im Rahmen forensischer Linguistik werden stilometrische Methoden beispielsweise zur Erstellung von Täterprofilen verwendet (vgl. [Fobbe 2011, 108]).

Stilometrische Methoden werden häufig genutzt, um anonym verfasste Texte konkreten Autoren zuzuordnen (vgl. [Argamon et al. 2009; Hoorn et al. 1999; Gelbukh et al. 2018]). Die beschriebene Aufgabenstellung wird als *Authorship-Attribution*-Problem bezeichnet (vgl. [Stamatatos 2008, 539f]). Die in dieser Arbeit genutzten *feature sets* (siehe Kapitel 3.6.5) wurden im Rahmen der *Authorship-Attribution*-Forschung entwickelt (vgl. [Abbasi/Chen 2008, 7; Brennan et al. 2012, 20]). Die anschließenden Kapitel erläutern, inwiefern *Authorship-Attribution*-Verfahren im Kontext der automatisierten Plagiatanalyse verwendet werden.

2.3 Plagiatanalyse

Die folgenden Kapitel erörtern den Begriff des *Plagiarismus* und beschreiben bestehende Verfahren der automatisierten Plagiatanalyse.

2.3.1 Plagiarismus

Verschiedene Studien zeigen, dass ein erheblicher Teil der Hochschulstudierenden im Verlauf des Studiums plagiiert (vgl. [Gipp 2014, 336ff]). Bei einer Befragung von 80.000 Studierenden an 12.000 Fakultäten gaben 38 % der Bachelorstudierenden und 25 % der Masterstudierenden an, innerhalb der letzten zwölf Monate plagiiert zu haben (vgl. [Mccabe 2005, 6]). Gipp definiert *Plagiarismus* in Anlehnung an Fishman (vgl. [Fishman 2009, 5]) als eine ohne Angabe von Quellen erfolgende Nutzung vorhandener Ideen und Wörter, die in einem Kontext, in dem Originalität gefordert ist, dem eigenen Vorteil dient (vgl. [Gipp 2014, 10]). Gipp unterscheidet zwischen wortgetreuem und verschleiertem Plagiarismus. Zu ersterem zählen Plagiate, die nach den Methoden des *copy and paste* oder des *shake and paste* angefertigt wurden (vgl. [Gipp 2014, 11f]). Durch *copy and paste* angefertigte Plagiate werden ohne Veränderung aus einer Quelle übernommen (vgl. [Maurer et al. 2006, 1051f]).

Bei Plagiaten, die nach der *Shake-And-Paste*-Methode entstehen, wird der Originaltext durch eine Änderung der Wortreihenfolge oder die Nutzung von Synonymen geringfügig verändert (vgl. [Weber-Wulff 2010, 1]). Die Methoden des Umformulierens, des technischen Verschleierns, des Übersetzens, des strukturellen Plagiierens und des Selbstplagiierens werden dem verschleierten Plagiarismus zugeordnet (vgl. [Gipp 2014, 11f]). Die erstgenannte Methode beschreibt die Umformulierung des Originaltextes in eigenen Worten (vgl. [Clough 2000, 6f]). Durch technisches Verschleiern werden Plagiatanalysesysteme gezielt getäuscht. Hierbei werden beispielsweise Leerzeichen durch weiße Buchstaben ersetzt, um eine Worterkennung zu erschweren (vgl. [Heather 2010, 647]). Als Plagiatanalysesysteme werden Softwareprodukte zur automatisierten Erkennung von Plagiaten bezeichnet (vgl. [Gipp 2014, 33ff]). Plagiate, die durch Übersetzung des Originaltextes entstehen, können nur durch spezielle *Cross-Language-Plagiarism-Detection*-(CLPD)-Systeme erkannt werden (vgl. [Weber-Wulff 2010, 1]). Im Rahmen des strukturellen Plagiierens werden fremde Konzepte oder Ideen übernommen, ohne deren Quelle zu würdigen (vgl. [Fröhlich 2006, 82]). Im Rahmen des Selbstplagiierens werden hingegen eigene Texte wiederverwendet, ohne dies entsprechend kenntlich zu machen (vgl. [Bretag/Mahmud 2009, 193]). Die zuvor betrachtete Taxonomie des Plagiarismus sowie die in ihrem Kontext angegebenen Literaturverweise wurden durch Gipp beschrieben (vgl. [Gipp 2014, 11f]). Anhang C zeigt eine alternative, durch Alzahrani et al. entwickelte Taxonomie (vgl. [Alzahrani et al. 2012, 134]).

Das im Rahmen dieser Arbeit entwickelte Verfahren ist wie auch ein großer Teil der bestehenden Verfahren insbesondere für die Erkennung von wortgetreuem Plagiarismus geeignet (vgl. [Gipp 2014, 41]). Die Plagiatforschung untersucht neben plagiierten Fließtexten auch Fälle von Plagiarismus in Computerprogrammquellcode (vgl. [Sraka/Kaucic (2009)]). Das in der vorliegenden Arbeit entwickelte Verfahren sowie die in den folgenden Kapiteln erläuterten Ansätze der automatisierten Plagiatanalyse sind für die Untersuchung von Fließtexten konzipiert.

2.3.2 Verfahren der automatisierten Plagiatanalyse

2.3.2.1 Extrinsische Verfahren

Im Rahmen der automatisierten Plagiatanalyse wird zwischen extrinsischen und intrinsischen Verfahren unterschieden. Bestehende Plagiatanalysesysteme implementieren ausschließlich extrinsische Verfahren (vgl. [Gipp 2014, 33ff]). Intrinsische Ansätze werden jedoch im Kontext der Plagiatforschung untersucht (vgl. [Eissen/Stein (2006), 567]).

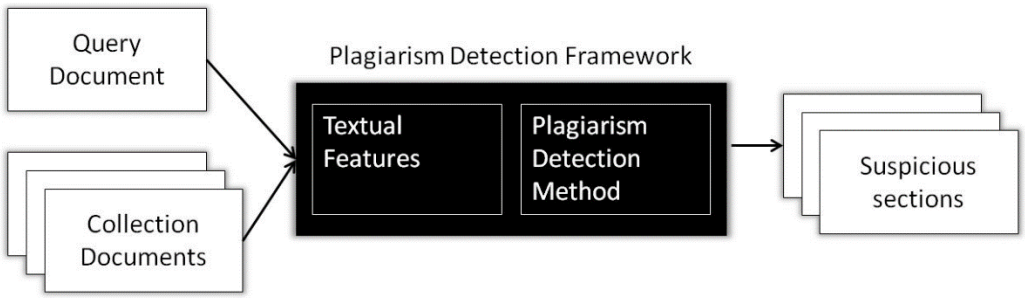


Abbildung 4: Extrinsische Plagiatanalysesysteme [Kraus 2016, 1]

Abbildung 4 zeigt die Nutzung eines extrinsischen Verfahrens im Rahmen eines Plagiatanalysesystems. Das Eingabedokument wird mit den im Referenzkorpus enthaltenen Dokumenten verglichen, um potentiell plagiierte Abschnitte zu markieren (vgl. [Kraus 2016, 1]). Der genutzte Referenzkorpus muss hierbei möglichst alle Dokumente enthalten, die als Quelle eines Plagiats verwendet werden können. Der Plagiatanalyseanbieter *Turnitin* indiziert beispielsweise gegenwärtig 62 Milliarden Websites, 734 Millionen Arbeiten von Studierenden und 165 Millionen Beiträge aus Fachzeitschriften (vgl. [o. V. 2018c]). Der beschriebene Korpus muss regelmäßig um neu entstandene Dokumente erweitert werden. Da der genutzte Referenzkorpus in der Regel zu groß ist, um jedes enthaltene Dokument detailliert mit dem Eingabedokument zu vergleichen (vgl. [Gipp 2014, 178]), nutzen extrinsische Verfahren ein mehrstufiges System. Die erste Phase der extrinsischen Plagiatanalyse löst ein *Information-Retrieval*-Problem. Anhand der aus dem Eingabedokument extrahierten Wörter sollen Dokumente im Referenzkorpus gefunden werden, die dem Eingabedokument möglichst ähnlich sind. Hierfür werden in der Regel Verfahren des *fingerprinting* oder Vektorraumdarstellungen genutzt (vgl. [Gipp 2014, 18f]). Bei letzteren werden das Eingabedokument und die Dokumente des Referenzkorpus als *bag of words* beziehungsweise als Vektorraummodell (siehe Kapitel 2.1.2) dargestellt. Die durch die Abbildung erzeugten Vektoren können effizient auf Differenz oder Ähnlichkeit untersucht werden. Das Verfahren des *fingerprinting* (vgl. [Gipp 2014, 22ff]) hingegen erzeugt für jedes Dokument eine Auswahl an Substrings und errechnet für diese einen Hashwert[1]. Jedes Dokument wird somit als eine Liste an Hashwerten repräsentiert (vgl. [Hoad/Zobel 2003, 208]). Die Ähnlichkeit zwischen dem Eingabedokument und einem Dokument des Referenzkorpus wird anschließend anhand

[1] Eine Hashfunktion bildet Strings beliebiger Länge auf einen als Hashwert bezeichneten String fixer Länge ab. Die Länge der Abbildung und die Abbildungsvorschrift werden so gewählt, dass Kollisionen in der Praxis nahezu ausgeschlossen sind. Als Kollision wird das Erzeugen gleicher Abbildungen für unterschiedliche Eingaben bezeichnet (vgl. [Aumasson et al. 2014, 1ff]).

der Schnittmenge der Hashwertlisten errechnet (vgl. [Gipp 2014, 22ff]). Die erste Phase der extrinsischen Plagiatanalyse erzeugt somit eine Auswahl an Dokumenten, die dem Eingabedokument ähnlich sind. Im Rahmen der zweiten Phase der extrinsischen Plagiatanalyse erfolgt eine ausführlichere Untersuchung der zuvor erzeugten Dokumentauswahl. Jedes in letzterer enthaltene Dokument wird detailliert mit dem Eingabedokument verglichen. Um plagiierte Abschnitte zu identifizieren wird häufig die Dichte an gemeinsamen Wort-N-Grammen geprüft (vgl. [Barrón-Cedeño/Rosso 2009]). Andere Ansätze nutzen zudem syntaktische (vgl. [Elhadi/Al-Tobi (2008)]) oder semantische (vgl. [Kraus 2016, 2f]) Analysen zur Identifikation von Plagiaten. Die dritte Phase der extrinsischen Plagiatanalyse wird als *Postprocessing*-Phase bezeichnet (vgl. [Gipp 2014, 18f]). Hierbei wird beispielsweise geprüft, ob in Abschnitten, die als potentielle Plagiate markiert wurden, korrekte Quellenangaben vorhanden sind (vgl. [Stein et al. 2007, 825]).

Spezialisiertere Ansätze der extrinsischen Plagiatanalyse nutzen die aus den Quelldokumenten übernommenen Zitationen, um Plagiate zu identifizieren (vgl. [Gipp 2014, 56ff]) oder greifen auf Verfahren der *authorship attribution* zurück. *Authorship-Attribution*-Verfahren ordnen Texte unbekannter Autoren auf Grundlage stilometrischer Analysen einem bekannten Autor zu. Die Schreibstile aller in der Ergebnismenge vertretenen Autoren müssen in einem Referenzkorpus hinterlegt werden (vgl. [Stamatatos 2008, 539f]). Lamas et al. entwickeln auf Basis der Arbeiten, die im Verlauf eines Studiums durch Studierende verfasst werden, eine Datenbank, die den Schreibstil der jeweiligen Studenten erfasst (vgl. [Lamas et al. 2014]). Auf Grundlage der genannten Datenbank können Textabschnitte studentischer Arbeiten, die durch andere Studenten verfasst wurden, dem Urheber zugeordnet und gegebenenfalls als Plagiat identifiziert werden. Textabschnitte, für welche keine Autorenerkennung möglich ist, werden hingegen im Rahmen eines *Authorship-Verification*-Verfahrens als verdächtig markiert. Verfahren der *authorship verification* werden insbesondere im Kontext der intrinsischen Plagiatanalyse genutzt und im folgenden Kapitel näher erläutert.

Jene Verfahren der extrinsischen Plagiatanalyse, welche die gefundenen, potentiellen Plagiate einem konkreten Quelldokument zuordnen, werden ihm Rahmen dieser Arbeit als *Document-Matching*-Verfahren bezeichnet.

2.3.2.2 Intrinsische Verfahren

Das Problem der *authorship verification* ist eine Spezialisierung des *Authorship-Attribution*-Problems. *Authorship attribution* ist als die Zuordnung konkreter Autoren zu einem Dokument unbekannten Autors definiert. Im Rahmen der *authorship verification* soll hingegen

entschieden werden, ob ein Dokument durch einen bestimmten Autor geschrieben wurde. Die Zielwerte entspringen somit nicht einer Menge an Autoren, sondern entsprechen einem booleschen Wert (vgl. [Koppel et al. 2011, 85; Gipp 2014, 30]).

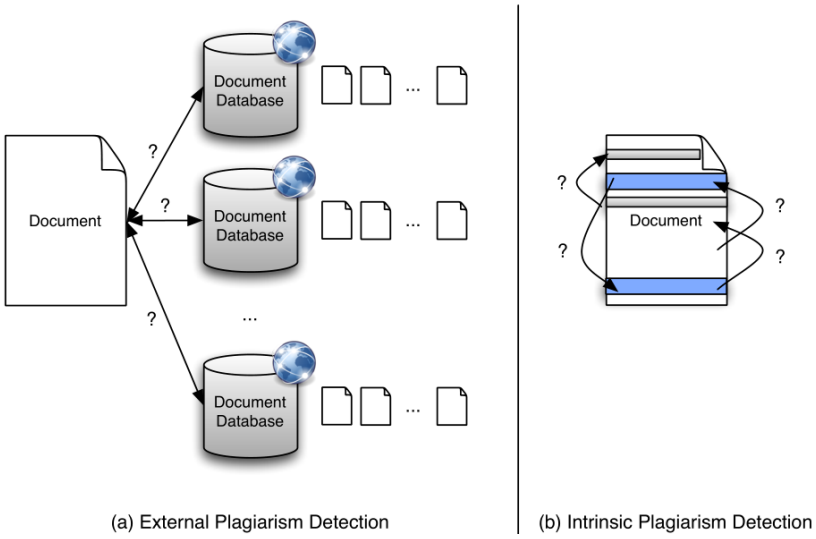


Abbildung 5: Extrinsische und intrinsische Plagiatanalyse [Tschuggnall 2014, 8]

Im Rahmen von Klassifikationsverfahren entspricht die Problemstellung der *authorship verification* einem *One-Class*-Klassifikationsproblem. Letzteres wird im Kontext intrinsischer Plagiatanalyseverfahren angewendet. Die hierbei definierte Klasse wird anhand eines mehrheitlich im Eingabedokument auftretenden Schreibstils beschrieben. Abschnitte, die diesem Schreibstil nicht entsprechen, werden als Ausreißer und somit als potentielle Plagiate markiert (vgl. [Stein et al. 2011, 63]). Abbildung 5 stellt ein intrinsisches Plagiatanalysesystem einem extrinsischen Plagiatanalysesystem gegenüber. Intrinsische Verfahren nutzen keinen Referenzkorpus, sondern ausschließlich die im Eingabedokument enthaltenen Informationen. Die in der Abbildung unter *b* dargestellten blauen und grauen Abschnitte entsprechen nicht dem weiß dargestellten mehrheitlichen Schreibstil und werden daher markiert. Intrinsische Verfahren können beispielsweise dann genutzt werden, wenn Quellen potentieller Plagiate nicht in digitaler Form vorliegen und somit in keinem Referenzkorpus enthalten sind (vgl. [Tschuggnall 2014, 8]). Gipp unterscheidet in Anlehnung an Stein et al. (vgl. [Stein et al. 2011, 67]) zwei Hauptkomponenten intrinsischer Plagiatanalysesysteme (vgl. [Gipp 2014, 31f]). Die Komponente der *Dekomposition* unterteilt das Eingabedokument in Segmente. Ein Segment ist beispielweise durch eine feste Anzahl an Zeichen oder Wörtern (vgl. [Stamatatos 2008, 553f]), eine Satzanzahl (vgl. [Muhr et al. 2009, 5]) oder durch die vorgegebenen Kapitel (vgl. [Uzuner et al. 2005, 40]) definiert (vgl. [Gipp 2014, 31f]). Die Komponente des *Stilmodells* nutzt stilometrische *feature sets* (siehe Kapitel 2.2), um lexikalische, syntaktische und strukturelle Eigenschaften der definierten Segmente zu

erfassen (vgl. [Gipp 2014, 31f]). Die Segmente des Eingabedokuments können anschließend auf Basis der erfassten Eigenschaften klassifiziert werden (vgl. [Stein et al. 2011, 63]).

Das im Rahmen dieser Arbeit entwickelte Verfahren ähnelt den Verfahren der *authorship attribution*. Statt eines konkreten Autors soll einzelnen Segmenten eines Eingabedokuments ein Dokumenttyp zugeordnet werden. Im Verlauf dieser Arbeit wird evaluiert, inwiefern das entwickelte Verfahren durch Nutzung externer Informationen eine höhere Performanz als intrinsische Verfahren erreicht. Das folgende Kapitel beschreibt die im Rahmen dieser Arbeit zu prüfende Hypothese und stellt die Eigenschaften des entwickelten Verfahrens den Eigenschaften bestehender Plagiatanalyseverfahren gegenüber.

3 Dokumenttypklassifikation im Kontext der Plagiatanalyse

Nachdem in den vorausgegangenen Kapiteln verschiedene Methoden des Data-Minings und bestehende Verfahren der Plagiatanalyse erläutert wurden, soll im Folgenden geprüft werden, inwiefern sich die Methoden der Dokumenttypklassifikation im Kontext der automatisierten Plagiaterkennung nutzen lassen.

3.1 Einordnung und Hypothese

Die in Kapitel 2.3.2.1 beschriebenen extrinsischen Verfahren erreichen in der Regel eine höhere Performanz als die in Kapitel 2.3.2.2 beschriebenen intrinsischen Verfahren (vgl. [Tschuggnall 2014, 8]). Letztere benötigen jedoch keinen zusätzlichen Referenzkorpus zur Erkennung potentieller Plagiate. Es werden ausschließlich die im Untersuchungsobjekt enthaltenen Informationen genutzt (vgl. [Stein et al. 2011, 63]). Ziel der vorliegenden Arbeit ist es, eine im Vergleich zu intrinsischen Verfahren performantere, extrinsische Methodik zu entwickeln, deren Referenzkorpus jedoch keiner regelmäßigen Erweiterung bedarf. Die folgenden Kapitel untersuchen, inwiefern hierfür die Methoden der Dokumenttypklassifikation geeignet sind.

Auf *authorship verification* basierende Plagiatanalyseverfahren versuchen, innerhalb eines Eingabedokuments stilistische Ausreißer zu identifizieren. Hierfür werden einzelne Textsegmente mit dem Rest des Dokuments verglichen. Als Referenz wird der mehrheitliche Schreibstil des Untersuchungsobjekts genutzt (vgl. [Gipp 2014, 30]). Auf *document matching* basierende Verfahren ordnen Textsegmente des Eingabedokuments konkreten Quelldokumenten zu. Nur Dokumente, welche im Referenzkorpus hinterlegt sind, können Teil der Ergebnismenge sein (vgl. [Gipp 2014, 22]). Die Identifikation eines unbekannten

Quelldokuments ist nicht möglich. Nur durch *document matching* kann ein direkter Plagiatnachweis erfolgen. Andere Verfahren markieren lediglich jene Textsegmente, welche potentiell durch einen anderen Autor verfasst wurden. Die zuvor beschriebenen Eigenschaften verschiedener Plagiatanalyseverfahren werden in Tabelle 1 zusammengefasst.

Art des Verfahrens	**Verfahren**	**Art des Referenzkorpus**	**Größe des Referenzkorpus**	**Identifikationsziel**	**Plagiatnachweis**	**Einschränkung der Eingabe**
intrinsisch	authorship verification	ein Dokument	sehr klein	stilistische Ausreißer	indirekt	keine
extrinsisch	document matching	alle Dokumente	sehr groß	Dokumente	direkt	keine
extrinsisch	Dokumenttypklassifikation	alle Dokumenttypen	klein	Dokumenttypen	indirekt	studentische Abschlussarbeiten

Tabelle 1: Dokumenttypklassifikation im Vergleich zu bestehenden Ansätzen der Plagiatanalyse

Das zu evaluierende, auf Dokumenttypklassifikation basierende Verfahren versucht Textsegmente einem Dokumenttyp zuzuordnen. Als Referenz dient ein Korpus mit Beispielen für die Stilausprägungen verschiedener Dokumenttypen. Grundlage hierbei ist die Annahme, dass verschiedene Dokumenttypen mit verschiedenen stilistischen Eigenschaften verknüpft sind. So könnte sich beispielsweise die durchschnittliche Satzlänge einer studentischen Abschlussarbeit von jener eines Fachartikels unterscheiden. Die eingesetzten stilometrischen *feature sets* dienen hierbei nicht der Identifikation einzelner Autoren, sondern der Bestimmung eines durchschnittlichen, für den jeweiligen Dokumenttyp typischen Schreibstils. Im Kontext der Plagiatprüfung studentischer Abschlussarbeiten würde ein menschlicher Prüfer nach der beschriebenen Methodik nicht wie bei intrinsischen Verfahren nach stilistischen Veränderungen in Bezug auf den Rest des Dokuments suchen, sondern auf sein Vorwissen über die typischen Stileigenschaften einer Bachelor- oder Masterarbeit zurückgreifen. Die Anzahl der zu definierenden Stile ist hierbei durch die Anzahl möglicher Dokumenttypausprägungen begrenzt.

Nach Manar und Shameem (vgl. [Manar/Shameem 2014, 753]) haben mehr als 83 % der Bachelorstudierenden mindestens einmal eine elektronische Quelle plagiiert. Mehr als 62 % plagiierten mindestens einmal eine Printquelle und mehr als 51 % plagiierten mindestens einmal einen Kommilitonen oder andere Menschen in ihrem Umfeld. Das zu evaluierende Verfahren ist auf Plagiatfälle begrenzt, in denen Studierende dokumenttypfremde Quellen plagiieren. Eine Quelle ist als dokumenttypfremd definiert, wenn sich ihr Dokumenttyp vom Dokumenttyp der plagiierenden Arbeit unterscheidet. Segmente innerhalb einer Bachelor-

oder Masterarbeit, deren Quelle eine andere Bachelor- oder Masterarbeit ist, werden nicht als potentielle Plagiate erkannt. Die Begrenzung auf studentische Abschlussarbeiten entspringt der Annahme, dass Plagiate innerhalb von anderen Dokumenttypen wie beispielsweise Fachartikeln nur selten auf dokumenttypfremde Quellen zurückzuführen sind.

Im Folgenden soll zunächst geprüft werden, ob eine automatisierte Unterscheidung von Dokumenttypen im Kontext der Plagiatanalyse studentischer Abschlussarbeiten möglich ist. Nach Überprüfung der Grundhypothese wird evaluiert, welche Ergebnisse im Vergleich zu anderen Plagiatanalyseverfahren erzielt werden. Zudem werden Konzepte und Trainingskorpora als Grundlage für eine prototypische Implementierung erarbeitet und zur Verfügung gestellt.

3.2 Methodik

Zur Entwicklung und Überprüfung des untersuchten Verfahrens werden Prozesse des Data-Minings und Evaluationsmethoden der automatisierten Plagiatanalyse genutzt. Die Kapitel 3.2.1 und 3.2.2 erläutern, welche grundlegenden Konzepte hierbei eine besondere Relevanz haben.

3.2.1 Evaluation von Plagiatanalyseverfahren

Für die Evaluation stilometrischer Verfahren im Kontext der Autorenerkennung stehen zahlreiche Korpora, wie der *Brennan-Greenstadt*- (vgl. [Brennan et al. 2012, 7]), der *CC04*- und der *FED*-Korpus (vgl. [Tschuggnall 2014, 70]) zur Verfügung. Die Evaluation von Plagiatanalyseverfahren erweist sich hingegen als schwieriger, da aufgrund juristischer und ethischer Bedenken von der Nutzung realer Plagiate abgesehen wird (vgl. [Kraus 2016, 2]). Die in der Forschung genutzten Evaluationskorpora basieren daher auf künstlichen oder simulierten Plagiatfällen. Künstlicher Plagiarismus beschreibt hierbei Plagiatfälle, welche durch die automatisierte Vermischung und Nachbearbeitung von Textstellen aus verschiedenen Quellen erzeugt werden. Simulierter Plagiarismus hingegen bezeichnet Plagiatfälle, die durch manuelles Umformulieren gegebener Textsegmente entstehen (vgl. [Kraus 2016, 2]). Der durch Potthast et al. erstellte *PAN Plagiarism Corpus 2010* (PAN-PC-10) schafft eine gemeinsame Basis zur Evaluation der Performanz verschiedener Plagiatanalyseverfahren (vgl. [Potthast et al. 2010]). Die genutzten Quelldokumente entstammen der freien Büchersammlung *Project Gutenberg*[2]. Die auf simuliertem Plagiarismus basierenden Plagiate

[2] http://www.gutenberg.org

wurden durch 907 verschiedene Bearbeiter der *Amazon Mechanical Turk*[3] erzeugt. Der größere Teil der Plagiate ist durch automatisierte Mischung von Quelldokumenten entstanden (vgl. [Potthast et al. 2010, 5]). Ein Teil der Plagiate wurde durch *Obfuskation* verändert. Letztere beschreibt die Löschung, Einfügung oder Ersetzung von Wörtern (vgl. [Muhr et al. 2009, 53]). Abbildung 6 zeigt verschiedene Beispiele für Obfuskationsmethoden im *PAN-PC-10*.

Obfuscation Examples
Original Text The quick brown fox jumps over the lazy dog.
Manual Obfuscation (by a human) Over the dog which is lazy jumps quickly the fox which is brown. Dogs are lazy which is why brown foxes quickly jump over them. A fast auburn vulpine hops over an idle canine.
Random Text Operations over The. the quick lazy dog <*context word*> jumps brown fox over jumps quick brown fox The lazy. the brown jumps the. quick dog The lazy fox over
Semantic Word Variation The quick brown dodger leaps over the lazy canine. The quick brown canine jumps over the lazy canine. The quick brown vixen leaps over the lazy puppy.
POS-preserving Word Shuffling The brown lazy fox jumps over the quick dog. The lazy quick dog jumps over the brown fox. The brown lazy dog jumps over the quick fox.

Abbildung 6: Beispiele für Obfuskationstechniken im PAN-PC-10 [Potthast et al. 2010, 6]

Insgesamt umfasst der beschriebene Korpus 27.073 Dokumente und 68.558 Plagiatfälle. Er ist in einen intrinsischen und einen extrinsischen Abschnitt mit jeweils abweichenden Eigenschaften unterteilt. Da die reale Häufigkeit und Verteilung von Plagiaten innerhalb von Dokumenten unbekannt ist, wurde diese geschätzt (vgl. [Potthast et al. 2010, 6]).

Keiner der zuvor aufgezählten Korpora kann direkt für die Evaluation des in der vorliegenden Arbeit entwickelten Verfahrens genutzt werden. Ein für eine direkte Evaluation geeigneter Korpus wäre aus gekennzeichneten Plagiaten innerhalb von studentischen Abschlussarbeiten zusammengesetzt. Die im Verlauf der Arbeit genutzten Techniken greifen jedoch auf das Konzept des künstlichen Plagiarismus zurück, um die zur Entwicklung und Überprüfung notwendigen Korpora zu erstellen. Zudem werden in Abschnitt 3.7.2 statistische

[3] https://www.mturk.com/

Metriken des *PAN Plagiarism Corpus 2011* (PAN-PC-11) zur besseren Beurteilung der Gesamtperformanz genutzt (vgl. [Potthast (2011), 2]).

3.2.2 Cross-Industry Standard Process for Data Mining

Zur Entwicklung und Evaluation des in der vorliegenden Arbeit untersuchten Klassifikationsverfahrens wird der durch *CRISP-DM* definierte Ablauf als Rahmen genutzt. *CRISP-DM* wurde entwickelt, um Forschern und Unternehmen einen domänenunabhängigen und anwendungsneutralen Standardprozess für die Durchführung von Data-Mining-Vorhaben zur Verfügung zu stellen (vgl. [Larose 2014, 4]). *CRISP-DM* bezieht sich hierbei vorrangig auf die Umsetzung von Data-Mining-Vorhaben, in deren Kontext komplexe Modellierungstechniken angewendet werden. Rein deskriptive Analysen können als Teilprozesse der Datenexploration und Datenvorverarbeitung eingeordnet werden (vgl. [Shearer 2000, 18]). Alle Phasen des Prozesses sind adaptiv. Abhängig vom Resultat eines Teilprozesses muss gegebenenfalls der vorausgegangene Prozess angepasst werden. Der gesamte Prozess ist zudem iterativ, das heißt, das durch den Gesamtprozess entstehende Wissen wird bei der Gestaltung weiterer Data-Mining-Prozesse berücksichtigt (vgl. [Larose 2014, 4]). Abbildung 7 zeigt alle Phasen des *CRISP-DM* und verdeutlicht das iterative Konzept.

Die Phase der Problemdefinition wird im Kontext von Unternehmungen auch als Phase des *business understanding* bezeichnet. Ziel des Schrittes ist die Definition und Eingrenzung der Ziele des Data-Mining-Vorhabens (vgl. [Roiger 2017, 215]). Zudem muss sichergestellt werden, dass die für die Erreichung der Ziele notwendigen Daten verfügbar oder beschaffbar sind (vgl. [Shearer 2000, 14f]).

Das Zusammentragen der benötigten Daten ist Teil der Datenexplorationsphase. Es erfolgt zudem eine Beschreibung der vorhandenen Formate und der gegebenenfalls erkannten Datenprobleme. Des Weiteren werden Datenstrukturen wie zum Beispiel Tabellenspalten oder Objektfelder erfasst und beurteilt. Durch die Anwendung explorativer Datenanalysetools werden vorhandene Muster und statistische Eigenschaften der Daten ermittelt (vgl. [Shearer 2000, 15f]). Zudem erfolgt eine Beurteilung der Prozesse, welche zur Erzeugung der Daten beigetragen haben (vgl. [Reese et al. 2017, 362]).

Im Teilprozess der Datenvorverarbeitung müssen die zu nutzenden Daten in eine für Data-Mining-Verfahren auswertbare Form gebracht werden (vgl. [Reese et al. 2017, 362]). Im Rahmen der Datenselektion werden für das Vorhaben relevante Informationen ausgewählt. Durch Datenintegrationsprozesse erfolgt die Zusammenführung und Harmonisierung verschiedener Quellen und Formate. Durch Datenaggregation kann die auszuwertende Menge

an Daten reduziert werden (vgl. [Shearer 2000, 16f]). Werden keine vorgefertigten *feature sets* genutzt, so erfolgt in dieser Phase auch das in Kapitel 2.1.2 beschriebene *feature engineering* (vgl. [Reese et al. 2017, 362]). Nach Shearer werden in der Regel 50 bis 70 % der Gesamtaufwendungen für die Datenvorverarbeitung erbracht (vgl. [Shearer 2000, 14f]).

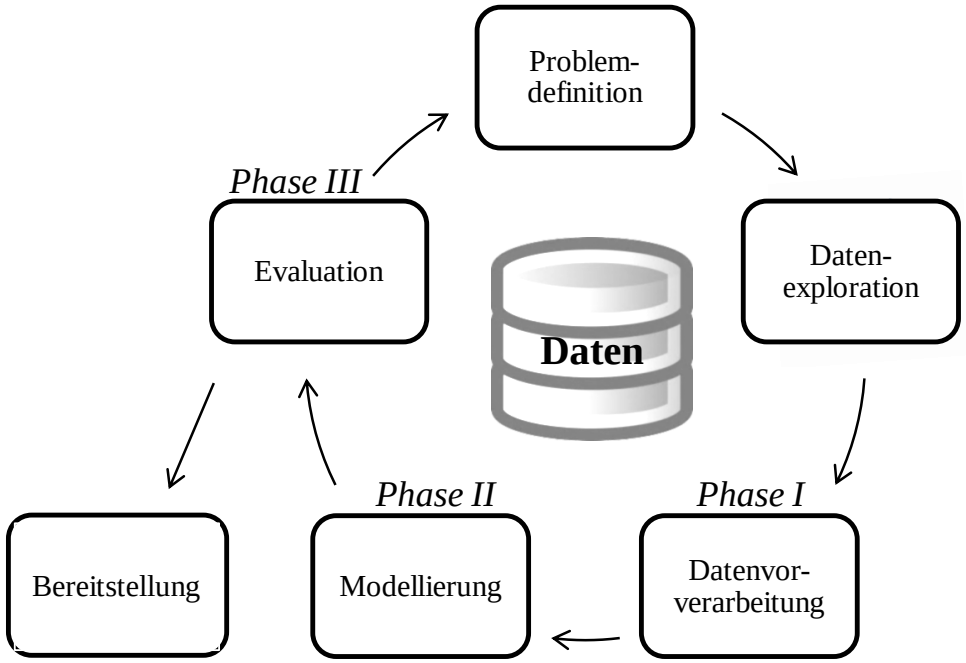


Abbildung 7*: Cross-Industry Standard Process for Data Mining* (in Anlehnung an [Witten et al. 2017a, 29])

Die Phase der Modellierung beginnt mit der Auswahl passender Modellierungstechniken. Hierzu zählen zum Beispiel Entscheidungsbäume oder neuronale Netzwerke. Sie bilden die Grundlage des Data-Mining-Prozesses und erstellen auf Basis der zuvor ausgewählten Daten ein passendes Modell (vgl. [Reese et al. 2017, 362]). Zudem werden Testszenarien erstellt und ausgewertet, um verschiedene Modelle aus technischer Sicht zu evaluieren. Nach einer Interpretation der Ergebnisse in Bezug auf den Anwendungsfall kann das am besten geeignete Modell in einen Evaluationsprozess unter möglichst realen Bedingungen überführt werden (vgl. [Shearer 2000, 17]).

Der Evaluationsprozess überprüft das zuvor ausgewählte Modell in Bezug auf den Anwendungsfall und die zu Beginn definierten Ziele (vgl. [Reese et al. 2017, 362]). Im Idealfall geschieht dies durch Applikation unter realen Bedingungen. Auf Grundlage der Evaluationsergebnisse wird entschieden, ob das entwickelte Modell eingesetzt werden kann oder weiterer Überarbeitung bedarf (vgl. [Shearer 2000, 17f]).

Im Rahmen des Bereitstellungsprozesses wird das entwickelte Modell in seine Produktivumgebung integriert (vgl. [Reese et al. 2017, 362]). Durch die Definition einer Bereitstellungsstrategie kann eine Fehlanwendung des Modells verhindert werden (vgl. [Shearer 2000, 18]).

Im Kontext dieser Arbeit werden die Schritte der Datenvorverarbeitung, der Modellierung und der Evaluation als die drei *Hauptphasen der Verfahrensentwicklung* bezeichnet. In den genannten Phasen werden jene Prozesse durchgeführt, die für die Umsetzung und Prüfung des entwickelten Verfahrens am relevantesten sind.

3.3 Problemdefinition

Das folgende Kapitel dient der Definition der durch das Data-Mining-Vorhaben zu lösenden Probleme. Zudem werden die Ziele des Vorhabens formuliert und eingegrenzt (vgl. [Roiger 2017, 215]).

Das in dieser Arbeit untersuchte Verfahren soll potentiell plagiierte Abschnitte innerhalb studentischer Abschlussarbeiten erkennen. Während der Entwicklung und Überprüfung des Verfahrens müssen verschiedene Teilprobleme gelöst und Annahmen erfüllt sein. Grundhypothese ist, dass die Markierung von potentiellen Plagiaten in Bachelor- und Masterarbeiten durch die Methoden der Dokumenttypklassifikation erfolgen kann. Ausgangspunkt dieser Annahme ist der Umstand, dass in studentischen Abschlussarbeiten besonders häufig Passagen aus Dokumenten anderen Typs übernommen werden (vgl. [Manar/Shameem 2014, 753]). Die im Verlauf der Arbeit genutzten Trainings- und Testkorpora müssen demzufolge sowohl Abschnitte aus Bachelor- und Masterarbeiten als auch Abschnitte anderen Typs enthalten. Da keine der in Kapitel 3.2.1 vorgestellten Dokumentsammlungen diese Eigenschaft erfüllt, müssen eigene Korpora erstellt werden. Da reale Plagiatfälle nicht zur Verfügung stehen, wird das Konzept des künstlichen Plagiarismus aufgegriffen. Abschnitte studentischer Abschlussarbeiten und dokumenttypfremde Abschnitte werden im Korpus gemischt, um die Eigenschaften plagiierter Dokumente zu simulieren. Auf Obfuskation wird hierbei zunächst verzichtet. Die neben *Bachelor-* und *Masterarbeiten* ausgewählten Dokumenttypen sind *Fachbücher*, *Fachartikel* und *Wikipediaartikel*. Die getroffene Auswahl ist nicht vollständig, aber deckt nach den Studien von Manar, Shameen (vgl. [Manar/Shameem 2014, 753]) und Turnitin (vgl. [o. V. 2011]) einen großen Teil der Plagiatquellen innerhalb von studentischen Abschlussarbeiten ab. Die genutzten Bachelor- und Masterarbeiten sollten möglichst wenig typfremde Segmente enthalten, da ein zu hoher Anteil letzterer die Performanz der Klassifikation negativ beeinflusst. Da die für die Klassifikation genutzten stilometrischen *feature sets* zum Teil sprachabhängig angepasst werden müssen (vgl. [Zheng et

al. 2006, 385f]), wird der Problembereich auf studentische Abschlussarbeiten in deutscher Sprache eingegrenzt.

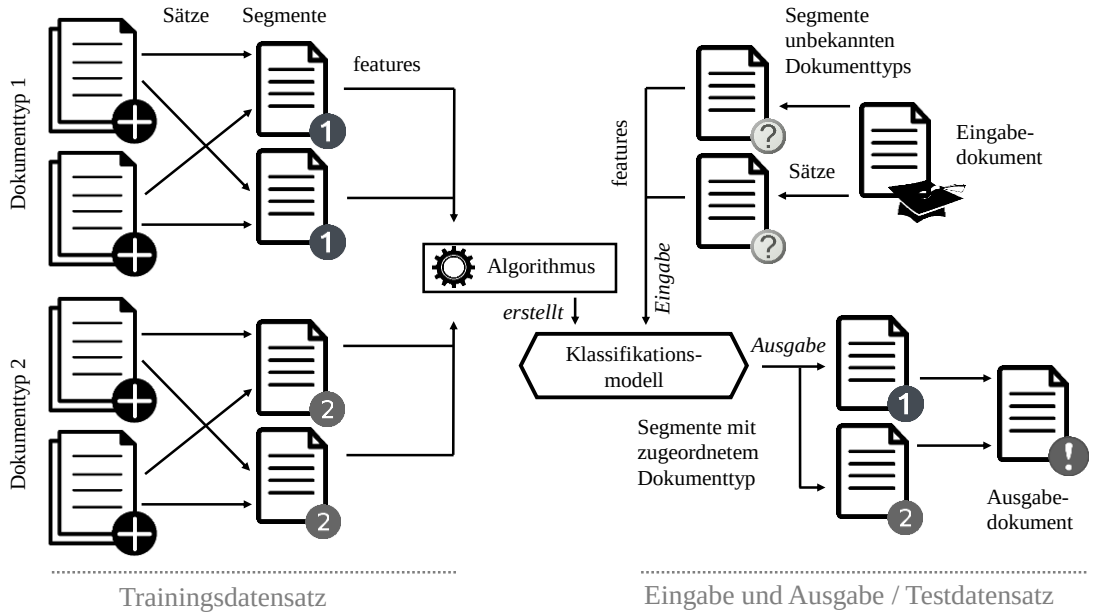


Abbildung 8: Technische Problemdefinition[4] (in Anlehnung an [Tan et al. 2009, 148])

Wie bereits in Kapitel 2.1.3 beschrieben, zählt die Dokumenttypklassifikation zu den Anwendungen des überwachten maschinellen Lernens. Aus einem Trainingskorpus mit markierten Textsegmenten aus Quellen unterschiedlichen Dokumenttyps werden Regeln abgeleitet, nach welchen der Dokumenttyp eines unbekannten Textsegments bestimmt werden kann. Zur Evaluation wird ein Testkorpus aus nicht markierten Textabschnitten unterschiedlichen Typs genutzt. Entspricht der Dokumenttyp nicht einer Bachelor- oder Masterarbeit, so soll dies als potentielles Plagiat gedeutet werden. Können einzelne Textabschnitte mit ausreichender Sicherheit einem Dokumenttyp zugeordnet werden, so ist das Verfahren prinzipiell geeignet, um dokumenttypfremde Segmente innerhalb studentischer Abschlussarbeiten als potentielle Plagiate zu markieren. Nach Bestätigung der Grundhypothese wird das Verfahren in Kapitel 3.7.3 unter möglichst realen Bedingungen getestet.

Abbildung 8 betrachtet die zuvor beschriebenen Teilprobleme aus technischer Sicht. Aus Segmenten eines Trainingsdatensatzes wird durch einen geeigneten Algorithmus ein Klassifikationsmodell abgeleitet, welches die Segmente eines Eingabedokuments einem Dokumenttyp zuordnet. Im Kontext dieser Arbeit werden die für die Evaluation genutzten Einga-

[4] https://commons.wikimedia.org/wiki/File:Gear_1.svg von Eugrafia,
https://en.wikipedia.org/wiki/File:Add_document_icon_(the_Noun_Project_27896).svg von A. Purwanto

bedokumente durch das Zusammensetzen von Segmenten verschiedenen Dokumenttyps simuliert. Die durch das Modell erzeugten Ausgaben werden zudem direkt auf Segmentebene betrachtet. Es erfolgt keine Zusammensetzung der Segmente zu einem Ausgabedokument.

Die Kapitel 3.4 und 3.5 beschreiben, inwiefern die Schritte der Datenexploration und Datenvorverarbeitung zur Erzeugung der benötigten Trainings- und Testkorpora beigetragen haben.

3.4 Datenexploration

Im Rahmen der Datenexploration erfolgt die Beschaffung, Analyse und Vorfilterung der für das Data-Mining-Vorhaben benötigen Daten (vgl. [Roiger 2017, 215; Shearer 2000, 15f]).

Wie im vorausgegangenen Kapitel beschrieben, müssen verschiedene, im Kontext studentischer Abschlussarbeiten häufig auftretende Dokumenttypen in den genutzten Korpora vertreten sein. Als repräsentativ für Printmedien wurden die Typen *Fachartikel* und *Fachbuch* ausgewählt (vgl. [Manar/Shameem 2014, 753]). Zur Abbildung elektronischer Medien werden *Wikipediaartikel* genutzt. *Wikipedia* ist die durch Studierende aus dem tertiären Bildungsbereich am häufigsten plagiierte Onlinequelle (vgl. [o. V. 2011]). Als repräsentativ für studentische Abschlussarbeiten wurden die Dokumenttypen *Bachelorarbeit* und *Masterarbeit* gewählt.

Für das Zusammentragen der benötigen Dokumente der Typen *Fachartikel*, *Fachbuch*, *Bachelorarbeit* und *Masterarbeit* wurde der durch *Open-Access*[5] bereitgestellte Korpus und die darin umgesetzte Taxonomie genutzt. *Open-Access* veröffentlicht Literatur kostenfrei und ohne gesetzliche Barrieren im Internet (vgl. [o. V. 2018e]). Taxonomien sind hierarchische Klassifikationsschemata und können als Ausgangspunkt für Klassifikationsvorhaben genutzt werden (vgl. [Bedford 2012, 13]). Die durch *Open-Access* angebotenen Dokumente werden nach Dokumenttyp und Genre klassifiziert. Letztere sind im beschriebenen Kontext als einzelne wissenschaftliche Fachbereiche definiert. Zudem werden bibliographische Daten und die Sprache des Dokuments erfasst. Für den im Kontext dieser Arbeit erstellten Korpus wurden nur Dokumente aus den Genres *Literaturwissenschaften*, *Naturwissenschaften*, *Philosophie*, *Sozialwissenschaften* und *Technikwissenschaften* genutzt. Die beschriebene Auswahl soll möglichst viele genreabhängige Abweichungen innerhalb der Dokumenttypen

[5] http://oansuche.open-access.net/oansearch/

abbilden. Die durch *Open-Access* angebotenen Bachelor- und Masterarbeiten sind fast ausschließlich auf Veröffentlichungen durch Hochschulen zurückzuführen. Es ist daher von einer vergleichsweise hohen Qualität der studentischen Abschlussarbeiten auszugehen.

Für das Zusammentragen von Wikipediaartikeln wurde auf die durch *Linguatools*[6] bereitgestellten Korpora zurückgegriffen. Die angebotenen Dokumentsammlungen sind für die Anwendung im NLP-Kontext konzipiert und erfassen unter anderem die Sprache der verwendeten Artikel. Die Veröffentlichung erfolgt im Rahmen der *Creative-Commons-Lizenz* (vgl. [o. V. 2018f]).

Insgesamt wurden 126 Bachelorarbeiten, 768 Masterarbeiten, 601 Fachartikel und 1.141 Bücher via *Open-Access* heruntergeladen. Die jeweilige Menge ist begrenzt durch das Angebot an deutschsprachigen Dokumenten im jeweiligen Dokumenttyp mit passendem Genre. Ein Download erfolgte nur, wenn der Volltext im PDF-Format über einen Direktlink zur Verfügung stand. Für jeden Dokumenttyp wurde zudem die Genrezuordnung erfasst. Aus dem durch *Linguatools* angebotenen Korpus wurden 43.790 deutschsprachige Wikipediaartikel im Extensible-Markup-Language-(XML)-Format extrahiert. Da der genannte Korpus nicht nach Genres klassifiziert, wurden nur Artikel extrahiert, die durch eine Schlagwortsuche den zuvor beschriebenen Genres zugeordnet werden konnten.

Da grundlegende Strukturen durch die von *Open-Access* übernommene Taxonomie vorgegeben sind, stand im weiteren Verlauf der Datenexploration vor allem die Vorfilterung der Daten im Fokus. Hierbei muss beurteilt werden, inwiefern die vorliegenden Daten zur Erreichung der in der Problemdefinitionsphase definierten Ziele geeignet sind (vgl. [Shearer 2000, 15f]). Die Vorfilterung der Wikipediaartikel erfolgte bereits beim Extrahieren der XML-Dateien. In Anlehnung an Kilgarriff et al. wurden nur Artikel mit einer Mindestlänge von 500 Wörtern verwendet (vgl. [Kilgarriff et al. (2010), 2]). Die durch *Open-Access* in der Kategorie *Bücher* angebotenen Dokumente wurden händisch gefiltert und auf den Typ *Fachbuch* reduziert. Auch ungeeignete Dokumente aus anderen Dokumenttypen wurden entfernt. Hierzu zählen unter anderem Dokumente ohne *optical character recognition* (OCR), Schriftstücke mit schlechter OCR-Qualität und Fachartikel, die keinen Volltext beinhalten. Nach Durchführung der beschriebenen Vorauswahl wurden insgesamt 1.875 PDF- und 43.790 XML-Dokumente in die Phase der Datenvorverarbeitung überführt.

[6] http://linguatools.org/tools/corpora/wikipedia-monolingual-corpora/

3.5 Datenvorverarbeitung

Im Teilprozess der Datenvorbereitung erfolgt eine Homogenisierung der Daten, um diese in eine für Data-Mining-Vorhaben nutzbare Form zu bringen (vgl. [Reese et al. 2017, 362]). Der genannte Schritt wird im Kontext dieser Arbeit als *Phase I der Hauptphasen der Verfahrensentwicklung* bezeichnet. Letztere dient der Erstellung eines Basiskorpus, aus welchem im weiteren Verlauf der Arbeit Trainings- und Testkorpora erstellt werden können (siehe Abbildung 9).

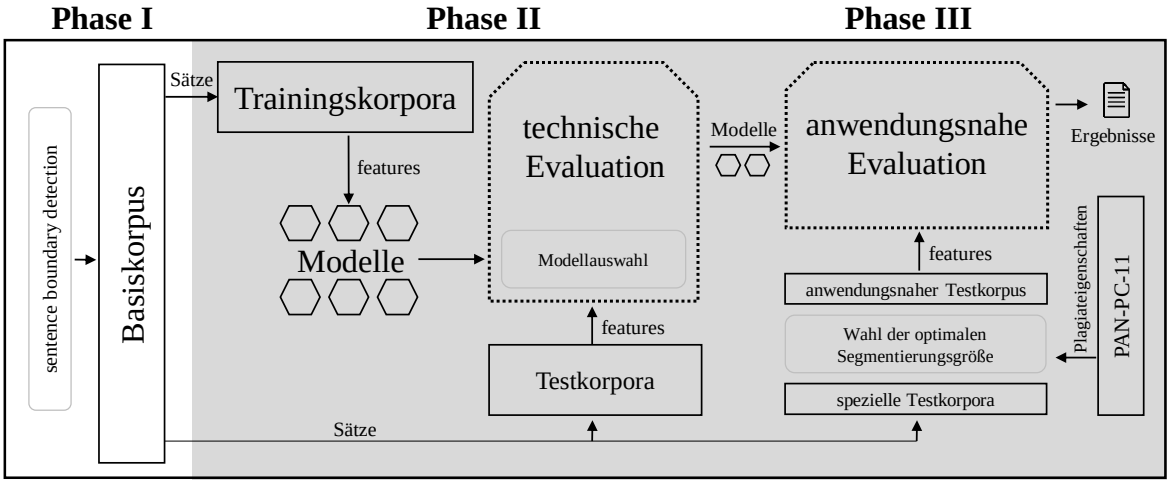


Abbildung 9: *Hauptphasen der Verfahrensentwicklung*, Phase I

3.5.1 Parsing

Im Rahmen des *parsing* werden die für das Data-Mining-Vorhaben benötigten Textdaten aus den eingegebenen PDF- und XML-Dateien extrahiert. Die Erkennung der relevanten Passagen innerhalb des layoutbasierten PDF-Formats ist ein komplexer Vorgang, da nur die Positionen einzelner Buchstaben und keine separierten Sinneinheiten im Dokument hinterlegt sind (vgl. [H. Bast/C. Korzen 2017, 1]). In der vorliegenden Arbeit wird hierfür das Programm *Science Parse*[7] verwendet. *Science Parse* nutzt Methoden des maschinellen Lernens, um wissenschaftliche Veröffentlichungen im PDF-Format in ein strukturiertes JavaScript-Object-Notation-(JSON)-Format zu überführen. Im Kontext des in dieser Arbeit entwickelten Verfahrens werden nur die von *Science Parse* als Paragraphen markierten Fließtexte aus den Dokumenten extrahiert. Informationen, die sich aus der Dokumentstruktur ergeben, wie Überschriften, Seitenzahlen oder Annotationen werden in Anlehnung an Grivas et al. und Stamatatos et al. (vgl. [Grivas et al. (2015), 2; Stamatatos et al. 2001, 203]) ignoriert. Das beschriebene Vorgehen beruht auf der Annahme, dass in plagiierten Passagen

[7] https://github.com/allenai/science-parse

vor allem Fließtexte des Quelldokuments übernommen werden. Strukturelle Informationen sollten daher nicht in die künstlich erzeugten Plagiate integriert werden. Die Überführung der erzeugten JSON-Dateien in eine Java-kompatible Repräsentation erfolgte auf Grundlage der *GSON*[8] und *Jsonschema2Pojo*[9] Bibliotheken.

Für das *parsing* der XML-Dateien wurde das durch *Linguatools* bereitgestellte Perl-Skript *XML2TXT.PL*[10] genutzt. Die XML-Daten verwenden die durch das *Hypertext-Markup-Language*-(HTML)-Format der Quelldokumente vorgegebene Struktur. Die benötigten, den Fließtext enthaltenden Paragraphen lassen sich somit anhand der bereits vorhandenen Markierungen extrahieren und in ein Textdokument übertragen. Nachdem die benötigten Textpassagen in eine geeignete Darstellung überführt wurden, kann der Teilschritt des *preprocessing* auf Basis einer in *Java* implementierten Filter- und Homogenisierungsprozedur erfolgen.

3.5.2 Preprocessing

Der Teilschritt des *preprocessing* ist insbesondere bei der Verwendung von stilometrischen *feature sets* von besonderer Relevanz (vgl. [Ramnial et al. 2016, 117]). Er harmonisiert unter anderem nicht konforme Verwendungen von Satzzeichen und überführt Abkürzungen in eine Darstellung ohne Punkte, um den anschließenden Prozess der Satzerkennung zu erleichtern (vgl. [Kiss/Strunk 2006, 485]). Zudem werden Elemente, die aufgrund eines fehlerhaften *parsing* als Teil des Fließtextes erkannt wurden, entfernt. Hierzu zählen Kapitelnummern, Bildunterschriften und diverse Steuerzeichen (vgl. [Tschuggnall 2014, 20]).

Die Harmonisierung der Punktuations- und Leerzeichenverwendung orientiert sich an einer durch Magerman et al. beschriebenen Prozedur (vgl. [Magerman et al. 2006, 27]). Hierbei wird beispielsweise sichergestellt, dass Satzendezeichen ein Leerzeichen nachgestellt und kein Leerzeichen vorausgestellt ist. Im Zuge der Harmonisierung werden zudem alle Zeichen entfernt, die keine Leerzeichen, Buchstaben oder Satzendezeichen darstellen. Harmonisierungsprozesse sind in der Regel Reduktionsprozesse und stellen somit einen Informationsverlust dar (vgl. [Han et al. 2012, 99f]). Inkorrekte Zeichensetzungen können im Rahmen stilometrischer Analysen zur Informationsgewinnung genutzt werden. Es muss daher abgewogen werden, inwiefern die Verbesserung der Satzerkennungsprozedur und die Entfernung von Störelementen den entstehenden Informationsverlust ausgleichen.

[8] https://github.com/google/gson
[9] https://github.com/joelittlejohn/jsonschema2pojo/
[10] https://www.dropbox.com/s/p3ta9spzfviovk0/xml2txt.pl?dl=0

Zur Verbesserung der automatisierten Satzerkennung wurden Abkürzungen in eine Darstellung ohne Punkte überführt. Hierbei wurde die durch das Projekt *TreeTagger*[11] zur Verfügung gestellte Liste deutschsprachiger Abkürzungen genutzt. Des Weiteren wurden verschiedene Klammerdarstellungen vereinheitlicht und einschließlich des Klammerinhalts entfernt. Ziel dieser Reduktion ist die Entfernung von Zitationsangaben (vgl. [Ramnial et al. 2016, 117]). Letztere erfolgen in verschiedenen Dokumenten nach unterschiedlichen Formatvorlagen und sollen daher nicht zur Auswertung genutzt werden. Da stilometrische *feature sets* zum Teil sprachabhängig angepasst werden müssen (vgl. [Zheng et al. 2006, 385f]) und das im Rahmen dieser Arbeit entwickelte Verfahren nur Eingabedokument in deutscher Sprache verarbeitet, wurden englische Abschnitte innerhalb der deutschsprachigen Dokumente entfernt. Hierbei kam das Analyseframework *Apache Tika*[12] zur Anwendung.

3.5.3 Sentence Boundary Detection und Erstellung des Basiskorpus

Sentence boundary detection ist die Aufgabe, einzelne Sätze innerhalb eines Textabschnittes zu markieren. Das Erkennen einzelner Sätze als Sinneinheit ist essentiell für NLP-Applikationen wie beispielsweise das maschinelle Übersetzen von Texten. Die Performanz des SBD-Systems hat direkten Einfluss auf die Performanz der jeweiligen NLP-Anwendungen (vgl. [Wang/Huang 2003, 1]). Die automatisierte Markierung von Sätzen ist nur durch komplexe Verfahren möglich, da Abkürzungen, Aufzählungen oder Dezimalpunkte die Erkennung von Satzendezeichen erschweren. Der folgende Satz verdeutlicht die beschriebene Problematik. ”*Prof. Dr. Pierre Vinken, a 61 year old U.S. citizen, will join the board as a nonexecutive director on Nov. 29. Mr. Vinken is chairman of Elsevier N.V., the Dutch publishing group.*” [Tschuggnall 2014, 20] Existierende SBD-Systeme nutzen aufwändige Regelwerke oder greifen auf Techniken des maschinellen Lernens wie neuronale Netze oder Entscheidungsbäume zurück (vgl. [Wang/Huang 2003, 1]).

Im Rahmen dieser Arbeit wurden verschiedene SBD-Systeme getestet. Der regelbasierte, in *Ruby* implementierte *Pragmatic Segmenter*[13] erzielte hierbei gute Ergebnisse und wurde daher zum Aufbau des benötigten Basiskorpus genutzt. Für eine Implementierung des Ruby-Programms innerhalb des Java-Kontextes kam *JRuby*[14] zur Anwendung. Ziel der SDB-Prozedur ist die Zerlegung der Eingabedokumente in einzelne Sätze. Ein einzelner Satz bildet

[11] http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/
[12] https://github.com/apache/tika
[13] https://github.com/diasks2/pragmatic_segmenter
[14] https://github.com/jruby/jruby

die kleinste Texteinheit, die alle zur Auswertung nutzbaren syntaktischen *features* aufweist. Es wird zudem davon ausgegangen, dass Plagiate in der Regel auf Basis von Sätzen aus den Quelldokumenten übernommen werden und nur selten die Länge eines Satzes unterschreiten. Aus einzelnen Sätzen bestehende Segmente bilden somit die Untergrenze für die im Verlauf der Arbeit festgelegte Segmentierungsgröße.

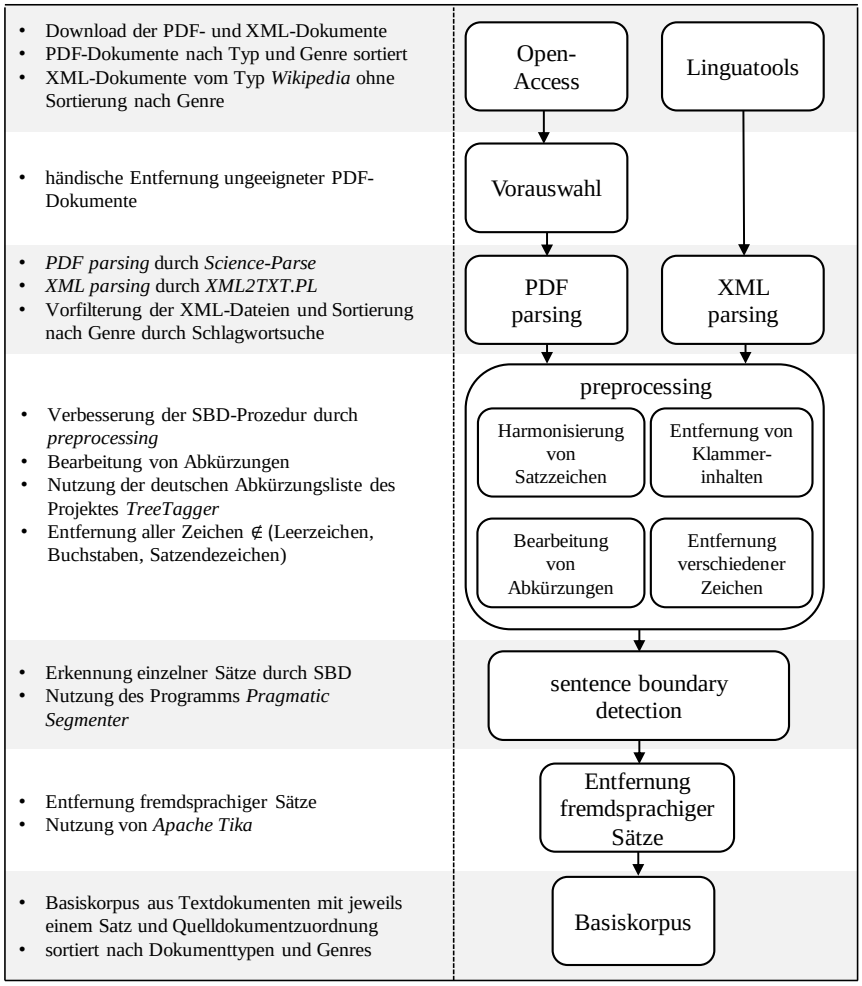


Abbildung 10: Phasen der Datenvorverarbeitung zur Erstellung des Basiskorpus

Die Segmentierungsgröße bestimmt, wie viele Zeichen oder Sätze das Klassifikationsverfahren nutzt, um eine Aussage über den Dokumenttyp des Quelldokuments zu treffen. Die Nutzung einzelner Sätze im Kontext stilometrischer Analysen wurde im Rahmen anderer Forschungsvorhaben erfolgreich evaluiert (vgl. [Muhr et al. 2009, 5; Kuznetsov et al. (2016), 2; Tschuggnall 2014, 20]). Da für die Wahl der optimalen Segmentierungsgröße zahlreiche Evaluierungsschritte notwendig sind, wurde ein Basiskorpus erstellt, aus welchem flexibel Segmente verschiedener Segmentierungsgrößen erzeugt werden können.

Abbildung 10 fasst die in Kapitel 3.4 und 3.5 erläuterten Prozesse zusammen und zeigt die Erstellung des beschriebenen Basiskorpus. Die aus einzelnen Sätzen bestehenden Dokumente können ihren Quelldokumenten zugeordnet und flexibel in Trainings- und Testkorpora verschiedener Segmentierungsgröße überführt werden. Die im Basiskorpus hinterlegten Dokumente sind nach fünf Dokumenttypen mit jeweils fünf Genres gruppiert. Die folgenden Kapitel erläutern unter anderem die Erstellung und Auswertung der genannten Trainings- und Testkorpora.

3.6 Modellierung

Im Kontext der Modellierungsphase erfolgt die Erstellung separater Trainings- und Testkorpora. Durch die gewählten Modellierungstechniken werden Modelle wie zum Beispiel Entscheidungsbäume oder neuronale Netze erzeugt, die zur Lösung der zuvor definierten Problemstellungen beitragen. Verschiedene zur Problemlösung geeignete Modelle werden aus technischer Sicht beurteilt und nach den gewählten Evaluationskriterien vorselektiert (vgl. [Shearer 2000, 17]). Im Rahmen des in der vorliegenden Arbeit verwendeten Klassifikationsverfahrens erfolgt zudem die Erstellung des durch die Modelle genutzten Klassenschemas. Der Teilprozess der Modellierung wird im Kontext dieser Arbeit als *Phase II der Hauptphasen der Verfahrensentwicklung* bezeichnet. Neben der Erstellung von Trainings- und Testkorpora erfolgt die Erzeugung verschiedener Modelle. Letztere werden durch Klassifikationsalgorithmen erstellt. Im Rahmen der technischen Evaluation werden die für einen Anwendungskontext am geeignetsten Modelle ausgewählt (siehe Abbildung 11).

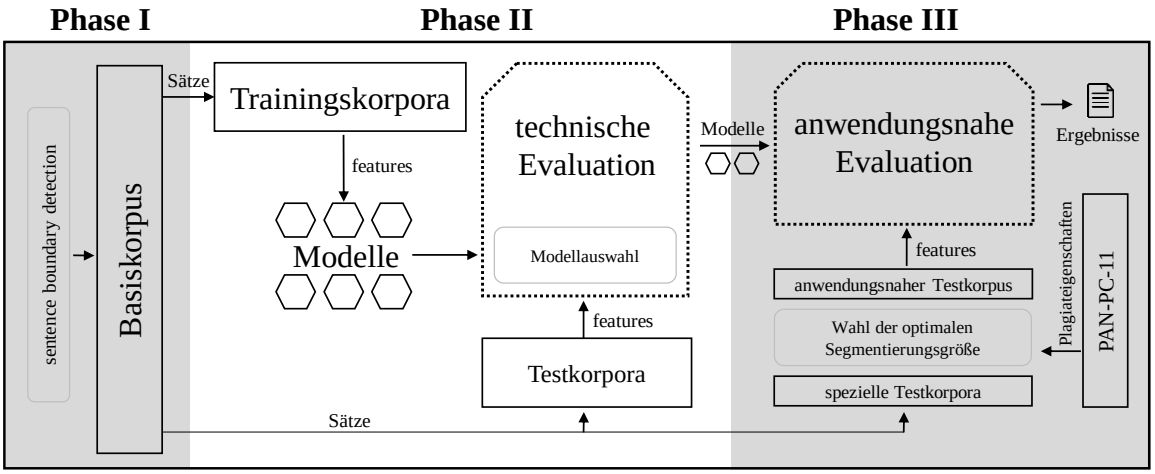


Abbildung 11: *Hauptphasen der Verfahrensentwicklung*, Phase II

3.6.1 Erstellung des Klassenschemas

Wie bereits in Kapitel 2.1.3 beschrieben, ordnen Klassifikationsverfahren Objekte ähnlicher Eigenschaften einer vorgegebenen Klasse zu. Im Kontext des überwachten maschinellen Lernens wird hierbei aus vorgegebenen Objektklassifikationen und Objekteigenschaften eine geeignete Logik abgeleitet, um unbekannte Objekte einzuordnen. Die genannte Vorgabe wird im Rahmen des überwachten maschinellen Lernens als Trainingsdatensatz bezeichnet (vgl. [Aggarwal 2015, 2]). In der vorliegenden Arbeit wird ein Trainingsdatensatz benötigt, welcher die Eigenschaften verschiedener Dokumenttypen erfasst und auf konkrete Dokumenttypklassen abbildet. Die Auswahl der genannten Klassen erfolgt in der Regel durch einen Domänenexperten (vgl. [Yangqiu Song/Dan Roth 2017, 2]). Alternativ kann eine bestehende Ontologie zur Ableitung des Klassenschemas genutzt werden (vgl. [Bedford 2012, 13]).

Für das zu prüfende Verfahren wurde auf die durch *Open-Access* vorgegebene Taxonomie zurückgegriffen. Die gewählten Dokumenttypenklassen müssen sicherstellen, dass die Instanzen innerhalb einer Klasse hinreichend homogen und im Vergleich zu den Instanzen anderer Dokumenttypklassen hinreichend verschieden sind (vgl. [Bock 1970, 36f]), um eine automatisierte Unterscheidung auf Grundlage stilometrischer Eigenschaften zu ermöglichen. Für die Unterscheidung auf technischer, ausschließlich stilometrischer Ebene wurden hierfür die Klassen *Bachelorarbeit*, *Masterarbeit*, *Fachbuch*, *Fachartikel* und *Wikipediaartikel* gewählt sowie jedem im Korpus enthaltenen Dokument zugeordnet. Für alle Dokumente im Trainingskorpus ist somit der reale Dokumenttyp $D_R \in \{Bachelorarbeit, Masterarbeit, Fachbuch, Fachartikel, Wikipediaartikel\}$ definiert. Die im Verlauf der Arbeit erstellten Testkorpora enthalten keine Dokumente, die bereits durch den Trainingskorpus genutzt werden und erfassen ebenfalls den realen Dokumenttyp für jedes Dokument. Dieser wird jedoch ausschließlich zu Evaluationszwecken genutzt und nicht für die Durchführung der Klassifikation. Den Dokumenten im Testkorpus soll durch das Klassifikationsverfahren ein berechneter Dokumenttyp D_Z zugeordnet werden. Kapitel 3.6.3 und 3.6.4 erläutern die Konzepte der Trainings- und Testkorpora im Detail.

Auf Anwendungsebene wurden zudem die Kategorien beziehungsweise Superklassen $C_{neg} = \{Bachelorarbeit, Masterarbeit\}$ und $C_{pos} = \{Fachbuch, Fachartikel, Wikipediaartikel\}$ definiert. Im Zuge der in Kapitel 3.6.9 und 3.7 durchgeführten Analysen wird unbekannten Dokumenten durch das zu prüfende Verfahren ein stilometrisch ermittelter Dokumenttyp D_Z zugeordnet. Anschließend wird geprüft, inwiefern D_Z und der reale Dokumenttyp D_R zur gleichen Kategorie C_{neg} oder C_{pos} gehören. Die beschriebene Kategoriezuordnung entspricht

der Unterscheidung zwischen *positives* und *negatives* beziehungsweise Plagiat und studentischer Abschlussarbeit. Hierbei ist es nicht relevant, zwischen Dokumenttypen innerhalb einer Kategorie zu unterscheiden. Das zu evaluierende Verfahren muss lediglich dokumenttypfremde Abschnitte innerhalb studentischer Abschlussarbeiten identifizieren. Eine direkte Trennung der Dokumente in die Kategorien C_{neg} und C_{pos} auf Ebene des Klassifikationsalgorithmus ist nicht sinnvoll, da diese aus stilometrischer Sicht nicht hinreichend homogen innerhalb ihrer Instanzen sind. Auf eine getrennte Kategorisierung von Bachelor- und Masterarbeiten wurde verzichtet, da die sich ähnelnden Eigenschaften beider Klassen eine automatisierte Unterscheidung erschweren.

Eine Unterscheidung von mehr als zwei Klassen wird als *Multi-Class*-Klassifikationsproblem bezeichnet (vgl. [Lorena et al. 2008, 20]). Auf technischer Ebene wird das beschriebene Problem häufig durch Dekomposition gelöst. Letztere bezeichnet die Aufteilung des *Multi-Class*-Klassifikationsproblems in eine Vielzahl binärer Probleme (vgl. [Ma/Guo 2014, 23f]). Kapitel 3.6.8 erläutert die Auswertung dieser Teilprobleme und die damit in Verbindung stehende Kategoriezuordnung. Wie bereits in Kapitel 2.3.2.2 beschrieben, verwenden intrinsische Methoden ein *One-Class*-Klassenmodell (vgl. [Stein et al. 2011, 63]). Die Nutzung eines *One-Class*-Klassenmodells, in welchem ausschließlich studentische Abschlussarbeiten definiert werden, wurde als Option in Erwägung gezogen. Alle anderen Dokumenttypen würden hierbei als Ausreißer identifiziert. *One-Class*-Klassifikatoren sollten jedoch nur genutzt werden, wenn für weitere Klassen nicht genügend Trainingsdaten zur Verfügung stehen. Liegen für alle Klassen hinreichend Informationen vor, so erreichen binäre oder *Multi-Class*-Klassifikatoren eine höhere Performanz (vgl. [Bellinger et al. 2012, 106]). Abbildung 12 fasst zusammen, wie die durch den Korpus vorgegebene Taxonomie zur Definition des präferierten *Multi-Class*-Klassenmodells genutzt wurde.

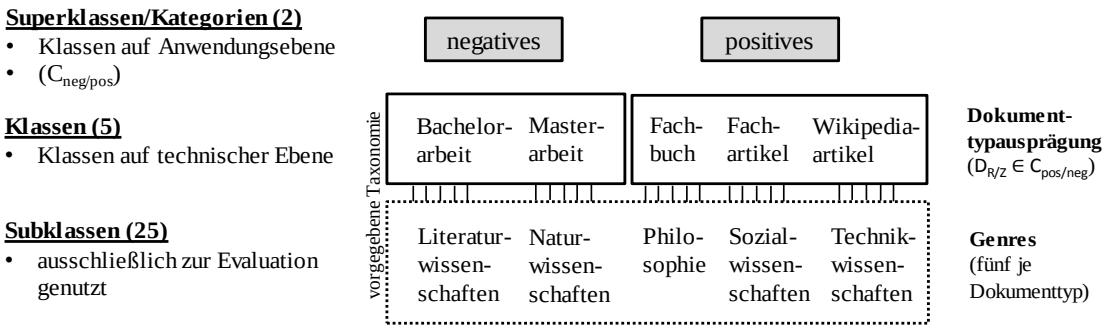


Abbildung 12: Taxonomie des Korpus und abgeleitete Klassen

Die in der Grafik gezeigten Subklassen werden in Kapitel 3.7.1.3 genutzt, um zu prüfen, inwiefern die Performanz des zu evaluierenden Verfahrens durch das Genre der zu prüfenden Dokumente beeinflusst wird. Das folgende Kapitel erläutert die für die Erzeugung der Trainings- und Testkorpora notwendigen Segmentierungsverfahren.

3.6.2 Segmentierungsverfahren

Im Rahmen der Segmentierung wird das Eingabedokument in eine Vielzahl von Segmenten gleicher Länge partitioniert. Jedes Segment wird anschließend überprüft und auf Grundlage seiner Eigenschaften den definierten Klassen und Kategorien zugeordnet. Da die Zuordnung auf Basis der Eigenschaften der Trainingsdokumente durchgeführt wird, muss die Segmentgröße für Trainings- und Testkorpora übereinstimmen.

Für das Einlesen und Partitionieren unbekannter Dokumente wird üblicherweise auf das *Sliding-Window*-Verfahren zurückgegriffen (vgl. [Mothe et al. 2015, 293]). Hierbei wird zunächst ein Ausschnitt fester Länge im Dokument betrachtet und extrahiert. Der nächste zu betrachtende Ausschnitt wird vor seiner Extraktion um eine zuvor definierte Stufengröße verschoben. Dieser Vorgang wird wiederholt, bis das Eingabedokument vollständig segmentiert ist. Die Stufengröße wird in der Regel sehr klein gewählt und beeinflusst über die Gesamtzahl der resultierenden Segmente vor allem die zur Auswertung benötige Rechenkapazität (vgl. [Mothe et al. 2015, 294]). Je kleiner die Stufengröße gewählt wurde, desto besser können die im Anwendungskontext relevanten Strukturen, wie zum Beispiel Plagiate, eingegrenzt beziehungsweise abgegrenzt werden. Im Optimalfall entspricht der betrachtete Ausschnitt exakt der zu findenden Struktur (vgl. [Mothe et al. 2015, 294]). Die Größe des betrachteten Abschnittes beziehungsweise die Segment- oder Fenstergröße wird anhand der Größe der zu findenden Strukturen definiert. Je größer das gewählte Fenster ist, umso mehr Informationen können anschließend für eine Auswertung der Segmenteigenschaften genutzt werden. Übersteigt die Fenstergröße jedoch die Größe der zu findenden Strukturen, so werden letztere durch die umliegenden Informationen verzerrt und gegebenenfalls nicht oder falsch erfasst (vgl. [Mothe et al. 2015, 294]).

Die Wahl der optimalen Segmentgröße im Kontext der Plagiatanalyse gestaltet sich schwierig, da keine Daten zur durchschnittlichen Länge eines Plagiats vorliegen. Die Eigenschaften der für den *PAN-PC-10* erstellten künstlichen und simulierten Plagiate beruhen auf Schätzungen (vgl. [Potthast et al. 2010, 6]). Bestehende auf stilometrischer Auswertung basierende Textanalyseverfahren wählen einzelne Sätze (vgl. [Kuznetsov et al. (2016), 2]), 200 Zeichen (vgl. [Suárez et al. 2010, 3]) oder 40 Wörter als Segmentgröße (vgl. [Eissen/Stein

(2006), 566]). Weitere stilometrische Verfahren nutzen hingegen Fenster zwischen 200 und 1.000 Wörtern (vgl. [Potthast (2011), 3]).[15]

Plagiatanalysesysteme erkennen Plagiate aufgrund des beschriebenen *Sliding-Window*-Verfahrens unter Umständen nicht zusammenhängend, sondern als eine Vielzahl verdächtiger Teilabschnitte. Viele Forschungsarbeiten erheben zur Abbildung dieses Verhaltens ein Granularitätsmaß (vgl. [Potthast et al. 2010, 3]). Der in Abbildung 13 dargestellten Fall verdeutlicht die beschriebene Problematik. Drei reale Plagiatabschnitte (s1, s2, s3) wurden als vier separate Plagiate (r1, r2, r3, r4) markiert (vgl. [Pereira 2010, 43]).

Abbildung 13: Granularität von Plagiatanalyseverfahren [Pereira 2010, 43]

Zur Reduktion der Komplexität wurden im Rahmen dieser Arbeit keine künstlichen Plagiate erzeugt, welche die gewählte Segmentgröße überschreiten. Zudem ist jedes erzeugte Plagiat genau einem Segment zugeordnet. Auf die Berechnung der Granularität und die Anwendung eines *Sliding-Window*-Verfahrens wurde daher verzichtet. Für eine Evaluation im Anwendungskontext (siehe Kapitel 3.7.3) wurde stattdessen ermittelt, wie viele Segmente um welche Satzanzahl von einer für die Klassifikation optimalen Zusammensetzung abweichen. Zudem wurde ein einfaches Segmentierungsverfahren verwendet, bei dem die Stufengröße der Fenstergröße entspricht. Ein *Sliding-Window*-Verfahren ist im Rahmen der verwendeten Evaluationsmethode nicht relevant. Letzteres sollte jedoch für einen Einsatz außerhalb des simulierten Kontextes implementiert werden (siehe Kapitel 3.8).

Für das im Verlauf dieser Arbeit entwickelte Verfahren werden im Rahmen der Modellierungsphase Trainingskorpora mit verschiedenen Segmentgrößen erstellt. Die Performanz der Modelle, die aus den unterschiedlich segmentierten Trainingskorpora resultieren, wird anschließend anhand passender Testkorpora geprüft. Hierbei soll evaluiert werden, inwiefern die durch das Modell ausgewertete Textmenge die Klassifikationsleistung beeinflusst.

[15] In geschriebener deutscher Sprache entspricht ein Wort durchschnittlich 11,7 Zeichen (vgl. [Reginald D. Smith 2012, 11]) und ein Satz durchschnittlich 21 Wörtern (vgl. [Dodd 2006, 166]).

Für die im Rahmen der Modellierungsphase durchgeführte technische Evaluation werden zunächst Testszenarien erstellt, die nur geringfügig von den Eigenschaften der Trainingskorpora abweichen. Die gewählten Segmente sind hierbei stets vollständig aus Sätzen einer einzelnen Klasse zusammengesetzt. Im Rahmen der im Kapitel 3.7 durchgeführten Evaluation werden Eigenschaften für die zum Testen genutzten Segmente ermittelt und umgesetzt, die sich stärker am Anwendungskontext orientieren.

Die folgenden Kapitel beschreiben die Erstellung der Trainings- und Testkorpora auf Grundlage verschiedener Segmentierungsgrößen.

3.6.3 Erstellung der Trainingskorpora

Wie bereits zuvor beschrieben, sind alle Dokumente im Trainingskorpus einer Klasse zugeordnet. Auf Grundlage dieser Zuordnung und der Eigenschaften der jeweiligen Dokumente können Regeln abgeleitet werden, die eine Klassifikation unbekannter Dokumente ermöglichen. Da die Eigenschaften unbekannter Dokumente auf Segmentebene betrachtet werden, muss die Definition eines Segments für Trainings- und Testdatensatz identisch sein. Für jede Segmentierungsvariante muss demnach ein eigener Trainingskorpus erzeugt werden. Insgesamt wurden zunächst fünf Korpora mit jeweils unterschiedlicher Segmentierungsgröße erstellt. Letztere wird über die im Segment enthaltene Anzahl an Sätzen definiert und im Folgenden als $S \in \{1, 5, 10, 15, 20\}$ bezeichnet. Zur Erzeugung der Segmente werden S Sätze je Segment dem in Kapitel 3.5.3 beschriebenen Basiskorpus entnommen. Die gewählten Segmentierungsgrößen orientieren sich an bestehenden auf Stilometrie basierenden Textanalyseverfahren (siehe Kapitel 3.6.2).

Um schon auf Ebene eines einzelnen Segments den Dokumenttyp und nicht einzelne Autoren oder Dokumente abzubilden, wird die Gesamtmenge der entnommenen Sätze vor ihrer Zusammensetzung zufällig durchmischt. Die Durchmischung der Sätze erfolgt nur innerhalb der jeweiligen Klassenzuordnungen. Die resultierende Liste an Sätzen ist für alle Trainingskorpora identisch. Die Segmentierungsgröße bestimmt lediglich, in wie viele Segmente die genannte Liste partitioniert wird. Die auf diese Art erzeugten Segmente sollen die Generalisierungsfähigkeit der aus den Trainingsdaten abgeleiteten Modelle sicherstellen. Da keine realen Plagiate oder Metainformationen für die im Anwendungskontext auftretenden Eingabedaten vorliegen, wurde ein möglichst allgemeines Modell einem spezialisierten Modell mit gegebenenfalls besserer Performanz vorgezogen. Sind die verwendeten Trainingsdaten zu speziell, so tendiert das erzeugte Modell zu einer Überanpassung. Letztere wird in der englischsprachigen Literatur als *overfitting* bezeichnet (vgl. [Larose 2014, 141]). Um die

Generalisierungsfähigkeit der Modelle zu gewährleisten, wurde zudem sichergestellt, dass jede Klasse mit gleicher Häufigkeit im Trainingskorpus vertreten ist[16]. Die Anzahl der für jede Klasse im Trainingskorpus hinterlegten Segmente musste hierfür an jene Klasse angepasst werden, für welche die geringste Anzahl an Sätzen im Basiskorpus hinterlegt ist. Das beschriebene Vorgehen wird in der Forschung als *downsampling* bezeichnet (vgl. [He/Ma 2013, 29f]). Bei ungleicher Ausprägung der Klassen im Trainingskorpus kommt es gegebenenfalls zum *Imbalanced-Classes*-Problem (vgl. [He/Ma 2013, 2]). Viele Klassifikationsalgorithmen versuchen für die Erstellung eines passenden Modells die Anzahl der Fehleinordnungen für die Instanzen des Trainingsdatensatzes zu reduzieren. Bei Verwendung der beschriebenen Optimierungsmetrik werden Minderheitenklassen nur in geringem Maß berücksichtigt (vgl. [He/Ma 2013, 24]).

Die Anzahl der für jedes Genre im Trainingskorpus vorhandenen Segmente ist in Tabelle 2 ablesbar. Auf eine gleichmäßige Ausprägung aller Genres wurde verzichtet, da dies die Anzahl der zur Verfügung stehenden Trainingsinstanzen weiter reduziert hätte und der Einfluss des Genres auf die Klassifikationsleistung als gering eingeschätzt wird. Kapitel 3.7.1.3 evaluiert den Einfluss des Genres auf die Performanz des Verfahrens.

Für die Erstellung und Evaluation der auf dem *feature set* „Writeprints“ (siehe Kapitel 3.6.5) basierenden Modelle mussten die Trainingskorpora für jede Segmentierungsvariante nachträglich verkleinert werden. Insgesamt wurden somit zwei Korpusvarianten je Segmentierungsvariante erzeugt. Bei einer Segmentierungsgröße von 20 Sätzen konnten maximal 599 Segmente je Klasse verarbeitet werden[17]. Da die verkleinerten und nicht verkleinerten Korpora identische strukturelle Eigenschaften aufweisen, wurde auf ein erneutes Erstellen bereits vorhandener Modelle verzichtet. Die auf dem *feature set* „8-Features“ basierenden Modelle (siehe Kapitel 3.6.5) wurden demnach aus unverkleinerten Trainingskorpora abgeleitet. Die auf dem *feature set* „Writeprints“ basierenden Modelle nutzten hingegen verkleinerte Korpora.

Tabelle 2 fasst die Eigenschaften der zehn erstellten Trainingskorpora zusammen. Die verwendeten Segmente sind stets zu 100 % aus den Sätzen einer einzelnen Klasse zusammengesetzt. Die Anwendung von Obfuskationstechniken ist nur im Rahmen der Evaluation

[16] Aufgrund von Rundungen bei der Konvertierung von Gleitkommazahlen in *Integer*-Werte können einzelne Klassen um maximal ein Segment von der vorgegebenen Klassengröße abweichen.
[17] Die Ausführung von *Writeprints* ist sehr rechenaufwändig (vgl. [Brennan et al. 2012, 10]).

sinnvoll, um zu prüfen, inwiefern die Ersetzung oder Löschung von Wörtern Einfluss auf die Klassifikationsleistung hat.

Klassenverteilung	Masterarbeiten: 20,00 %	Bachelorarbeiten: 20,00 % Wikipediaartikel: 20,00 %
	Fachartikel: 20,00 %	Fachbücher: 20,00 %
Genreverteilung	Technikwissenschaften: 24,94 %	Sozialwissenschaften: 16,98 %
	Naturwissenschaften: 16,36 %	Literaturwissenschaften: 22,30 % Philosophie: 19,41 %
Plagiatlängen (Sätze)	S Sätze: 100,00 % S € {1, 5, 10, 15, 20}	
Weitere Eigenschaften	Segmentgröße: S Sätze S € {1, 5, 10, 15, 20}	Satzzusammensetzung: zufällig Obfuskation: nein

Tabelle 2: Eigenschaften der Trainingskorpora

Die Entnahme der Sätze aus dem Basiskorpus wurde so eingegrenzt, dass ausreichend Sätze für verschiedene Testszenarien im Basiskorpus verbleiben. Kapitel 3.6.4 beschreibt das für die Erstellung der Testkorpora angewendete Verfahren.

3.6.4 Erstellung der Testkorpora

Das folgende Kapitel beschreibt zunächst, welches grundlegende Verfahren für die Erstellung aller Testkorpora angewendet wurde. Anschließend wird im Speziellen auf die Erstellung des für die technische Evaluation (siehe Kapitel 3.6.9) genutzten Testszenarios eingegangen. Weitere verwendete Testszenarien werden im späteren Verlauf der Arbeit beschrieben.

Um die Performanz eines Klassifikationsverfahrens zu beurteilen, wird dieses auf anwendungsbezogenen Dokumenten getestet, die dem zugrundeliegenden Modell unbekannt sind (vgl. [Tan et al. 2009, 186ff]). Eine Sammlung der genannten Dokumente wird als Testdatensatz bezeichnet. Wenn die Eigenschaften des Trainingsdatensatzes dem Anwendungskontext entsprechen, kann gegebenenfalls auf die Erstellung eines speziellen Testdatensatzes verzichtet und ein Kreuzvalidierungsverfahren verwendet werden. Hierbei wird das Modell auf jeweils unterschiedlichen Partitionen des Trainingsdatensatzes trainiert und getestet. Dieser Vorgang wird wiederholt, bis jede der gebildeten Partitionen zur Evaluation genutzt wurde. Die Gesamtperformanz wird anschließend aus dem Durchschnitt aller Testdurchläufe errechnet (vgl. [Tan et al. 2009, 187]). Ein Kreuzvalidierungsverfahren konnte jedoch im Rahmen dieser Arbeit nicht verwendet werden, da alle genutzten Testszenarien spezielle Anwendungsfälle darstellen und somit von den allgemein gehaltenen Eigenschaften der Trainingskorpora abweichen.

Die im Verlauf der Arbeit beschriebenen Testdatensätze wurden aus jenen Dokumenten erzeugt, die nach der Erstellung der Trainingskorpora im Basiskorpus verblieben. Insgesamt

wurden 90 % der Dokumente für die Trainingskorpora und 10 % der Dokumente für die Testkorpora genutzt. Das beschriebene Vorgehen wird als *Hold-Out*-Methode bezeichnet (vgl. [Tan et al. 2009, 186]). Die gewählten Größen entsprechen der üblichen Evaluationsmethodik im Kontext von Data-Mining-Vorhaben (vgl. [Francillon/Rohatgi 2014, 104]).

Im Verlauf der Modellierungsphase erfolgt zunächst eine Evaluation der erzeugten Modelle aus technischer Sicht. Auf Basis der Testergebnisse kann anschließend eine Vorauswahl der geeignetsten Modelle erfolgen. Die Erstellung der für die technische Evaluation benötigten Testkorpora folgt der in Kapitel 3.6.3 beschriebenen Methodik. Für jede Segmentierungsgröße wurde ein passender Testkorpus erstellt. Da die genannten Testkorpora deutlich weniger Dokumente enthalten als die entsprechenden Trainingskorpora, konnte auf die Erstellung verkleinerter Testkorpusvarianten verzichtet werden. Insgesamt wurden somit fünf Testkorpora erzeugt.

Klassenverteilung	Masterarbeiten: 25,00 %	Bachelorarbeiten: 25,00 %	Wikipediaartikel: 16,66 %
	Fachartikel: 16,66 %	Fachbücher: 16,66 %	
Genreverteilung	Technikwissenschaften: 24,94 %	Sozialwissenschaften: 16,98 %	
	Naturwissenschaften: 16,36 %	Literaturwissenschaften: 22,30 %	Philosophie: 19,41 %
Plagiatlängen (Sätze)	S Sätze: 100,00 % S € {1, 5, 10, 15, 20}		
Weitere Eigenschaften	Segmentgröße: S Sätze S € {1, 5, 10, 15, 20}	Satzzusammensetzung: zufällig	Obfuskation: nein

Tabelle 3: Eigenschaften der für die technische Evaluation verwendeten Testkorpora

Da auf Evaluationsebene erfasst werden soll, inwiefern das Klassifikationsverfahren zur Unterscheidung der Kategorien C_{neg} und C_{pos} geeignet ist, wurden die auf technischer Ebene definierten Klassen so verkleinert, dass beide Kategorien zu gleichen Teilen im jeweiligen Testkorpus vertreten sind. Die Performanz kann somit unabhängig von der Verteilung genannter Kategorien betrachtet werden. Tabelle 3 fasst die Eigenschaften der erzeugten Testkorpora zusammen.

Das für die technische Evaluation erzeugte Testszenario entspricht somit einem zu 50 % plagiierten Eingabedokument. Alle Dokumenttypen sind innerhalb der definierten Kategorien zu gleichen Teilen vertreten. Eine Mischung von Bachelor- und Masterarbeitssegmenten wäre im realen Kontext als Plagiat zu deuten. Im Rahmen dieser Arbeit werden jedoch beide Dokumenttypen aufgrund ihrer ähnlichen Eigenschaften innerhalb einer Kategorie evaluiert. Die Zusammensetzung der Sätze erfolgte nach der in Kapitel 3.6.3 beschriebenen Methodik. Eine zufällige Anordnung der Sätze entspricht nicht dem Anwendungskontext, erzeugt jedoch einen in Bezug auf die Originaldokumente durchschnittlicheren Testfall. Für die in Kapitel 3.7 durchgeführte Evaluation werden spezifischere Testkorpora genutzt, die

sich näher am Anwendungskontext orientieren. Aufgrund von Größe und Anzahl der für die technische Evaluation genutzten Testkorpora wurde auf eine rechenintensive Obfuskation der Segmente verzichtet. Die folgenden Kapitel beschreiben die für die Erstellung der Modelle gewählten *feature sets* und Klassifikationsalgorithmen.

3.6.5 Verwendete Feature Sets

Die aus den Segmenten abgeleiteten Eigenschaften werden als *features* bezeichnet. Eine Vielzahl definierter Eigenschaften bildet ein *feature set*. Durch letztere werden die vorhandenen Daten in eine mathematisch auswertbare Form überführt. Die Auswahl beziehungsweise die Bildung von *feature sets* zählt zu den wichtigsten Aufgaben im Kontext von Klassifikationsvorhaben. Aufgrund der hohen Dimensionalität von Texteigenschaften ist die *Feature*-Auswahl im Rahmen von *Text-Mining*-Aufgaben besonders relevant (vgl. [Aggarwal/Zhai 2012, 167]). Für das in dieser Arbeit zu evaluierende Verfahren werden etablierte stilometrische *feature sets* verwendet, welche lexikalische, syntaktische, inhaltliche und idiosynkratische Eigenschaften von textuellen Daten abbilden. Lexikalische *features* quantifizieren Eigenschaften auf Ebene des im Text verwendeten Wort- und Zeichensatzes. Syntaktische *features* analysieren grammatikalische Eigenschaften des Textes und idiosynkratische *features* quantifizieren spezifische Besonderheiten wie Rechtschreibfehler.

Im Rahmen der technischen Evaluation werden zwei verschiedenen *feature sets* zur Erfassung stilometrischer Eigenschaften getestet. Das durch Brennan et al. entwickelte *feature set* „Basic-9“ nutzt ausschließlich lexikalische Features und kann aufgrund des geringen Berechnungsaufwandes auch für große Datenmengen verwendet werden (vgl. [Brennan et al. 2012, 20]). Lexikalische *features* sind weitestgehend sprach- und themenunabhängig und werden daher sehr häufig für stilometrische Aufgabenstellungen verwendet (vgl. [Dregvaite/Damasevicius 2015, 435]). Tabelle 4 zeigt alle der in *Basic-9* erfassten *features*. Im Rahmen der vorliegenden Arbeit wurde *Basic-9* auf acht *features* reduziert. Eine Erfassung der Satzanzahl ist nicht notwendig, da jedes Segment eine identische Anzahl an Sätzen aufweist. In Anlehnung an die Java-Implementierung des *feature sets* in *JStylo* (siehe Kapitel 3.6.7) wird daher im weiteren Verlauf der Arbeit die Bezeichnung *8-Features* verwendet.

Writeprints wurde durch Abassi und Chen (vgl. [Abbasi/Chen 2008, 7]) auf Basis der Vorarbeiten von Zhen et al. (vgl. [Zheng et al. 2006, 378]) entwickelt und war ursprünglich als vollständiges *Authorship-Attribution*-Verfahren konzipiert. Im Rahmen stilometrischer Forschung wird vor allem das von *Writeprints* bereitgestellte *feature set* verwendet (vgl. [Dolev/Lodha 2017, 120; Juola et al. 2013, 394; Brennan et al. 2012, 10]).

Ebene	Feature
lexikalisch	Anzahl an Inhaltswörtern
lexikalisch	Anzahl an Inhaltswörtern in Relation zur Anzahl aller Wörter (lexikalische Dichte)
lexikalisch	Anzahl der Sätze
lexikalisch	durchschnittliche Satzlänge (in Wörtern)
lexikalisch	durchschnittliche Silbenanzahl je Wort
lexikalisch	Gunning-Fog Readabillity Index
lexikalisch	Flesch Reading Ease Score
lexikalisch	Anzahl an Zeichen
lexikalisch	Anzahl an Buchstaben

Tabelle 4: Von *Basic-9* erfasste *features*[18] (in Anlehnung an [Brennan et al. 2012, 9])

Ebene	Features
lexikalisch	Wortanzahl, durchschnittliche Wortlänge, Anzahl langer Wörter, Anzahl an Inhaltswörtern
lexikalisch	Anzahl der Zeichen, Anteil der Zahlen, Anteil der Buchstaben, Anteil der Großbuchstaben
lexikalisch	Häufigkeiten einzelner Buchstaben
lexikalisch	Häufigkeiten einzelner Buchstabenbigramme
lexikalisch	Häufigkeiten einzelner Buchstabentrigramme
lexikalisch	Häufigkeiten einzelner Wortlängen
lexikalisch	verschiedene Kennzahlen zur Messung der lexikalischen Diversität (zum Beispiel Simpson Diversity Index)
lexikalisch	Häufigkeiten einzelner Sonderzeichen
lexikalisch	Häufigkeiten einzelner Zahlen
lexikalisch	Häufigkeiten einzelner Zahlenbigramme
lexikalisch	Häufigkeiten einzelner Zahlentrigramme
syntaktisch	Häufigkeiten einzelner Funktionswörter
syntaktisch	Häufigkeiten einzelner Satzzeichen
syntaktisch	Häufigkeiten einzelner POS tags
syntaktisch	Häufigkeiten einzelner POS-Tag-Bigramme
syntaktisch	Häufigkeiten einzelner POS-Tag-Trigramme
inhaltlich	Häufigkeiten einzelner Wörter
inhaltlich	Häufigkeiten einzelner Wortbigramme
inhaltlich	Häufigkeiten einzelner Worttrigramme
idiosynkratisch	Häufigkeiten einzelner, typischer Schreibfehler

Tabelle 5: Von *Writeprints* erfasste *features*[19] (in Anlehnung an [Stolerman 2012, 4])

Writeprints zählt zu den performantesten stilometrischen Verfahren, ist aber im Vergleich zu anderen Methoden sehr rechenaufwändig (vgl. [Brennan et al. 2012, 10]). Die Implemen-

[18] Der *Gunning-Fog Readabillity Index* (vgl. [Gunning 1969, 3]) sowie der *Flesch Reading Ease Score* (vgl. [Flesch 1948, 221]) versuchen, die Lesbarkeit eines Textes auf Grundlage lexikalischer Eigenschaften zu erfassen.

[19] Der *Simpson Diversity Index* ist eine Kennzahl zur Erfassung der lexikalischen Diversität (vgl. [Simpson 1949, 688])

tierung von *Writeprints* in *JStylo* erfolgte durch Stolerman (vgl. [Stolerman 2012, 4]). Tabelle 5 zeigt die durch *Writeprints* erfassten *features*. *Writeprints* erfasst neben lexikalischen *features* auch syntaktische, inhaltliche und idiosynkratische Eigenschaften des Textes und muss daher für jede Sprache spezifisch angepasst werden. Da bigramm- und trigrammbasierte *features* die Dimensionalität des *feature sets* stark erhöhen, werden diese in der Praxis auf die 50 relevantesten Ausprägungen begrenzt (vgl. [McDonald et al. 2012, 306]). Die Relevanz eines *features* wird im Rahmen eines automatisierten *Feature-Selection*-Prozesses auf Basis des entropiebasierten Maßes *information gain* bestimmt (vgl. [Stolerman 2012, 3; Liu/Motoda 1998, 162]). In der vorliegenden Arbeit wurde eine für deutschsprachige Dokumente angepasste und in *JStylo* implementierte Variante von *Writeprints* verwendet. Auf die Erfassung von *features*, welche ausschließlich die Häufigkeiten der im Text auftretenden Zahlen analysieren, wurde verzichtet, da letztere im Kontext des *Preprocessing*-Schrittes aus allen Eingabetexten entfernt wurden.

3.6.6 Verwendete Klassifikationsalgorithmen

Klassifikationsalgorithmen verarbeiten die durch die *feature sets* aus den Trainings- und Testkorpora extrahierten Eigenschaften. Die in den Trainingsdaten vorhandenen Klassenzuordnungen werden genutzt, um Regeln abzuleiten, auf deren Grundlage die in den Testdaten vorhandenen Instanzen klassifiziert werden können. Die abgeleitete Menge an Regeln wird als Modell bezeichnet (vgl. [Tiwari et al. 2017, 5]).

Im Rahmen der technischen Evaluation wird die Performanz der durch verschiedene Klassifikationsalgorithmen erzeugten Modelle gegenübergestellt. Das folgende Kapitel vermittelt ein Grundverständnis für die Funktionsweise der gewählten Algorithmen. Im Kontext stilometrischer Aufgabenstellungen erzielen *Naive Bayes*, *Support Vector Machines* (SVM) und *C4.5* gute Ergebnisse (vgl. [Dolev/Lodha 2017, 120; Hadjidj et al. 2009, 130; Dewan et al. (2014), 10]). Die genannten Verfahren zählen nach Wu et al. zu den wichtigsten Data-Mining-Algorithmen (vgl. [Wu et al. 2008, 1]) und werden aufgrund ihrer grundverschiedenen Lösungsstrategien häufig gegenübergestellt (vgl. [Lu 2013; Gelbukh et al. 2014]).

C4.5 zählt zu den auf Entscheidungsbäumen basierenden Algorithmen (vgl. [Aggarwal/Zhai 2012, 176ff]) und wird seit 1992 stetig weiterentwickelt (vgl. [Quinlan 2014, VII]). Auf Grundlage der Trainingsdaten wird ein Baum an Entscheidungsregeln erzeugt, dessen Endknoten die jeweilige Klasse der Eingabeinstanz bestimmen (vgl. [Tan et al. 2009, 150]). Abbildung 14 zeigt, wie anhand von Eigenschaften eine Zuordnung zu *Säugetieren* oder *Nicht-Säugetieren* erfolgt. Die Ebene eines Entscheidungsknotens innerhalb des Baumes

wird auf Grundlage von Entropieberechnungen bestimmt (vgl. [Quinlan 2014, 21; Liu/Motoda 1998, 162]).

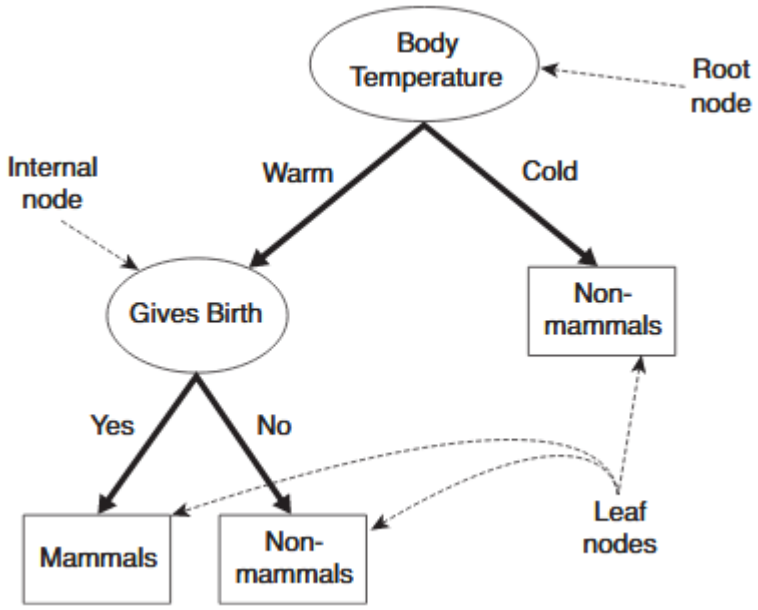


Abbildung 14: Beispiel für einen Entscheidungsbaum [Tan et al. 2009, 151]

Naive Bayes zählt zu den probabilistischen Klassifikatoren (vgl. [Aggarwal/Zhai 2012, 181]) und findet erstmals in den Arbeiten von Maron und Kuhns Anwendung (vgl. [Maron/Kuhns 1960, 216]). Der Name des Verfahrens leitet sich aus dessen Annahmen bezüglich der Eigenschaften des Trainingsdatensatzes ab. Naive-Bayes-Klassifikatoren gehen aus Gründen der Vereinfachung davon aus, dass alle *features* statistisch unabhängig voneinander auftreten und jedes definierte *feature* die gleiche Relevanz für eine Klassenzuordnung hat. Obwohl beide Annahmen im realen Kontext nur selten erfüllt sind, erzielt das Verfahren bei einer ausreichenden Anzahl an Trainingsinstanzen gute Ergebnisse. *Naive Bayes* bestimmt anhand der Trainingsdaten, mit welcher Häufigkeit die Ausprägung eines einzelnen *features* in jeder Klasse vorhanden ist. Aus diesen Häufigkeiten kann anschließend für jede Kombination aus *features* eine Wahrscheinlichkeit berechnet werden, mit der die genannte *Feature*-Kombination zu einer bestimmten Klassenzuordnung führt (vgl. [Nédellec et al. 1998, 4ff]).

SVM werden den funktionsbasierten Klassifikatoren zugeordnet (vgl. [Witten et al. 2017b, 35]) und seit 1964 stetig weiterentwickelt (vgl. [Wang 2005, 2]). In ihrer heutigen Form wurden *SVM* erstmals 1992 angewendet (vgl. [Boser et al. 1992, 144]). Das Verfahren trennt die Daten des Trainingsdatensatzes über eine Hyperebene in zwei Bereiche. Da somit nur zwei Klassen unterschieden werden können, müssen *Multi-Class*-Problemstellungen in eine Vielzahl von binären Problemen geteilt werden (vgl. [Ma/Guo 2014, 23f]). Abbildung 15

zeigt, wie die genannte Hyperebene die grün und blau dargestellten Instanzen voneinander trennt.

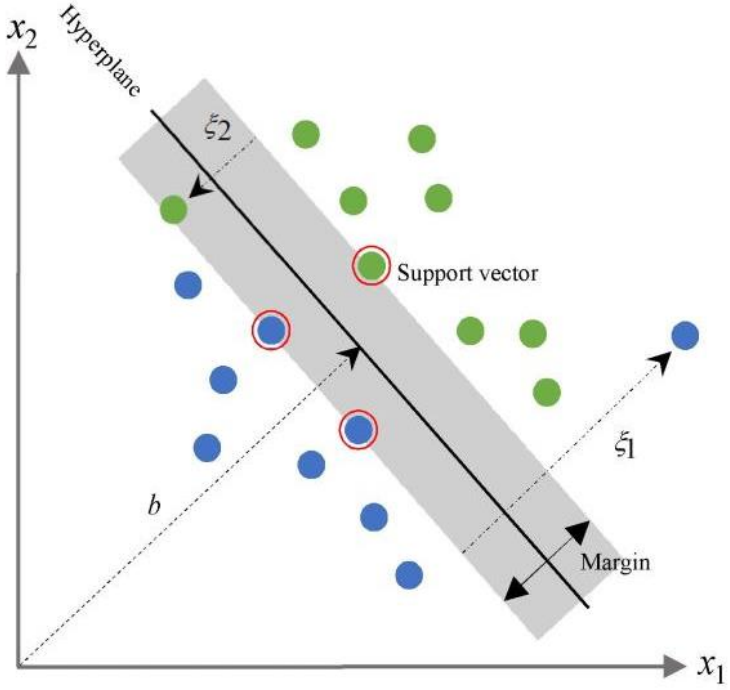


Abbildung 15: Funktionsweise einer *Support Vector Machine* [Nanda et al. 2018, 5]

Um die Position *b* der Hyperebene zu bestimmen, werden im Wesentlichen zwei Optimierungsprobleme gelöst. Der Abstand (*Margin*) von mindestens zwei Vektoren unterschiedlicher Klassenzuordnung zur Hyperebene soll maximiert werden. Die beschriebenen Vektoren sind in der Grafik rot markiert. Gleichzeitig sollen die Kosten, die durch Vektoren entstehen, welche die erstellte Trennung verletzten, minimiert werden. Die beschriebenen Kosten werden in der Grafik über die Länge der mit *ξ1* und *ξ2* beschrifteten Pfeile abgebildet (vgl. [Nanda et al. 2018, 4f]). Durch die Maximierung des Abstands zur Trennebene wird das Risiko einer Überanpassung des Modells reduziert [Wang/Lin 2015, 187].

Das folgende Kapitel erläutert die Anwendung der beschriebenen Algorithmen zur Erstellung der Klassifikationsmodelle.

3.6.7 JStylo und Modellerstellung

Für das Einlesen der Trainings- und Testdatensätze, das Extrahieren der Features und die Erstellung der Modelle über die Klassifikationsalgorithmen wurde *JStylo*[20] verwendet. *JStylo* ist ein quelloffenes, an der *Drexel University* in Philadelphia entwickeltes

[20] https://github.com/psal/jstylo

Java-Programm, das für Forschungsarbeiten im Kontext der *authorship attribution* verwendet wird (vgl. [Afroz et al. (2014), 212; McDonald et al. 2012, 303; Brennan et al. 2012, 9]). Es basiert auf dem *Java Graphical Authorship Attribution Program* (JGAAP)[21] und unterstützt Nutzer bei allen Schritten, die im Kontext stilometrischer Klassifikationsvorhaben relevant sind (vgl. [Stolerman 2015, 40f]). Die durch *JStylo* ausgegebene Konfusionsmatrix (siehe Kapitel 3.6.8) zeigt die Performanz der Modelle auf dem eingelesenen Testdatensatz. Für die Erstellung der Modelle integriert *JStylo* das *Waikato Environment for Knowledge Analysis* (WEKA) [22]. *WEKA* ist eine Kollektion von Algorithmen für maschinelles Lernen und *Preprocessing*-Werkzeugen, die an der Universität von Waikato in Neuseeland entwickelt wurde (vgl. [Witten et al. 2017b, 7]).

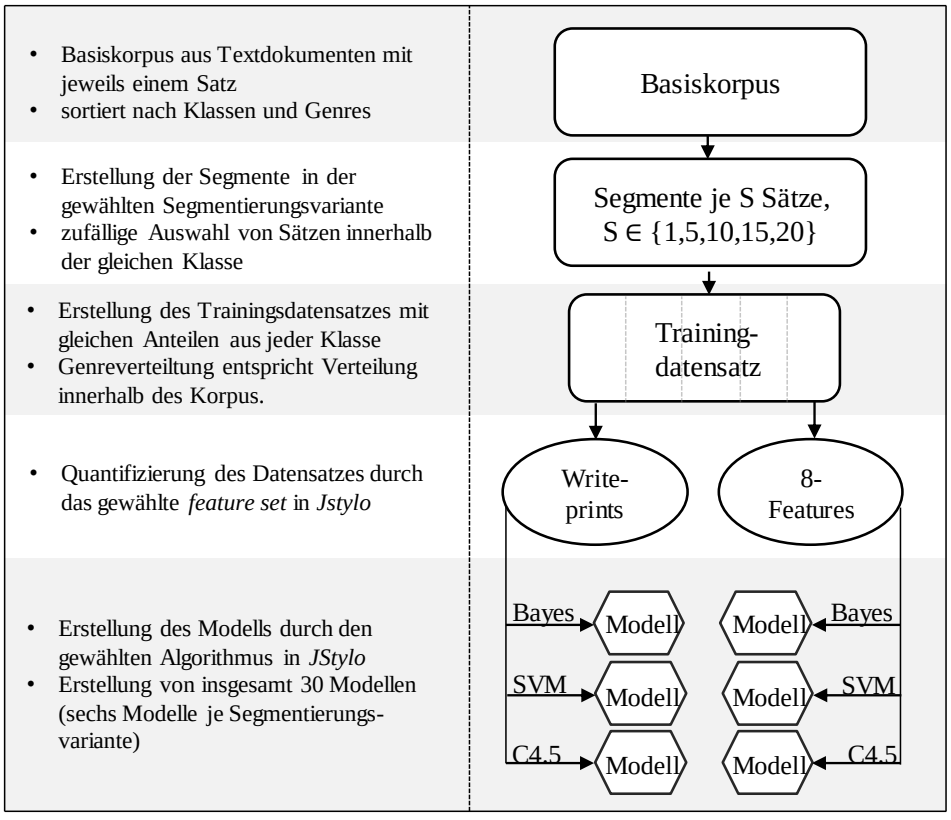


Abbildung 16: Phasen der Trainingsdaten- und Modellerstellung

Die WEKA-Implementierung von *C4.5* wird als *J48* bezeichnet. *SVM* wurde unter der Bezeichnung *SMO* implementiert (vgl. [Witten et al. 2017b, 35ff]). Für die Erstellung der Modelle wurden die in *JStylo* vorgegebenen Standardparameter zur Konfiguration der Klassifikationsalgorithmen genutzt.

[21] https://github.com/evllabs/JGAAP
[22] https://www.cs.waikato.ac.nz/ml/weka/

Abbildung 16 zeigt die zwischen Trainingsdatensatz- und Modellerstellung durchgeführten Teilschritte. Die Erstellung der Testdatensätze erfolgte separat nach der in Kapitel 3.6.4 erläuterten Methodik und im Kontext der anwendungsbezogenen Evaluation. Für jede der fünf Segmentierungsvarianten des Trainingskorpus wurden sechs Modelle erzeugt. Die im Rahmen der technischen Evaluation ausgewählten Modelle werden unverändert übernommen und im weiteren Verlauf der Arbeit in verschiedenen, anwendungsbezogenen Testszenarien evaluiert.

Das folgende Kapitel erläutert die in allen Evaluationsschritten genutzten Performanzmetriken, die durch *JStylo* ausgegebenen Konfusionsmatrizen und die für die Bezeichnung der Modelle genutzten Parameter.

3.6.8 Performanzmetriken und Parameter

Die erstellten Modelle unterscheiden sich hinsichtlich der Segmentierungsgröße des genutzten Trainingsdatensatzes, des verwendeten *feature sets* und bezüglich der zur Erstellung genutzten Klassifikationsalgorithmen. Für die Bezeichnung der Modelle gelten daher die im Folgenden beschriebenen Parameter.

Jedem Modell $M_{S,FS,A}$ ist eine Segmentierungsgröße $S \in \{1, 5, 10, 15, 20\}$, ein *feature set* $FS \in \{WP, 8F\}$ und ein Algorithmus $A \in \{NB, C4.5, SVM\}$ zugeordnet. *WP* steht hierbei für *Writeprints* und *8F* für *8-Features*. *Naive Bayes* wird als *NB* bezeichnet.

D_R | D_Z	Bachelorarbeit	Masterarbeit	Fachbuch	Fachartikel	Wikipediaartikel	N_{Test} [D_R]	N_{Test} [C_{neg}/C_{pos}]	N_{Test} [gesamt]
Bachelorarbeit	299	78	13	9	3	402	804	1608
Masterarbeit	109	229	42	21	1	402		
Fachbuch	5	41	154	66	2	268	804	
Fachartikel	6	16	46	200	0	268		
Wikipediaartikel	0	0	1	0	267	268		

Tabelle 6: Konfusionsmatrix für das Modell $M_{20,WP,SVM}$,
true negatives (grün), *false negatives* (blau), *false positives* (gelb), *true positives* (rot)

Die farbig markierten Zellen von Tabelle 6 zeigen die durch *JStylo* ausgegebene Konfusionsmatrix für ein Beispielmodell. Zusätzlich sind die identischen Ausprägungen von C_{neg} und C_{pos} sowie die Gleichverteilung der Klassen innerhalb der Kategorien erkennbar. Wie bereits in Kapitel 3.6.1 beschrieben, können *Multi-Class*-Klassifikationsprobleme in eine

Vielzahl binärer Teilprobleme aufgeteilt werden. Im Kontext des in dieser Arbeit entwickelten Verfahrens sind nur jene Teilprobleme relevant, welche zur Unterscheidung zwischen den Kategorien C_{neg} und C_{pos} beitragen. Die grün dargestellten Teilunterscheidungen werden hierbei als *true negatives* (TN) interpretiert. Letztere bezeichnen somit jene Segmente, die durch das entwickelte Verfahren korrekt als Abschnitt einer studentischen Abschlussarbeit erkannt wurden. Die blau markierten Teilprobleme zeigen hingegen *false negatives* (FN), also jene Segmente, die das Verfahren fälschlicherweise als Abschnitt einer studentischen Abschlussarbeit markiert hat. Die gelb dargestellten Teilprobleme werden als *false positives* (FP) interpretiert. *False positives* sind Segmente, die fälschlicherweise als Plagiat erkannt wurden. Die rot abgebildeten Teilprobleme zeigen *true positives* (TP), also jene Segmente, die korrekt als Plagiat markiert wurden. Die Anzahl der *TP* und *TN* sollte möglichst hoch sein, während die Anzahl der *FP* und *FN* möglichst gering sein sollte (vgl. [Guillet/Hamilton 2007, 282]).

Aus den genannten Kennzahlen können verschiedene Performanzmetriken berechnet werden. Die Fehlerrate *E* beschreibt im Rahmen dieser Arbeit die Größe des Anteils an Segmenten mit falscher Zuordnung und wird über die Formel

$$E = \frac{FP + FN}{FP + FN + TP + TN} = \frac{FP + FN}{N_{Test}}$$

berechnet. N_{Test} bezeichnet die Gesamtanzahl der verwendeten Testsegmente. Die Gesamtanzahl der verwendeten Trainingssegmente wird im weiteren Verlauf der Arbeit als N_{Train} bezeichnet. Sind ausreichend Testsegmente vorhanden, so kann die Fehlerrate als Durchschnitt betrachtet und somit als Grundlage für die Berechnung eines Konfidenzintervalls genutzt werden (vgl. [Roiger 2017, 231]). Hierfür muss zunächst die Standardabweichung

$$SE = \sqrt{\frac{E * (1 - E)}{N_{Test}}}$$

ermittelt werden. Das 95 % Konfidenzintervall für die Fehlerrate bezeichnet den Bereich zwischen $E - 2SE$ und $E + 2SE$ und beschreibt auf statistischer Basis, welche Werte die Fehlerrate für die Grundgesamtheit annehmen kann, wenn letztere durch den genutzten Testkorpus repräsentiert wird (vgl. [Roiger 2017, 231]). Die Fehlerrate kann als Qualitätsmaß für Klassifikationsverfahren genutzt werden, wenn die für die Berechnung genutzten Klassen zu gleichen Teilen im Testdatensatz vertreten sind. Bei ungleicher Verteilung der Klassen spiegelt die Fehlerrate nicht die anwendungsbezogene Performanz des Verfahrens wider.

He und Ma (vgl. [He/Ma 2013, 2]) beschreiben in diesem Kontext ein Beispiel, in welchem ein Klassifikationsverfahren erkrankte Patienten stets als gesund einstuft. Da erkrankte Patienten nur einen kleinen Teil des Gesamtdatensatzes ausmachen, fällt die Fehlerrate gering aus, obwohl das eingesetzte Verfahren für die Erkennung von Erkrankungen unbrauchbar ist.

Als Standardmetriken für die Messung der Performanz im Rahmen von Klassifikationsvorhaben werden daher *Recall*-, *Precision*- und F-Werte eingesetzt (vgl. [Guillet/Hamilton 2007, 140f]). Der auch als Sensitivität bezeichnete *Recall*-Wert beschreibt im Kontext der vorliegenden Arbeit den Anteil der korrekt erkannten Plagiate an allen im Testkorpus vorhandenen Plagiaten und wird über die Formel

$$R = \frac{TP}{TP + FN}$$

berechnet. Der auch als Genauigkeit bezeichnete *Precision*-Wert beschreibt hingegen den Anteil der korrekt erkannten Plagiate an allen erkannten Plagiaten und wird somit als

$$P = \frac{TP}{TP + FP}$$

errechnet. In der Praxis muss häufig zwischen hoher Sensitivität und hoher Genauigkeit abgewogen werden (vgl. [Cios et al. 2007, 458]). Um beide beschriebenen Metriken in einer Kennzahl zusammenzufassen, werden F-Werte genutzt. Je nach Anwendungsfall können *Precision*- und *Recall*-Wert hierfür unterschiedlich gewichtet werden. Der über die Formel

$$F_1 = 2 * \frac{P * R}{P + R}$$

berechnete F_1-Wert bildet ein harmonisches Mittel beider Werte und nähert sich seinem Maximum von $F_1 = 1$ somit nur, wenn beide Metriken möglichst hoch sind (vgl. [Guillet/Hamilton 2007, 140f]). Dies gilt ebenso für den als

$$F_2 = 5 * \frac{P * R}{4P + R}$$

berechneten F_2-Wert. Letzterer gewichtet jedoch einen hohen *Recall*-Wert stärker als einen hohen *Precision*-Wert (vgl. [Mahmoud et al. (2012), 187]). Der F_2-Wert findet daher beispielsweise Anwendung in der Domäne der Schadsoftwareerkennung, in welcher die Identifikation möglichst vieler Viren relevanter ist als die eventuelle Fehlklassifikation von virenfreien

Dateien (vgl. [Gavrilut et al. (2009), 735]). In den folgenden Kapiteln wird das im Kontext dieser Arbeit entwickelte Verfahren auf Grundlage der beschriebenen Performanzmetriken evaluiert.

3.6.9 Technische Evaluation

Im Rahmen dieses Kapitels werden die für den Anwendungskontext am besten geeigneten Modelle ausgewählt. Der Auswahlprozess beschränkt sich hierbei auf die Festlegung des *feature sets* und des genutzten Algorithmus, da für eine Beurteilung der optimalen Segmentierungsgröße weitere, anwendungsbezogenere Evaluationsschritte notwendig sind.

Modell	**N_{Train}**	**N_{Test}**	**E (Prozent)**	**2*SE (Prozentpunkte)**	**R**	**P**	**F_1**
$M_{1,WP,NB}$	56995	27326	40,94	0,60	0,407	0,643	**0,499**
$M_{1,WP,C4.5}$	56995	27326	45,57	0,60	0,613	0,539	**0,574**
$M_{1,WP,SVM}$	56995	27326	35,57	0,58	0,665	0,639	**0,652**
$M_{1,8F,NB}$	273260	27326	47,53	0,60	0,674	0,519	**0,587**
$M_{1,8F,C4.5}$	273260	27326	44,65	0,60	0,619	0,547	**0,581**
$M_{1,8F,SVM}$	273260	27326	45,67	0,60	0,764	0,530	**0,626**
$M_{5,WP,NB}$	11395	6439	33,75	1,18	0,547	0,711	**0,618**
$M_{5,WP,C4.5}$	11395	6439	36,76	1,20	0,722	0,612	**0,663**
$M_{5,WP,SVM}$	11395	6439	23,93	1,06	0,813	0,736	**0,772**
$M_{5,8F,NB}$	64390	6439	40,78	1,23	0,506	0,611	**0,554**
$M_{5,8F,C4.5}$	64390	6439	44,99	1,24	0,640	0,542	**0,587**
$M_{5,8F,SVM}$	64390	6439	40,50	1,22	0,826	0,565	**0,671**
$M_{10,WP,NB}$	5695	3228	22,71	1,48	0,765	0,778	**0,771**
$M_{10,WP,C4.5}$	5695	3228	32,59	1,65	0,652	0,682	**0,667**
$M_{10,WP,SVM}$	5695	3228	15,89	1,29	0,843	0,840	**0,841**
$M_{10,8F,NB}$	32280	3228	37,79	1,71	0,546	0,644	**0,591**
$M_{10,8F,C4.5}$	32280	3228	39,44	1,72	0,704	0,588	**0,641**
$M_{10,8F,SVM}$	32280	3228	36,31	1,69	0,848	0,596	**0,700**
$M_{15,WP,NB}$	3795	2148	20,53	1,74	0,811	0,785	**0,798**
$M_{15,WP,C4.5}$	3795	2148	28,59	1,95	0,751	0,699	**0,724**
$M_{15,WP,SVM}$	3795	2148	12,99	1,45	0,879	0,864	**0,871**
$M_{15,8F,NB}$	21480	2148	34,36	2,05	0,602	0,675	**0,637**
$M_{15,8F,C4.5}$	21480	2148	39,90	2,11	0,708	0,583	**0,639**
$M_{15,8F,SVM}$	21480	2148	33,10	2,03	0,864	0,622	**0,723**
$M_{20,WP,NB}$	2995	1608	16,79	1,86	0,815	0,844	**0,829**
$M_{20,WP,C4.5}$	2995	1608	24,13	2,13	0,833	0,725	**0,775**
$M_{20,WP,SVM}$	2995	1608	9,76	1,48	0,915	0,892	**0,904**
$M_{20,8F,NB}$	16080	1608	29,85	2,28	0,613	0,745	**0,673**
$M_{20,8F,C4.5}$	16080	1608	33,33	2,35	0,781	0,636	**0,701**
$M_{20,8F,SVM}$	16080	1608	27,36	2,22	0,902	0,668	**0,767**

Tabelle 7: Performanz der erstellten Modelle im Kontext der technischen Evaluation

Die genutzten Trainings- und Testkorpora haben die in Tabelle 2 und Tabelle 3 beschriebenen Eigenschaften. Tabelle 7 zeigt die Performanz aller erstellten Modelle auf den genannten Testkorpora. Die vollständigen Konfusionsmatrizen des durchgeführten Tests sind in Anhang D und Anhang E dargestellt. Die auf *Writeprints* basierenden Modelle erzielen in fast allen der erhobenen Performanzmetriken die besten Ergebnisse. Die auf Basis von *8-Features* erstellten Modelle erreichen jedoch bei kleinen Segmentierungsgrößen gute *Recall*-Werte. Die Performanz des C4.5-Algorithmus steigt mit zunehmender Segmentierungsgröße weniger stark als die der anderen Algorithmen. Ab einer Segmentgröße von *S=10* ist *Naive Bayes* performanter als *C4.5*. *SVM* erzielt stets die besten Ergebnisse im Rahmen dieser Evaluation. Das Modell $M_{20,WP,SVM}$ klassifiziert bei identischer Ausprägung von C_{neg} und C_{pos} mehr als 90 % der Eingabesegmente korrekt und bestätigt somit, dass das in dieser Arbeit untersuchte Verfahren für eine Implementierung im Anwendungskontext geeignet ist.

Abbildung 17 verdeutlicht am Beispiel SVM-basierter Modelle, dass mit zunehmender Segmentierungsgröße bessere Klassifikationsergebnisse erzielt werden. Je größer das Eingabesegment, desto mehr Informationen können für eine Zuordnung genutzt werden. Die auf *Writeprints* basierenden Modelle profitieren besonders stark, wenn die Segmentierungsgröße von *S=1* auf *S=10* erhöht wird.

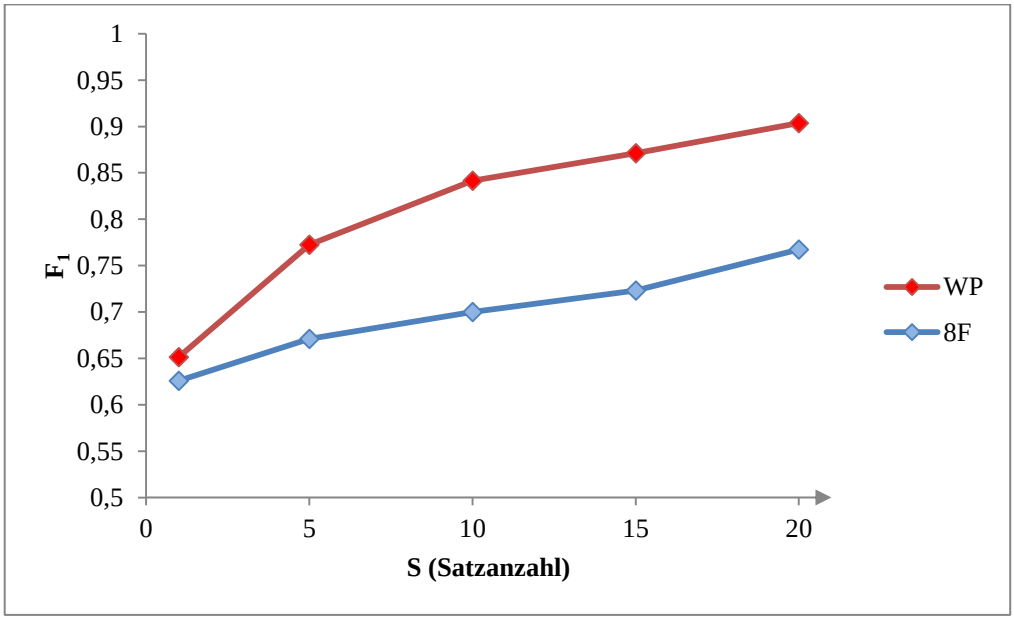


Abbildung 17: F_1-Wert der mit *SVM* erstellten Modelle im Kontext der technischen Evaluation

Das dargestellte Klassifikationsverhalten gilt, wenn alle Testsegmente zu 100 % aus Sätzen einer Klasse zusammengesetzt werden. Im Anwendungskontext sinkt jedoch die Performanz, wenn die Segmentierungsgröße zu hoch gewählt wird (siehe Kapitel 3.7.1.4), da die

gesuchten Strukturen gegebenenfalls durch umliegende Informationen verdeckt werden (vgl. [Mothe et al. 2015, 294]).

Auf Grundlage der in Tabelle 7 dargestellten Performanzmetriken wurden die auf *Writeprints* basierenden und durch *SVM* erstellten Modelle als geeignete Kandidaten für eine Evaluation im Anwendungskontext ausgewählt. Insgesamt konnten somit 25 der 30 erstellten Modelle ausgeschlossen werden. Die gewählten Modelle werden in den folgenden Kapiteln unverändert übernommen und auf weiteren Testkorpora evaluiert. Hierbei wird unter anderem eine optimale Segmentierungsgröße bestimmt.

3.7 Evaluation

Im Rahmen der Evaluationsphase wird die anwendungsbezogene Performanz der entwickelten Modelle überprüft (vgl. [Roiger 2017, 215]). Für die Erstellung und Beurteilung des in Kapitel 3.7.3 betrachteten Testszenarios wurden zunächst Einflussfaktoren im Anwendungskontext identifiziert. Die Evaluationsphase wird im Kontext dieser Arbeit als *Phase III der Hauptphasen der Verfahrensentwicklung* bezeichnet. Neben speziellen Testszenarien wird auf Grundlage des *PAN-PC-11* ein anwendungsnaher Testkorpus erstellt. Zudem erfolgt die Wahl einer optimalen Segmentierungsgröße (siehe Abbildung 18).

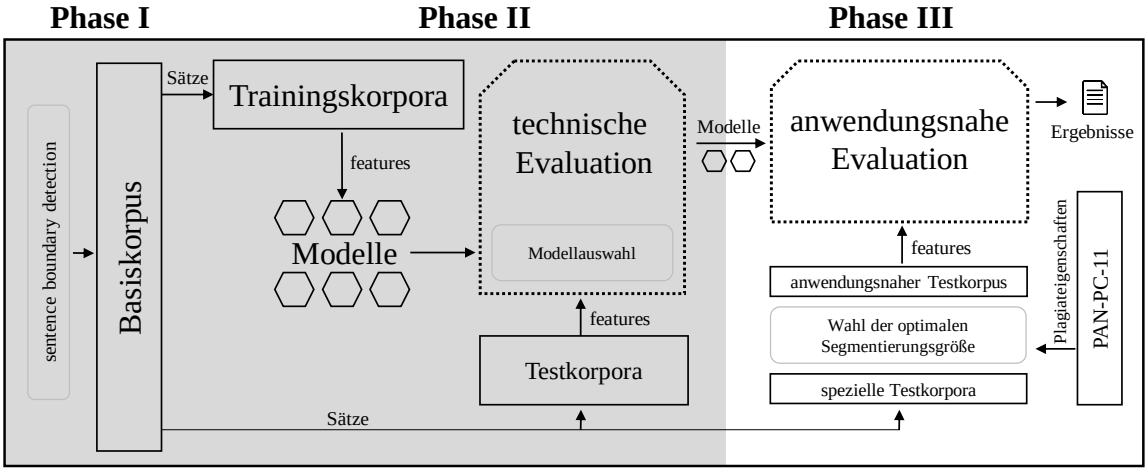


Abbildung 18: *Hauptphasen der Verfahrensentwicklung*, Phase III

3.7.1 Einflussfaktoren im Anwendungskontext

3.7.1.1 Performanz in Abhängigkeit des Dokumenttyps

Das folgende Kapitel beleuchtet, inwiefern sich die Performanz des entwickelten Verfahrens für verschiedene Dokumenttypen unterscheidet. Im Anwendungskontext müssen die in Kategorie C_{neg} enthaltenen Dokumenttypen von jenen der Kategorie C_{pos} unterschieden werden.

Eine Unterscheidung der Klassen innerhalb der genannten Kategorien ist nicht relevant. Zur Beurteilung der dokumenttypabhängigen Performanz wird die in Tabelle 6 dargestellte Konfusionsmatrix ausgewertet. Letztere wurde im Rahmen der technischen Evaluation durch das Modell $M_{20,WP,SVM}$ erzeugt. Das genannte Modell erzielte aufgrund der hohen Segmentierungsgröße die besten Ergebnisse der in Kapitel 3.6.9 ausgewählten Modelle.

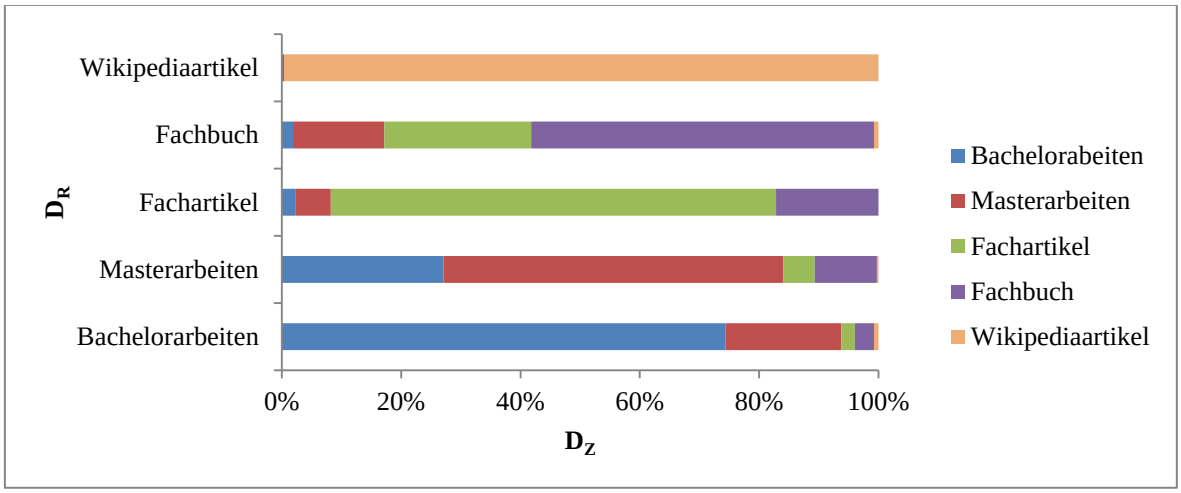


Abbildung 19: Anteile der zugeordneten Dokumenttypen je Dokumenttyp für das Modell $M_{20,WP,SVM}$

Abbildung 19 stellt die in Tabelle 6 erfassten Daten als Balkendiagramm dar. Insbesondere Bachelor- und Masterarbeiten können häufig nicht voneinander unterschieden werden. Auch eine Unterscheidung von Segmenten aus Fachbüchern und Fachartikeln führt häufig zu Fehleinordnungen. Eine Trennung in die Kategorien C_{neg} und C_{pos} funktioniert in Relation zu den zuvor genannten Unterscheidungen zuverlässig. Segmente aus Masterarbeiten werden jedoch zum Teil als Fachbuchsegmente klassifiziert und erhöhen somit die Anzahl der *false positives*.

DZ	**1 - E (Prozent)**
Bachelorarbeit	92,44
Masterarbeit	75,34
Fachbuch	74,16
Fachartikel	81,78
Wikipediaartikel	97,03

Tabelle 8: Performanz des Modells $M_{20,WP,SVM}$ nach Dokumenttyp

Tabelle 8 fasst die Performanz des Modells $M_{20,WP,SVM}$ anhand des Anteils korrekt zugeordneter Segmente je Dokumenttyp zusammen. Insbesondere Segmente des Typs *Wikipediaartikel* werden korrekt klassifiziert. Nach manueller Prüfung der genannten Segmente könnte dies auf die vergleichsweise geringen Satzlängen und die hohe Anzahl an Eigennamen und Fachbegriffen je Segment zurückzuführen sein.

Die Auswertungen im Kontext dieses Kapitels haben gezeigt, dass der Dokumenttyp starken Einfluss auf die Performanz des Klassifikationsverfahrens hat. Nach den Studien von Manar, Shameem (vgl. [Manar/Shameem 2014, 753]) und Turnitin (vgl. [o. V. 2011]) ist die Unterscheidung von Segmenten des Typs *Wikipediaartikel* und des Typs *Bachelorarbeit* besonders relevant im Kontext der Plagiatanalyse. Beide Dokumenttypen konnten im Rahmen der technischen Evaluation durch das in dieser Arbeit entwickelte Verfahren mit hoher Genauigkeit automatisiert unterschieden werden. Weitere Evaluationsschritte müssen zeigen, inwiefern diese Ergebnisse in einen anwendungsnäheren Kontext übertragbar sind.

3.7.1.2 Performanz in Abhängigkeit von Obfuskation

Obfuskationstechniken (siehe Abbildung 6) verändern die aus den Quelldokumenten übernommenen Textabschnitte und erschweren somit die Erkennung von Plagiaten. Die im Rahmen der Evaluation von Plagiatanalyseverfahren erzeugten künstlichen Plagiate werden obfuskiert, um in ihren Eigenschaften realen Plagiaten zu entsprechen. Um zu prüfen, inwiefern die Performanz des in dieser Arbeit entwickelten Verfahrens durch Obfuskation beeinflusst wird, wurde eine Methodik implementiert, welche umgangssprachlich als *rogeting* bezeichnet wird. Der Begriff *rogeting* bezieht sich auf den Thesaurus *Roget's Thesaurus* (vgl. [Mawson 1911, 1]) und beschreibt eine auffällig häufige und teilweise unpassende Nutzung von Synonymen. Durch *rogeting* sollen Plagiatanalyseverfahren überwunden werden, um gänzlich plagiierte Artikel in *predatory journals* (vgl. [Beall 2015]) zu veröffentlichen (vgl. [Gasparyan et al. 2017, 1223]).

Modell	**Obfuskation**	**TP**	**FN**	**E (Prozent)**	**2*SE (Prozentpunkte)**	**R**	**P**	**F1**
$M_{20,WP,SVM}$	Nein	736	68	9,76	1,48	0,915	0,892	**0,904**
$M_{20,WP,SVM}$	Ja	759	45	8,33	1,38	0,944	0,895	**0,919**

Tabelle 9: Performanz des Modells $M_{20,WP,SVM}$ in Abhängigkeit von Obfuskation

Für eine Simulation des beschriebenen *rogeting* wurde der im Rahmen der technischen Evaluation für das Modell $M_{20,WP,SVM}$ erstellte Testkorpus nachträglich verändert und anschließend erneut für eine Evaluation genutzt. Alle der Kategorie C_{pos} zugehörigen Segmente wurden unter Nutzung des *Java wrappers* „TT4J"[23] durch *TreeTagger* analysiert. Hierbei wurde jedem Wort ein *POS tag* zugeordnet. Die als Substantiv gekennzeichneten

[23] https://github.com/reckart/tt4j

Worte wurden anschließend durch das jeweils erste gefundene Synonym ersetzt. Hierfür wurde auf eine von *OpenThesaurus* bereitgestellte Synonymliste[24] zurückgegriffen.

Tabelle 9 zeigt, dass die verwendete Obfuskationstechnik keinen Einfluss auf die Performanz des in dieser Arbeit entwickelten Verfahrens hat. Die Ersetzung der Substantive verändert demnach nicht jene Eigenschaften, die durch das entwickelte Verfahren für eine Klassifikation der Segmente genutzt werden. Die automatisierte Erkennung der Segmente der Kategorie C_{pos} erfolgte zum Teil zuverlässiger als ohne Anwendung von Obfuskation. Da die Segmente der Kategorie C_{neg} nicht verändert wurden, blieb die Anzahl der *false positives* und *true negatives* unverändert. Weitere Untersuchungen müssen zeigen, inwiefern andere Obfuskationstechniken die Ergebnisse der Klassifikation beeinflussen.

3.7.1.3 Performanz in Abhängigkeit des Genres

Mikros und Argiri zeigen in ihrer Arbeit, dass einige der im Rahmen stilometrischer Analysen erfassten Texteigenschaften mit dem Thema des Textes korrelieren (vgl. [Mikros/Argiri 2007, 1f]). Da den im Kontext dieser Arbeit genutzten Dokumenten kein Thema zugeordnet ist und eine manuelle Themenzuordnung aufgrund der Menge an Dokumenten nicht möglich ist, wird stattdessen der Einfluss des Genres untersucht. Die Untersuchung und Veränderung von Substrukturen in den genutzten Testkorpora ermöglicht zudem eine bessere Abschätzung der Performanz des entwickelten Verfahrens unter sich ändernden Realbedingungen (vgl. [Lipka et al. (2012), 972]). Die für die Untersuchung gewählten Substrukturen beziehungsweise Subklassen wurden nicht statistisch ermittelt, sind aber ontologisch begründet (vgl. [Raedt et al. 2001, 2], siehe Kapitel 3.6.1).

Die im Kontext dieser Arbeit für den Trainingskorpus genutzten Segmente wurden aus Sätzen verschiedener Dokumente aus verschiedenen Genres zusammengesetzt. Die durch den Klassifikationsalgorithmus erfassten Eigenschaften wurden somit über verschiedene Genres und Dokumente gemittelt, um eine Überanpassung des erzeugten Modells zu verhindern. Das folgende Kapitel evaluiert anhand von Subklassen, inwiefern dennoch ein Zusammenhang zwischen der Genrezuordnung der Testsegmente und der Klassifikationsleistung des entwickelten Verfahrens besteht. Die erzeugten Testkorpora bestehen aus Segmenten jeweils eines Genres. Alle anderen Eigenschaften folgen den in Tabelle 3 dargestellten Eigenschaften. Der Umfang der Testkorpora ist durch die Anzahl der im Basiskorpus für das jeweilige

[24] https://www.openthesaurus.de/about/download

Genre vorhandenen Sätze begrenzt. Durch die notwendige Verkleinerung einiger Trainingskorpora (siehe Kapitel 3.6.3) konnten zusätzliche Testdokumente genutzt werden. Die bei einer Segmentierungsgröße von *S=20* maximal analysierbare Menge an Segmenten liegt durch die Nutzung von *Writeprints* weiterhin bei 599 Segmenten je Klasse. Da eine optimale Segmentierungsgröße in weiteren Evaluationsschritten ermittelt werden muss, wurde das Modell $M_{20,WP,SVM}$ zur Überprüfung der genreabhängigen Performanz genutzt.

Genre	**N_{Test}**	**TP**	**FP**	**TN**	**FN**	**E (Prozent)**	**2*SE (Prozentpunkte)**	**R**	**P**	**F_1**
Literaturwissenschaften	2395	1117	111	1087	80	7,98	1,11	0,933	0,910	**0,921**
Naturwissenschaften	811	375	33	373	30	7,77	1,88	0,926	0,919	**0,923**
Philosophie	475	214	64	174	23	18,32	3,55	0,903	0,770	**0,831**
Sozialwissenschaften	794	371	121	277	25	18,39	2,75	0,937	0,754	**0,836**
Technikwissenschaften	1020	463	43	467	47	8,82	1,78	0,908	0,915	**0,911**

Tabelle 10: Genreabhängige Performanz des Modells $M_{20,WP,SVM}$

Tabelle 10 zeigt, dass die Performanz für Testkorpora der Genres Literaturwissenschaften, Naturwissenschaften und Technikwissenschaften in etwa der in Kapitel 3.6.9 erfassten Performanz entspricht. In den Genres Philosophie und Sozialwissenschaften ist der erreichte *Precision*-Wert des Verfahrens durch die vergleichsweise hohe Anzahl an *false positives* niedriger als in anderen Genres.

D_R | D_Z	**Bachelorarbeit**	**Masterarbeit**	**Fachbuch**	**Fachartikel**	**Wikipediaartikel**
Bachelorarbeit	79	66	39	15	0
Masterarbeit	29	103	54	13	0
Fachbuch	2	11	79	39	1
Fachartikel	3	7	12	110	0
Wikipediaartikel	1	1	10	6	114

Tabelle 11: Konfusionsmatrix des Modells $M_{20,WP,SVM}$ im Genre Sozialwissenschaften.

Tabelle 11 zeigt die aus den Tests resultierende Konfusionsmatrix für das Genre Sozialwissenschaften. Der Anteil der als Fachbuchsegment klassifizierten Segmente aus Bachelor- und Masterarbeiten ist im Vergleich zu anderen Genres um ein Vielfaches höher. Selbiges gilt für das Genre Philosophie. Die Ursachen des beschriebenen Klassifikationsverhaltens

müssen in weiteren Arbeiten untersucht werden. Es ist möglich, dass Studierende der genannten Genres in ihren Abschlussarbeiten sehr umfangreich aus Fachbüchern zitieren. Da das im Rahmen dieser Arbeit entwickelte Verfahren alle dokumenttypfremden Abschnitte erkennt, werden auch korrekt zitierte Abschnitte als potentielles Plagiat klassifiziert. Die zuverlässige Identifikation von *false positives*, die durch korrekt zitierte Abschnitte entstehen, ist nur durch einen Abgleich mit Quelldokumenten möglich und kann somit nur durch *document-matching*-basierte Verfahren oder manuelle Prüfung realisiert werden (vgl. [Gipp 2014, 19]).

3.7.1.4 Performanz in Abhängigkeit der Segmentzusammensetzung

Kapitel 3.6.2 beschreibt, inwiefern die Wahl der Segmentierungsgröße die Performanz des entwickelten Verfahrens beeinflusst. Abbildung 17 zeigt, dass kleine Segmentierungsgrößen zu schlechteren Klassifikationsergebnissen führen. Das folgende Kapitel evaluiert hingegen den Effekt zu groß gewählter Segmente. Unterschreitet die Länge des Plagiats die Segmentierungsgröße, so erschwert dies seine Erkennung, da relevante Strukturen durch umliegende Informationen überdeckt werden (vgl. [Mothe et al. 2015, 294]). Zur Simulation des beschriebenen Effekts werden künstliche Plagiate erzeugt, die nur zu Teilen aus Sätzen der Kategorie C_{pos} gebildet wurden. Alle anderen Eigenschaften des Testkorpus folgen den in Tabelle 3 dargestellten Attributen.

N_{Test}	Plagiatanteil (Prozent)	TP	FN	E (Prozent)	2*SE (Prozentpunkte)	R
599	5	128	471	78,63	3,35	**0,214**
599	25	191	408	68,11	3,81	**0,319**
599	50	326	273	45,58	4,07	**0,544**
599	75	429	170	28,38	3,68	**0,716**
599	100	515	84	14,02	2,84	**0,860**

Tabelle 12: Performanz des Modells $M_{20,WP,SVM}$ in Abhängigkeit der Segmentzusammensetzung.

Der in Tabelle 12 dargestellte Plagiatanteil bezieht sich auf die Anzahl der Sätze je Segment, die dem Dokumenttyp *Fachbuch* zugeordnet sind. Die verbleibenden Sätze im jeweiligen Segment entstammen Dokumenten des Typs *Masterarbeit*. Als Performanzmetrik wurde ausschließlich der *Recall*-Wert für die Klasse *Fachbuch* angegeben, da nur die Segmente letzterer verändert wurden. Die Performanz für die verbleibenden Klassen sowie die erzielten *Precision*-Werte blieben somit unverändert.

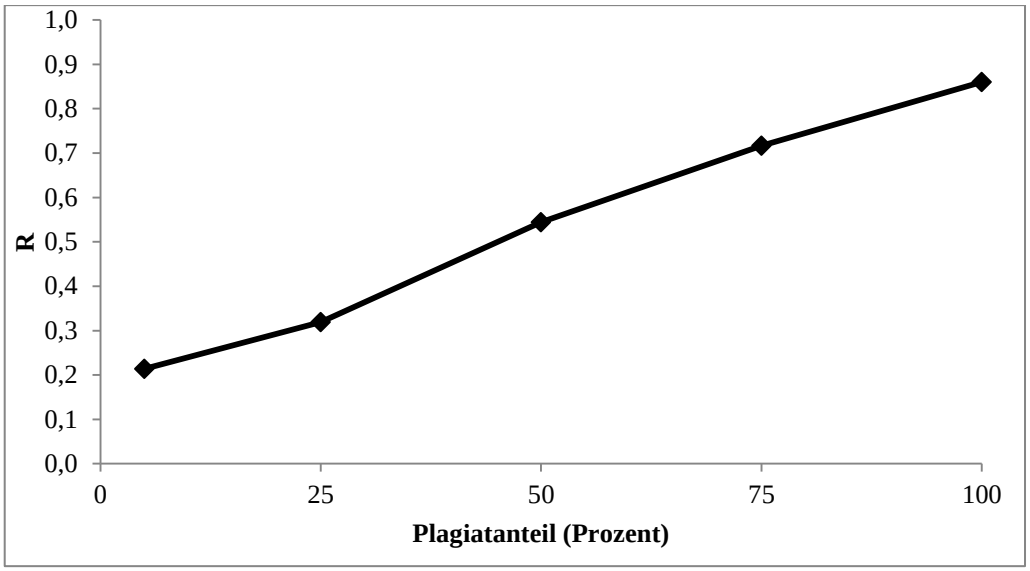


Abbildung 20: *Recall* des Modells $M_{20,WP,SVM}$ in Abhängigkeit der Segmentzusammensetzung

Tabelle 12 und Abbildung 20 verdeutlichen, dass die Segmentzusammensetzung starken Einfluss auf die Performanz des Klassifikationsverfahrens hat. Besteht ein Segment zu jeweils 50 % aus Sätzen verschiedener Dokumenttypen, so ist die Zuordnung eines konkreten Dokumenttyps nicht möglich. Eine Fehlerquote von mehr als 45 % spiegelt die beschriebene Problematik wider. Die durch die Mischung der Sätze erzeugten Segmente haben Texteigenschaften, die sehr nahe an der durch *SVM* definierten Trennebene liegen (siehe Abbildung 15). Überwiegt die Anzahl der Sätze eines Dokumenttyps im Segment, so verbessert sich die Klassifikationsleistung in Richtung der entsprechenden Klasse. Im Kontext der Plagiaterkennung bedeutet dies jedoch, dass Segmente, die zu weniger als 50 % aus Sätzen der Kategorie C_{pos} zusammengesetzt sind, mit hoher Wahrscheinlichkeit nicht als Plagiate erkannt werden. Der im Anwendungskontext interpretierte *Recall*-Wert für die genannten Segmente ist dementsprechend niedrig.

Die beschriebenen Ergebnisse zeigen, dass die Segmentierungsgröße des entwickelten Verfahrens die Länge der im Eingabedokument enthaltenen Plagiate möglichst selten überschreiten sollte. Werden neben Sätzen der Kategorie C_{pos} auch Sätze der Kategorie C_{neg} im Segment erfasst, so kann dies die Klassifikationsleistung des Verfahrens stark vermindern. Da zur durchschnittlichen Länge und Verteilung von Plagiaten keine Informationen vorliegen (vgl. [Potthast et al. 2010, 6]), wird als Grundlage für die Wahl der Segmentierungsgröße der *PAN-PC-11* analysiert.

3.7.2 Analyse des PAN Plagiarism Corpus 2011

Die in Kapitel 3.7.1 durchgeführten Analysen ermöglichen eine bessere Beurteilung der in Kapitel 3.7.3 erfolgenden Abschlussevaluation. Zudem wurde die Segmentzusammensetzung als wichtiger Einflussfaktor identifiziert. Das folgende Kapitel analysiert den *PAN-PC-11* hinsichtlich seiner Zusammensetzung und ermittelt eine optimale Segmentierungsgröße für das im Kontext dieser Arbeit entwickelte Verfahren.

Der *PAN-PC-11* ist eine angepasste Variante des *PAN-PC-10* und enthält 61.064 künstliche oder simulierte Plagiatfälle in 26.939 Dokumenten (vgl. [Potthast (2011), 2]). Der für die Evaluation intrinsischer Verfahren entwickelte Teil des Korpus besteht aus 4.753 Dokumenten und umfasst 11.443 Plagiate. Letzterer wurde für eine Analyse gewählt, um die Performanz des in dieser Arbeit entwickelten Verfahrens mit jener von intrinsischen Ansätzen zu vergleichen.

Segmentgrößen des entwickelten Verfahrens (Sätze)	**Durchschnittliche Segmentgrößen des entwickelten Verfahrens (Zeichen)**	**Durchschnittliche Segmentgrößen des entwickelten Verfahrens (Wörter)**	**Anteil der Plagiate im PAN-PC-11, die größer oder gleich der angegebenen durchschnittlichen Segmentgrößen in Zeichen sind (Prozent)**
1	144	19	99,99
5	663	87	61,77
10	1327	174	51,69
15	1993	261	41,83
20	2658	348	33,56

Tabelle 13: Durchschnittliche Segmentgrößen des entwickelten Verfahrens in Bezug zur Zusammensetzung des *PAN-PC-11* für intrinsische Plagiatanalyse

Die durchschnittliche Länge der Plagiate im genannten Korpus beträgt 3.968 Zeichen. Die in Tabelle 13 dargestellten Perzentilen zeigen jedoch, dass circa 48 % der Plagiate eine Länge von 1.327 Zeichen unterschreiten. Eine Segmentierungsgröße von *S=10* Sätzen entspricht in etwa dem Median über alle im intrinsischen *PAN-PC-11* auftretenden Plagiatlängen und wurde daher für eine Evaluation im Anwendungskontext ausgewählt. Bei einer Segmentierungsgröße von *S=10* stehen ausreichend Informationen für eine zuverlässige Klassifikation zur Verfügung (siehe Abbildung 17). Gleichzeitig wird der in Abbildung 20 dargestellte Effekt für einen Großteil der zu klassifizierenden Segmente vermieden. Bestehende Plagiatanalyseverfahren, die speziell für die Auswertung des intrinsischen *PAN-PC-11* angepasst wurden, nutzen Fenstergrößen zwischen 200 und 1.000 Wörtern (vgl. [Potthast (2011), 3]). Die im Rahmen dieser Arbeit gewählte Segmentierungsgröße von *S=10* ent-

spricht durchschnittlich 174 Wörtern und ist somit geringer als bei vergleichbaren intrinsischen Ansätzen. Insgesamt wurden 45.408.121 der 748.872.218 Zeichen des intrinsischen *PAN-PC-11* als Plagiate markiert. Dies entspricht in etwa 6 %. In einer 30-seitigen Bachelorarbeit wären nach den ermittelten Werten circa 1,8 Seiten plagiiert.

Klassenverteilung	Masterarbeiten: 47,00 %	Bachelorarbeiten: 47,00 %	Wikipediaartikel: 2,00 %
	Fachartikel: 2,00 %	Fachbücher: 2,00 %	
Genreverteilung	Technikwissenschaften: 20,00 %	Sozialwissenschaften: 20,00 %	
	Naturwissenschaften: 20,00 %	Literaturwissenschaften: 20,00 %	Philosophie: 20,00 %
Plagiatlängen (Sätze)	10 Sätze: 50,00 %	3 Sätze: 40,00%	7 Sätze: 10,00%
Weitere Eigenschaften	Segmentgröße: 10 Sätze	Satzzusammensetzung: nach Quelldokument	Obfuskation: ja

Tabelle 14: Eigenschaften des anwendungsnahen Testkorpus

Tabelle 14 zeigt den Testkorpus, der auf Grundlage der Eigenschaften des *PAN-PC-11* erstellt wurde. Der Anteil der Segmente aus der Kategorie C_{pos} beträgt 6 % und ist gleichmäßig über die enthaltenen Klassen verteilt. Für jedes Genre wurde eine identische Anzahl an Segmenten im Testkorpus hinterlegt. Für jede der in Kategorie C_{pos} enthaltenen Klassen wurde die dargestellte Plagiatlängenverteilung umgesetzt. Letztere orientiert sich an den in Tabelle 13 dargestellten Perzentilen. 50 % der Segmente aus den genannten Klassen enthalten zehn Sätze des ursprünglichen Dokumenttyps. 40 % enthalten drei und 10 % enthalten sieben Sätze des ursprünglichen Dokumenttyps. Um sicherzustellen, dass jedes Segment dennoch insgesamt zehn Sätze enthält, wurden die zuvor verkürzten Segmente mit Sätzen des Dokumenttyps *Masterarbeit* aufgefüllt. Wie schon in Kapitel 3.7.1.4 beschrieben, werden auf diese Weise Plagiate simuliert, welche die gewählte Segmentierungsgröße unterschreiten. Die Zusammensetzung der einzelnen Sätze des Basiskorpus zu Segmenten erfolgt für alle Klassen entsprechend des Anwendungskontextes nicht mehr zufällig, sondern entsprechend ihrer Quelldokumente. Die Segmente der Kategorie C_{pos} wurden zudem nach der in Kapitel 3.7.1.2 beschriebenen Methodik obfuskiert. Der beschriebene Korpus wird im folgenden Kapitel für eine anwendungsnahe Evaluation genutzt.

3.7.3 Performanz im Anwendungskontext

Im folgenden Kapitel wird das entwickelte Verfahren unter möglichst anwendungsnahen Bedingungen evaluiert. Die erzielten Ergebnisse werden anschließend der Performanz vergleichbarer intrinsischer Verfahren gegenübergestellt. Der genutzte Testkorpus ist in Kapitel 3.7.2 beschrieben. Die Anzahl der Testdokumente für die Kategorie C_{neg} orientiert sich an der maximal durch *Writeprints* analysierbaren Menge an Segmenten je Klasse für die Segmentierungsgröße $S=10$.

N_{Test} [C_{pos}]	N_{Test} [C_{neg}]	TP	FP	TN	FN	E (Prozent)	2*SE (Prozentpunkte)	R	P	F_1	F_2
138	2278	74	660	1618	64	29,97	1,86	0,536	0,101	**0,170**	**0,288**

Tabelle 15: Performanz des Modells $M_{10,WP,SVM}$ im anwendungsnahen Test

Tabelle 15 zeigt die durch das Verfahren erreichten Klassifikationsergebnisse. Ein *Recall*-Wert von 0,536 bedeutet, dass circa 53 % der im Testkorpus vorhandenen Plagiate erkannt wurden. Der erzielte *Precision*-Wert ist hingegen niedrig. Nur circa 10 % der als Plagiat erkannten Segmente entsprechen korrekten Zuordnungen. Dies ist auf die ungleiche Verteilung von C_{pos} und C_{neg} zurückzuführen. Fast 80 % der Segmente aus Bachelorarbeiten und circa 62 % der Segmente aus Masterarbeiten wurden korrekt zugeordnet (siehe Tabelle 16). Da beide Dokumenttypen insgesamt 94 % des Testdatensatzes ausmachen, wird dennoch eine hohe Anzahl an *false positives* produziert. Letztere verdecken die korrekt erkannten, aber selten auftretenden *true positives* und senken somit den erreichten *Precision*-Wert.

D_Z	1 - E (Prozent)
Bachelorarbeit	79,72
Masterarbeit	62,34
Fachbuch	56,52
Fachartikel	50,00
Wikipediaartikel	54,35

Tabelle 16: Anteil der korrekt erkannten Segmente nach Dokumenttyp im anwendungsnahen Test

Tabelle 16 zeigt, dass der Anteil korrekt erkannter Segmente sowohl für Bachelor-, als auch für Masterarbeiten im Vergleich zu Tabelle 8 gesunken ist. Dies ist neben der Verkleinerung der Segmentierungsgröße vor allem auf die veränderte Satzzusammenstellung zurückzuführen. Die in den Segmenten enthaltenen Sätze sind nicht mehr zufällig aus verschiedenen Dokumenten zusammengestellt, sondern wurden in korrekter Reihenfolge einem einzelnen Dokument entnommen. Je Segment wurde hierfür ein zufälliges Dokument passender Genre- und Klassenzuordnung ausgewählt.

D_R | D_Z	Bachelorarbeit	Masterarbeit	Fachbuch	Fachartikel	Wikipediaartikel
Bachelorarbeit	665	243	111	79	41
Masterarbeit	289	421	215	125	89
Fachbuch	1	19	10	11	5
Fachartikel	1	22	8	14	1
Wikipediaartikel	1	20	8	2	15

Tabelle 17: Konfusionsmatrix des Modells $M_{10,WP,SVM}$ im anwendungsnahen Test

Die so erzeugten Segmente sind spezieller hinsichtlich ihrer stilometrischen Eigenschaften und unterscheiden sich stärker von den im Trainingskorpus genutzten, über alle Dokumente gemittelten Segmenten.

Die im Vergleich zu Tabelle 8 gesunkene Performanz für Klassen der Kategorie C_{pos} ist insbesondere auf die Verwendung gemischter Segmente zurückzuführen. Die Hälfte aller Segmente der genannten Kategorie enthält Sätze der Klasse *Masterarbeit*. Die in Tabelle 17 dargestellte Konfusionsmatrix zeigt, dass der überwiegende Teil der *false negatives* auf eine Einordnung als Masterarbeit zurückzuführen ist.

Entwickler des Verfahrens	**R**	**P**	**F_1**	**F_2**
Oberreuter et al. [Oberreuter et al. (2011)]	0,312	0,340	0,325	0,317
Luyckx et al. [Luyckx et al. 2011]	0,108	0,428	**0,172**	**0,127**
Gupta et al. [Gupta et al. 2011]	0,078	0,108	**0,091**	**0,083**
Akiva [Akiva 2011]	0,066	0,128	**0,087**	**0,073**

Tabelle 18: Performanz der besten vier Einreichungen im *PAN-PC-11* Wettbewerb für intrinsische Plagiatanalyse (in Anlehnung an [Potthast (2011), 4])

Tabelle 18 zeigt die vier performantesten Einreichungen des *PAN-PC-11* Wettbewerbs für intrinsische Plagiatanalyse (vgl. [Potthast (2011), 4]). Ein Teilziel der vorliegenden Arbeit war es, durch eingeschränkte Nutzung externer Daten bessere Ergebnisse als intrinsische Methoden zu erzielen. Da die Evaluation des entwickelten Verfahrens nicht direkt auf dem *PAN-PC-11* erfolgen kann (siehe Kapitel 3.2.1), wurde ein in seinen Eigenschaften möglichst ähnlicher Testkorpus erzeugt. Die durch das entwickelte Verfahren erzielten Ergebnisse sind dennoch nur eingeschränkt mit den in Tabelle 18 dargestellten Ergebnissen vergleichbar.

Die durch Oberreuter et al. entwickelte Methodik (vgl. [Oberreuter et al. (2011)]) wurde aus Gründen der Vollständigkeit in die Gegenüberstellung aufgenommen, sollte aber nicht für einen Vergleich genutzt werden. Oberreuter et al. nutzen Eigenschaften des *PAN-PC-11*, die im realen Anwendungskontext von Plagiatanalyseverfahren nicht vorhanden sind (vgl. [Potthast (2011), 5]). Das in dieser Arbeit entwickelte Verfahren erzielt im Vergleich zu Luyckx et al. einen ähnlichen F_1-Wert (vgl. [Luyckx et al. 2011]). Die erzielten *Precision*- und *Recall*-Werte sind hierbei jedoch sehr unterschiedlich. Die durch Luyckx et al. entwickelte Methodik erkennt nur circa 10 % der vorhandenen Plagiate, produziert aber weniger *false positives*. Der erreichte *Precision*-Wert ist somit deutlich höher. Wie auch andere intrinsische Verfahren betrachtet die durch Luyckx et al. entwickelte Methodik Plagiate als stilistische Ausreißer in Bezug auf den Rest des Dokumentes (vgl. [Luyckx et al. 2011, 5]). Da die

für die Klassifikation genutzten Informationen ausschließlich aus der Mehrheitsklasse abgeleitet werden, erfolgt zu häufig eine Einordnung in letztere.

Die beschriebenen Ergebnisse zeigen, dass das in dieser Arbeit entwickelte Verfahren prinzipiell für eine Identifikation von Plagiaten in studentischen Abschlussarbeiten geeignet ist. Im simulierten Anwendungskontext konnten 71,03 % der Segmente aus der Kategorie C_{neg} und 53,62 % der Segmente aus der Kategorie C_{pos} korrekt eingeordnet werden. Für eine reale Anwendung sollten die erreichten *Precision*-Werte verbessert werden. Die Performanz des Verfahrens steigt, wenn weniger Plagiate in ihrer Länge die gewählte Segmentierungsgröße unterschreiten oder der Anteil plagiierter Abschnitte in Bezug auf den Rest des Eingabedokuments zunimmt.

Die erzielten Ergebnisse sind hinsichtlich des F_1-Wertes mit intrinsischen Verfahren vergleichbar. Die erreichten *Recall*-Werte sind hierbei deutlich höher. Im Rahmen des angegebenen F_2-Wertes werden *Recall*-Werte höher gewichtet als *Precision*-Werte. Die durch das untersuchte Verfahren und intrinsische Methoden markierten Plagiate müssen stets händisch geprüft (vgl. [Gipp 2014, 19]) oder über extrinsische Methoden einem konkreten Dokument zugeordnet werden. Die Markierung der Plagiate kann somit als Vorauswahl interpretiert werden. Eine höhere Gewichtung des *Recall*-Wertes bedeutet, dass eine Verkleinerung der Vorauswahlmenge weniger relevant ist als die Erfassung möglichst vieler Plagiate in letzterer. Der durch Luyckx et al. erzielte *Precision*-Wert von 0,108 verkleinert die Vorauswahlmenge sehr stark, bedeutet aber auch, dass fast 90 % aller Plagiate nicht erfasst werden. Kapitel 3.9 erläutert unter anderem, inwiefern die hohen *Recall*-Werte des entwickelten Verfahrens genutzt werden können, um im Rahmen eines hybriden Ansatzes die Ergebnisse intrinsischer Verfahren zu verbessern.

Das folgende Kapitel erläutert ein Konzept zur Implementierung eines Prototyps auf Basis des untersuchten Verfahrens.

3.8 Bereitstellung

Im Kontext des *CRISP-DM* dient die Bereitstellungsphase der Implementierung des erstellten Modells in die Produktivumgebung (vgl. [Reese et al. 2017, 362]). Im Rahmen der vorliegenden Arbeit wird hierfür der in Abbildung 21 gezeigte Java-basierte Prototyp beschrieben, der in Anlehnung an das entwickelte Modell Plagiate in studentischen Ab-

schlussarbeiten markiert. Die Eingabe letzterer erfolgt im PDF-Format über eine Benutzerschnittstelle. Die Eingabedateien werden durch *Science Parse*[25] eingelesen und weiterverarbeitet.

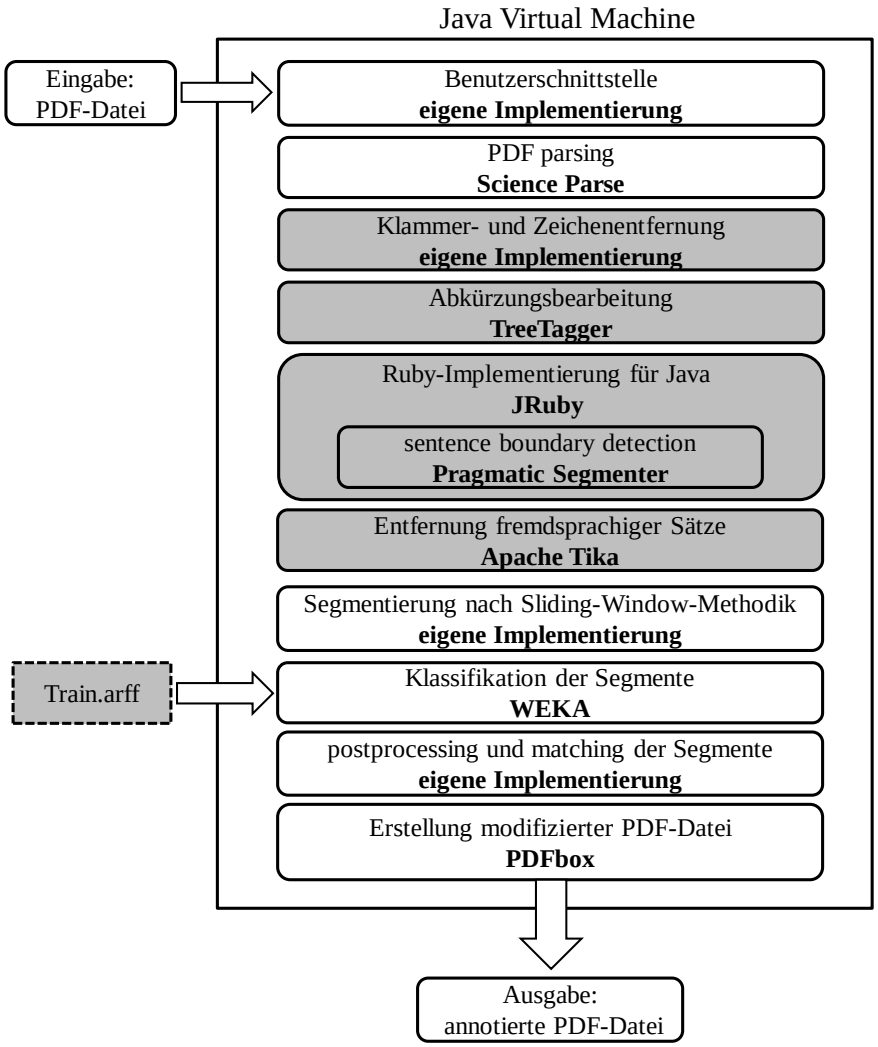


Abbildung 21: Konzept für die Implementierung eines Prototyps auf Basis des untersuchten Verfahrens.

Die grau markierten Abschnitte wurden als Quellcode zur Verfügung gestellt[26] und können für eine Implementierung genutzt werden. Die durch *Science Parse* erzeugte Ausgabe kann entweder über *GSON*[27] und *Jsonschema2Pojo*[28] in Java-Objekte konvertiert werden, oder durch Modifikation von *Science Parse* direkt im Java-Kontext erfolgen. Die Weiterverarbeitung der Ausgaben erfolgt durch das in Kapitel 3.5.2 beschriebene *Preprocessing*-Verfahren. Im Text enthaltene Abkürzungen werden über die von *TreeTagger*[29] zur Verfügung gestellte

[25] https://github.com/allenai/science-parse, Apache Licence 2.0
[26] https://goo.gl/kZjMkV
[27] https://github.com/google/gson, Apache Licence 2.0
[28] https://github.com/joelittlejohn/jsonschema2pojo, Apache Licence 2.0
[29] http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger, frei für Forschungszwecke, Erlaubnis zur Verteilung der Abkürzungsliste im Rahmen dieser Arbeit durch den Autor gewährt

Abkürzungsliste in eine SBD-geeignete Darstellung überführt. Die Satzerkennung erfolgt durch das Ruby-Programm *Pragmatic Segmenter*[30], welches über *JRuby*[31] in den Java-Kontext integriert wird. Die Filterung der extrahierten Sätze nach Sprache erfolgt durch *Apache Tika*[32]. Einzelne Sätze werden anschließend zu Segmenten zusammengesetzt, die eine satzweise Verschiebung eines *Sliding-Window*-Bereiches repräsentieren.

Die Klassifikation der genannten Segmente erfolgt auf Grundlage des *WEKA frameworks*[33]. Letzteres liest die in Form von *Attribute-Relation-File-Format*-(ARFF)-Dateien zur Verfügung gestellten Feature-Listen (vgl. [Witten et al. 2017a, 59]) ein und bildet auf Grundlage des gewählten Algorithmus ein Klassifikationsmodell. Für jede der in Kapitel 3.6.9 aufgeführten Segmentierungs- und *Feature-Set*-Varianten wurde eine ARFF-Datei bereitgestellt. Die verwendeten Trainingskorpora können aus rechtlichen Gründen nicht in Textform verbreitet werden (vgl. [o. V. 2018a]). Die klassifizierten Segmente müssen anschließend im Rahmen eines *Postprocessing*-Schrittes anhand der erfassten Zuordnungswahrscheinlichkeiten bewertet werden, um plagiierte Abschnitte möglichst genau einzugrenzen. Letztere müssen im folgenden Schritt, beispielsweise über *String-Matching*-Verfahren, ihrer Position im Eingabe-PDF zugeordnet werden. Auf Grundlage der erfassten Positionsdaten kann *PDFbox*[34] abschließend eine PDF-Datei erstellen, in welcher potentiell plagiierte Abschnitte hervorgehoben sind. In Anlehnung an das durch Niezgoda und Way entwickelte Tool *SNITCH* (vgl. [Niezgoda/Way 2006, 51]) kann zudem ein Suchmaschinenlink für markierte Segmente hinterlegt werden, um eine händische Prüfung auf Grundlage einer Internetrecherche zu erleichtern.

3.9 Kritik und Ausblick

Das folgende Kapitel betrachtet Schwächen des untersuchten Verfahrens und der angewendeten Evaluationsmethodik. Die kritische Betrachtung bezieht sich hierbei auf die verwendeten Eingabedaten, das erstellte Klassenschema, die Nutzung von *feature sets* und Klassifikationsalgorithmen sowie die verwendeten Testkorpora. Zudem werden potentielle Ansätze für weiterführende Forschungsvorhaben erläutert.

[30] https://github.com/diasks2/pragmatic_segmenter, Massachusetts Institute of Technology License
[31] https://github.com/jruby/jruby, Eclipse Public License 2.0 und General Public License 2.0
[32] https://github.com/apache/tika, Apache Licence 2.0
[33] https://www.cs.waikato.ac.nz/~ml/weka/downloading.html, General Public License 3.0
[34] https://github.com/apache/pdfbox, Apache Licence 2.0

Die für die Erstellung des Basiskorpus genutzte Stichprobe an Dokumenten ist in Bezug auf die Gesamtanzahl aller möglichen Dokumente sehr klein. Für eine Vergrößerung des Basiskorpus könnten weitere, bereits vorhandene Korpora spezifischer Dokumenttypen genutzt werden. Zudem sollten zusätzliche Dokumenttypen, wie zum Beispiel Blog-Einträge oder Artikel von Nachrichtenportalen integriert werden, um im realen Kontext möglichst viele potentiell plagiierte Segmente erkennen zu können. Der erstellte Basiskorpus ist zudem auf deutschsprachige Dokumente begrenzt. Bei einer Implementierung des Verfahrens für andere Sprachen müssen Dokumente für letztere im Korpus hinterlegt werden.

Da PDF-Dateien layoutbasiert sind (vgl. [H. Bast/C. Korzen 2017, 1]), ist das zum Einlesen genutzte *Parsing*-Verfahren komplex und fehlerbehaftet. Die Software *Grobid*[35] ähnelt dem hierfür verwendeten Programm *Science Parse* und wurde anhand verschiedener Testszenarien evaluiert (vgl. [o. V. 2018b]). Für *Science Parse* liegen hingegen keine Evaluationsergebnisse vor. Auch die Performanz des genutzten SBD-Verfahrens ist nicht dokumentiert. Weitere Evaluationsschritte sollten daher die Korrektheit der durch das *PDF Parsing* und das SDB-Verfahren produzierten Ergebnisse überprüfen. Nach Ceska und Fox ist der Einfluss der *Preprocessing*-Phase auf die Performanz von Plagiatanalyseverfahren gering (vgl. [Ceska/Fox 2009, 55]). Der Einfluss des vorangestellten *Parsing*-Schrittes wird hierbei jedoch nicht berücksichtigt.

Bei der Erstellung des Klassenschemas wurde die Annahme getroffen, dass sich Masterarbeiten und Bachelorarbeiten für eine automatisierte Unterscheidung stilistisch zu ähnlich sind. Abbildung 19 bestätigt diese Ähnlichkeit, zeigt jedoch auch, dass eine Unterscheidung unter Umständen möglich ist. Nach Manar und Shameem haben 51,53 % der Bachelorstudierenden mindestens einmal die Arbeiten von Menschen aus ihrem Umfeld plagiiert (vgl. [Manar/Shameem 2014, 753]). Die Erkennung von Masterarbeiten in Bachelorarbeiten und umgekehrt könnte somit zur Verbesserung der Performanz im realen Kontext beitragen. Die beschriebenen Fälle sind durch die in dieser Arbeit genutzte Evaluationsmethodik nicht abgebildet, da beide Dokumenttypen in einer Kategorie zusammengefasst werden.

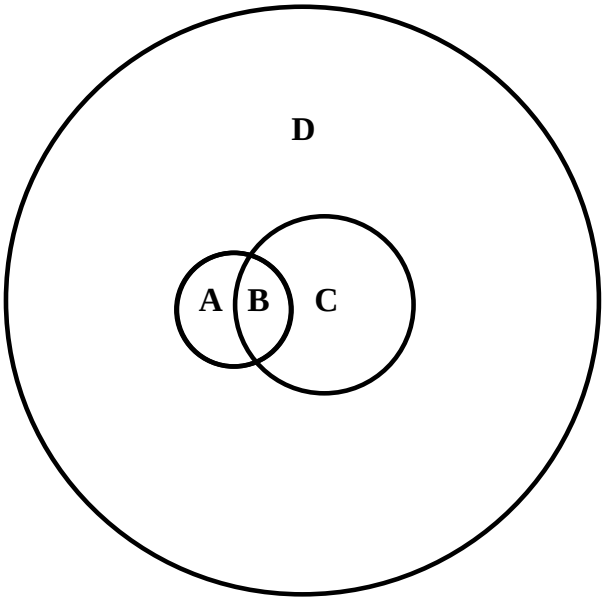
Die verwendeten Klassifikationsalgorithmen wurden mit den in *JStylo* hinterlegten Konfigurationsparametern ausgeführt. Eine Anpassung letzterer kann die Performanz des erstellten Modells unter Umständen erhöhen (vgl. [Frohlich/Zell (2005), 1431]). Dies gilt ebenso für die genutzten *feature sets*. Durch eine anwendungsbezogene Anpassung der durch

[35] https://github.com/kermitt2/grobid

Writeprints erfassten *features* werden nach Almishari et al. bessere Ergebnisse erzielt (vgl. [Almishari et al. 2014, 5]). Anhang F zeigt die bei einer Segmentierungsgröße von *S=10* durch *Writeprints* extrahierten *features*. Vereinzelte *features*, wie beispielsweise das Worttrigramm „war ein deutscher“ wirken sehr inhaltsspezifisch und müssen im Rahmen weiterführender Evaluationen geprüft werden. Um eine Überanpassung der Modelle auszuschließen, können gegebenenfalls nur lexikalische und syntaktische *features* genutzt werden. Von den 646 in Anhang F gezeigten *features* beziehen sich drei auf das Zeichen NULL. Letzteres entsteht, wenn der durch die ARFF-Dateien genutzte ASCII-Zeichensatz (vgl. [Paynter 2008]) die durch *Writeprints* erfassten Zeichen nicht darstellen kann. Dies ist relevant für eine Weiterverarbeitung der bereitgestellten ARFF-Dateien, hatte jedoch auf die Performanz des Verfahrens im Kontext dieser Arbeit keinen Einfluss.

Ein Grundproblem bei der Evaluation von Plagiatanalyseverfahren ist das Fehlen von realen Plagiaten, da letztere aufgrund ethischer und rechtlicher Bedenken nicht genutzt werden können (vgl. [Kraus 2016, 2]). Im Rahmen dieser Arbeit wurden daher einzelne Segmente aus verschiedenen Dokumenttypen als Plagiate interpretiert. Für eine anwendungsnähere Evaluation des untersuchten Verfahrens sollte stattdessen ein Testkorpus aus zusammenhängenden Texten erstellt werden. Die Eigenschaften des *PAN-PC-11* können hierbei als Basis dienen (vgl. [Potthast (2011), 2]). Die angewendeten Obfuskationstechniken sollten sich an Abbildung 6 orientieren. Für das Einlesen des beschriebenen Testkorpus muss ein *Sliding-Window*-Verfahren (siehe 3.6.2) implementiert werden.

Wie bereits in Kapitel 2.3.2.2 erläutert, stehen intrinsischen Verfahren zur Abgrenzung von Plagiaten nur Trainingsdaten aus der nicht als Plagiat zu kennzeichnenden Klasse zur Verfügung. Die beschriebene Problemstellung entspricht einem *One-Class*-Klassifikationsproblem (vgl. [Stein et al. 2011, 63]). Das Fehlen von Trainingsdaten einer zweiten Klasse erschwert hierbei die Abgrenzung der vorhandenen Klasse. Die in Kapitel 3.7.3 beleuchteten intrinsischen Verfahren erzielen vergleichsweise geringe *Recall*-Werte. Letztere entstehen, wenn Plagiate nicht markiert und somit als *false negatives* gewertet werden. Im Rahmen eines hybriden Ansatzes, welcher intrinsische Methoden mit dem untersuchten Verfahren kombiniert, könnten bessere *Recall*-Werte erreicht werden. Zudem ließen sich somit auch Plagiate identifizieren, die nicht auf einen abweichenden Dokumenttyp zurückzuführen sind. Abbildung 22 veranschaulicht den beschriebenen Ansatz.



A	intrinsisch als *positives* klassifiziert
B	intrinsisch und via Dokumenttypklassifikation als *positives* klassifiziert
C	via Dokumenttypklassifikation als *positives* klassifiziert
D	verbleibende Einordnungen als *negatives*
A ∪ B ∪ C ∪ D	Menge der klassifizierten Sätze

Abbildung 22: Hybrider Ansatz zur Verbesserung intrinsischer Ansätze

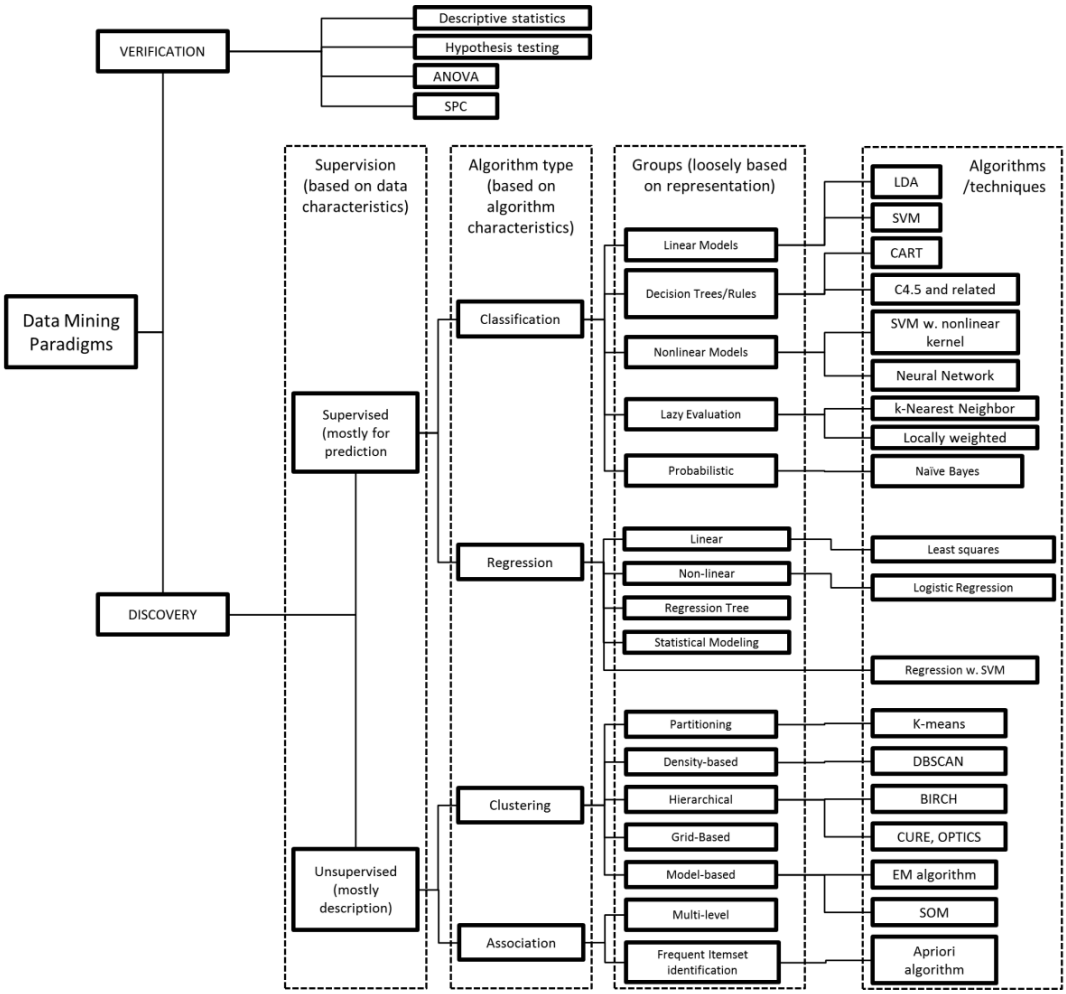
Die in Menge B enthaltenen Segmente könnten als besonders kritisch markiert werden, da diese von beiden Verfahren erkannt wurden. Werden Menge A und C als *positives* betrachtet, so sinkt die Anzahl der erfassten *negatives* in Menge D. Die steigende Menge an *false positives* ginge mit einem niedrigeren *Precision*-Wert einher. Wie bereits in Kapitel 3.7.3 beschrieben, kann eine Steigerung des *Recall*-Werts im Kontext intrinsischer Plagiaterkennung dennoch sinnvoll sein. Die markierten Segmente sind stets nur als plagiatsverdächtig zu interpretieren. Für eine Bestätigung der Plagiate muss die Zuordnung eines konkreten Dokuments erfolgen. Hierbei wird auf manuelle Recherche oder extrinsische Verfahren zurückgegriffen (vgl. [Gipp 2014, 19]). Eine größere Menge an *positives* erhöht den Aufwand dieses Validierungsschrittes, aber steigert die Wahrscheinlichkeit, dass alle realen Plagiate durch die Vorauswahl erfasst wurden.

4 Fazit

In der vorliegenden Arbeit wurde ein Klassifikationsverfahren entwickelt und evaluiert, welches anhand stilistischer Merkmale den Dokumenttyp eines eingegebenen Textabschnittes bestimmt. Studentische Abschlussarbeiten können nach der untersuchten Methodik auf Segmente untersucht werden, welche nicht dem durchschnittlichen Stil einer Bachelor- oder Masterarbeit entsprechen. Im Anwendungskontext der Plagiatanalyse konnten 53,62 % der künstlichen Plagiate und 71,03 % der Segmente aus realen Bachelor- und Masterarbeiten korrekt den Kategorien C_{pos} *(Fachbuch, Fachartikel, Wikipediaartikel)* und

C_{neg} *(Bachelorarbeit, Masterarbeit)* zugeordnet werden. Das beschriebene Verfahren ist somit prinzipiell für die automatisierte Plagiatanalyse studentischer Abschlussarbeiten geeignet. Für einen praktischen Einsatz sollte insbesondere der vergleichsweise niedrige *Precision*-Wert verbessert werden. Der erreichte *Recall*-Wert von 0,536 ist im Vergleich zu intrinsischen Verfahren mit ähnlicher Gesamtperformanz deutlich höher. Weiterführende Untersuchungen müssen prüfen, inwiefern die hohen *Recall*-Werte des entwickelten Verfahrens geeignet sind, um im Rahmen eines hybriden Ansatzes die Ergebnisse intrinsischer Verfahren zu verbessern. Das in Kapitel 3.8 dargestellte Konzept kann als Grundlage für die Implementierung eines Prototyps auf Basis des entwickelten Verfahrens genutzt werden. Die aus den Trainingskorpora extrahierten *features* wurden als ARFF-Dateien zur Verfügung gestellt.

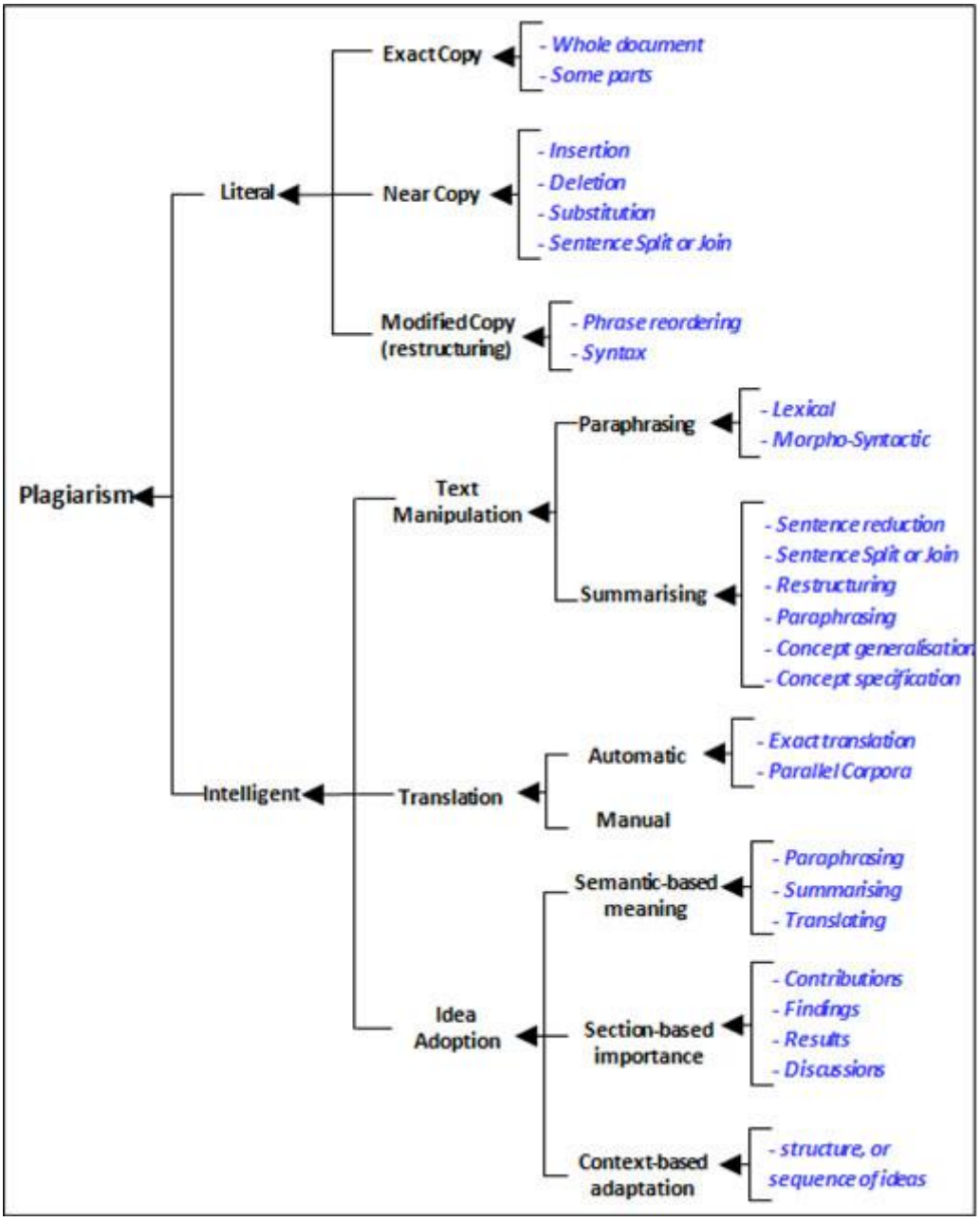
Anhang



Anhang A: Taxonomie für Data-Mining-Algorithmen [Gavrilovski et al. 2016, 4]

Format	▲
☐ Artikel	16.157 ⊟
☐ Zeitschrift, E-Journal	9.245 ⊟
☐ Hochschulschrift	5.443 ⊟
☐ Buch, E-Book	5.352 ⊟

Anhang B: Dokumenttypklassifikation der Universitätsbibliothek Leipzig [o. V. 2018d]



Anhang C: Arten des Plagiarismus [Alzahrani et al. 2012, 134]

	S	**1**																
	N_{Train}	**11399*5**																
A	D_R | D_Z	Bachelor-arbeit	Fach-buch	Fach-artikel	Master-arbeit	Wikipe-diaartikel	N_{Test} [D_R]	N_{Test} [C_{pos}]	N_{Test} [C_{neg}]	TP	FP	TN	FN	E (Prozent)	2*SE (Prozentpunkte)	R	P	**F_1**
NB	Bache-lorarbeit	5318	141	906	189	278	6832	13662	13664	5562	3086	10578	8100	40,94	0,60	0,407	0,643	**0,499**
	Fach-buch	2579	229	1234	154	358	4554											
	Fachartikel	2616	189	1253	151	345	4554											
	Master-arbeit	4847	226	1035	224	500	6832											
	Wikipe-diaartikel	2510	103	491	90	1360	4554											
SVM	Bache-lorarbeit	3551	549	1155	1106	471	6832	13662	13664	9086	5144	8520	4576	35,57	0,60	0,665	0,639	**0,652**
	Fach-buch	1011	753	1586	668	536	4554											
	Fachartikel	962	551	1864	722	455	4554											
	Master-arbeit	2520	787	1378	1343	804	6832											
	Wikipe-diaartikel	789	326	717	424	2298	4554											
C4.5	Bache-lorarbeit	1927	1313	1167	1479	946	6832	13662	13664	8374	7163	6501	5288	45,57	0,58	0,613	0,539	**0,574**
	Fach-buch	865	1051	997	971	670	4554											
	Fachartikel	877	970	1144	917	646	4554											
	Master-arbeit	1592	1378	1331	1503	1028	6832											
	Wikipe-diaartikel	890	770	696	768	1430	4554											

	S	**5**																
	N_{Train}	**2279*5**																
A	$D_R \mid D_Z$	Bachelor-arbeit	Fach-buch	Fach-artikel	Master-arbeit	Wikipe-diaartikel	N_{Test} [D_R]	N_{Test} [C_{pos}]	N_{Test} [C_{neg}]	TP	FP	TN	FN	E (Prozent)	2*SE (Prozentpunkte)	R	P	**F_1**
NB	Bache-lorarbeit	1170	150	113	143	34	1610	3219	3220	1760	714	2506	1459	33,75	1,18	0,547	0,711	**0,618**
	Fach-buch	297	339	268	158	11	1073											
	Fachartikel	386	209	307	135	36	1073											
	Master-arbeit	971	245	138	222	34	1610											
	Wikipe-diaartikel	388	55	45	95	490	1073											
SVM	Bache-lorarbeit	811	103	172	463	61	1610	3219	3220	2616	938	2282	603	23,93	1,20	0,813	0,736	**0,772**
	Fach-buch	65	328	402	220	58	1073											
	Fachartikel	87	172	602	165	47	1073											
	Master-arbeit	370	221	239	638	142	1610											
	Wikipe-diaartikel	17	30	18	49	959	1073											
C4.5	Bache-lorarbeit	627	214	239	378	152	1610	3219	3220	2325	1473	1747	894	36,76	1,06	0,722	0,612	**0,663**
	Fach-buch	153	301	289	201	129	1073											
	Fachartikel	178	251	373	163	108	1073											
	Master-arbeit	411	355	309	331	204	1610											
	Wikipe-diaartikel	99	79	87	100	708	1073											

	S	**10**																
	N_{Train}	**1139*5**																
A	D_R | D_Z	Bachelor-arbeit	Fach-buch	Fach-artikel	Master-arbeit	Wikipe-diaartikel	N_{Test} [D_R]	N_{Test} [C_{pos}]	N_{Test} [C_{neg}]	TP	FP	TN	FN	E (Prozent)	2*SE (Prozentpunkte)	R	P	**F_1**
NB	Bache-lorarbeit	599	58	39	76	35	807	1614	1614	1234	353	1261	380	22,71	1,48	0,765	0,778	**0,771**
	Fach-buch	98	229	126	64	21	538											
	Fachartikel	123	124	176	72	43	538											
	Master-arbeit	433	132	52	153	37	807											
	Wikipe-diaartikel	14	25	18	9	472	538											
SVM	Bache-lorarbeit	544	25	35	202	1	807	1614	1614	1361	260	1354	253	15,89	1,65	0,843	0,840	**0,841**
	Fach-buch	36	188	202	103	9	538											
	Fachartikel	34	66	372	64	2	538											
	Master-arbeit	217	105	75	391	19	807											
	Wikipe-diaartikel	4	4	3	12	515	538											
C4.5	Bache-lorarbeit	374	90	96	229	18	807	1614	1614	1052	490	1124	562	32,59	1,29	0,652	0,682	**0,667**
	Fach-buch	96	149	149	120	24	538											
	Fachartikel	93	116	198	111	20	538											
	Master-arbeit	263	130	131	258	25	807											
	Wikipe-diaartikel	72	80	65	70	251	538											

	S	**15**															
	N_{Train}	**759*5**															
A	D_R | D_Z	Bachelor-arbeit	Fach-buch	Fach-artikel	Master-arbeit	Wikipe-diaartikel	N_{Test} [D_R]	N_{Test} [C_{pos}]	N_{Test} [C_{neg}]	TP	FP	TN	FN	E (Prozent)	2*SE (Prozentpunkte)	R	P	**F_1**
NB	Bache-lorarbeit	366	50	27	84	10	537	1074	1074	871	238	836	203	20,53	1,74	0,811	0,785	**0,798**
	Fach-buch	25	202	76	49	6	358											
	Fachartikel	56	87	144	61	10	358											
	Master-arbeit	245	95	45	141	11	537											
	Wikipe-diaartikel	9	11	11	3	324	358											
SVM	Bache-lorarbeit	378	20	18	119	2	537	1074	1074	944	149	925	130	12,99	1,95	0,879	0,864	**0,871**
	Fach-buch	17	184	97	57	3	358											
	Fachartikel	15	66	243	33	1	358											
	Master-arbeit	155	57	44	273	8	537											
	Wikipe-diaartikel	1	5	0	7	345	358											
C4.5	Bache-lorarbeit	254	60	53	156	14	537	1074	1074	807	347	727	267	28,59	1,45	0,751	0,699	**0,724**
	Fach-buch	41	125	105	71	16	358											
	Fachartikel	48	101	145	53	11	358											
	Master-arbeit	143	93	98	174	29	537											
	Wikipe-diaartikel	30	25	40	24	239	358											

	S	**20**															
	N_{Train}	**599*5**															
A	$D_R \mid D_Z$	Bachelor-arbeit	Fach-buch	Fach-artikel	Master-arbeit	Wikipe-diaartikel	N_{Test} [D_R]	N_{Test} [C_{pos}]	N_{Test} [C_{neg}]	TP	FP	TN	FN	E (Prozent)	2*SE (Prozentpunkte)	R	P	**F_1**
NB	Bache-lorarbeit	306	26	12	54	4	402	804	804	655	121	683	149	16,79	1,86	0,815	0,844	**0,829**
	Fach-buch	10	152	55	49	2	268											
	Fachartikel	41	69	114	42	2	268											
	Master-arbeit	167	57	21	156	1	402											
	Wikipe-diaartikel	3	8	2	4	251	268											
SVM	Bache-lorarbeit	299	13	9	78	3	402	804	804	736	89	715	68	9,76	2,13	0,915	0,892	**0,904**
	Fach-buch	5	154	66	41	2	268											
	Fachartikel	6	46	200	16	0	268											
	Master-arbeit	109	42	21	229	1	402											
	Wikipe-diaartikel	0	1	0	0	267	268											
C4.5	Bache-lorarbeit	205	32	39	120	6	402	804	804	670	254	550	134	24,13	1,48	0,833	0,725	**0,775**
	Fach-buch	22	106	87	40	13	268											
	Fachartikel	21	73	132	35	7	268											
	Master-arbeit	113	77	85	112	15	402											
	Wikipe-diaartikel	10	11	8	6	233	268											

Anhang D: Ausführliche Ergebnisse der auf *Writeprints* basierenden Modelle im Kontext der technischen Evaluation

	S	**1**																
	N_{Train}	**54652*5**																
A	D_R | D_Z	Bachelor-arbeit	Fach-buch	Fach-artikel	Master-arbeit	Wikipe-diaartikel	N_{Test} [D_R]	N_{Test} [C_{pos}]	N_{Test} [C_{neg}]	TP	FP	TN	FN	E (Prozent)	2*SE (Prozentpunkte)	R	P	**F_1**
NB	Bachelor-arbeit	2198	99	1239	396	2900	6832	13662	13664	9213	8538	5126	4449	47,53	0,60	0,674	0,519	**0,587**
	Fachbuch	1190	97	1376	281	1610	4554											
	Fachartikel	1297	94	1476	368	1319	4554											
	Masterarbeit	2061	62	1360	471	2878	6832											
	Wikipediaartikel	1074	41	735	239	2465	4554											
SVM	Bachelor-arbeit	2337	968	680	9	2838	6832	13662	13664	10433	9250	4414	3229	45,67	0,60	0,764	0,530	**0,626**
	Fachbuch	923	914	1030	9	1678	4554											
	Fachartikel	1250	820	1123	11	1350	4554											
	Masterarbeit	2059	994	877	9	2893	6832											
	Wikipediaartikel	1033	532	455	3	2531	4554											
C4.5	Bachelor-arbeit	2568	934	929	1140	1261	6832	13662	13664	8452	6990	6674	5210	44,65	0,60	0,619	0,547	**0,581**
	Fachbuch	886	986	987	811	884	4554											
	Fachartikel	853	903	1185	820	793	4554											
	Masterarbeit	1622	1202	1217	1344	1447	6832											
	Wikipediaartikel	1009	831	702	831	1181	4554											

	S	**5**																
	N_{Train}	**12878*5**																
A	D_R | D_Z	Bachelor-arbeit	Fach-buch	Fach-artikel	Master-arbeit	Wikipe-diaartikel	N_{Test} [D_R]	N_{Test} [C_{pos}]	N_{Test} [C_{neg}]	TP	FP	TN	FN	E (Prozent)	2*SE (Prozentpunkte)	R	P	**F_1**
NB	Bachelor-arbeit	1028	95	212	120	155	1610	3219	3220	1630	1037	2183	1589	40,78	1,23	0,506	0,611	**0,554**
	Fachbuch	385	195	370	73	50	1073											
	Facharti-kel	381	128	391	76	97	1073											
	Masterar-beit	942	122	295	93	158	1610											
	Wikipe-diaartikel	603	65	76	71	258	1073											
SVM	Bachelor-arbeit	618	194	186	34	578	1610	3219	3220	2660	2049	1171	559	40,50	1,24	0,826	0,565	**0,671**
	Fachbuch	147	283	331	27	285	1073											
	Facharti-kel	186	199	442	19	227	1073											
	Masterar-beit	478	255	278	41	558	1610											
	Wikipe-diaartikel	166	60	104	14	729	1073											
C4.5	Bachelor-arbeit	438	259	247	322	344	1610	3219	3220	2060	1738	1482	1159	44,99	1,22	0,640	0,542	**0,587**
	Fachbuch	185	240	281	185	182	1073											
	Facharti-kel	178	220	342	186	147	1073											
	Masterar-beit	394	322	248	328	318	1610											
	Wikipe-diaartikel	233	175	138	192	335	1073											

	S	**10**																
	N_{Train}	**6456*5**																
A	D_R | D_Z	Bachelor-arbeit	Fach-buch	Fach-artikel	Master-arbeit	Wikipe-diaartikel	N_{Test} [D_R]	N_{Test} [C_{pos}]	N_{Test} [C_{neg}]	TP	FP	TN	FN	E (Prozent)	2*SE (Prozentpunkte)	R	P	**F_1**
NB	Bachelor-arbeit	521	94	48	67	77	807	1614	1614	882	488	1126	732	37,79	1,71	0,546	0,644	**0,591**
	Fachbuch	149	135	181	52	21	538											
	Fachartikel	128	76	227	70	37	538											
	Masterarbeit	455	113	83	83	73	807											
	Wikipediaartikel	286	30	29	47	146	538											
SVM	Bachelor-arbeit	364	117	61	34	231	807	1614	1614	1368	926	688	246	36,31	1,72	0,848	0,596	**0,700**
	Fachbuch	66	185	141	20	126	538											
	Fachartikel	60	129	267	16	66	538											
	Masterarbeit	239	126	103	51	288	807											
	Wikipediaartikel	70	20	43	14	391	538											
C4.5	Bachelor-arbeit	268	122	83	189	145	807	1614	1614	1136	795	819	478	39,44	1,69	0,704	0,588	**0,641**
	Fachbuch	76	118	155	85	104	538											
	Fachartikel	53	127	213	78	67	538											
	Masterarbeit	188	128	126	174	191	807											
	Wikipediaartikel	107	71	43	79	238	538											

	S	**15**																
	N_{Train}	**4296*5**																
A	D_R | D_Z	Bachelor-arbeit	Fach-buch	Fach-artikel	Master-arbeit	Wikipe-diaartikel	N_{Test} [D_R]	N_{Test} [C_{pos}]	N_{Test} [C_{neg}]	TP	FP	TN	FN	E (Prozent)	2*SE (Prozentpunkte)	R	P	**F_1**
NB	Bachelor-arbeit	336	68	19	61	53	537	1074	1074	647	311	763	427	34,36	2,05	0,602	0,675	**0,637**
	Fachbuch	39	155	88	62	14	358											
	Fachartikel	71	53	147	64	23	358											
	Masterarbeit	255	83	42	111	46	537											
	Wikipediaartikel	164	15	17	27	135	358											
SVM	Bachelor-arbeit	274	69	30	29	135	537	1074	1074	928	565	509	146	33,10	2,11	0,864	0,622	**0,723**
	Fachbuch	18	150	89	21	80	358											
	Fachartikel	33	97	160	26	42	358											
	Masterarbeit	162	94	55	44	182	537											
	Wikipediaartikel	39	8	22	9	280	358											
C4.5	Bachelor-arbeit	199	88	46	101	103	537	1074	1074	760	543	531	314	39,90	2,03	0,708	0,583	**0,639**
	Fachbuch	31	103	94	69	61	358											
	Fachartikel	47	79	151	58	23	358											
	Masterarbeit	126	101	81	105	124	537											
	Wikipediaartikel	51	30	29	58	190	358											

	S	**20**																
	N_{Train}	**3216*5**																
A	D_R | D_Z	Bachelor-arbeit	Fach-buch	Fach-artikel	Master-arbeit	Wikipe-diaartikel	N_{Test} [D_R]	N_{Test} [C_{pos}]	N_{Test} [C_{neg}]	TP	FP	TN	FN	E (Prozent)	2*SE (Prozentpunkte)	R	P	**F_1**
NB	Bachelor-arbeit	267	31	20	61	23	402	804	804	493	169	635	311	29,85	2,28	0,613	0,745	**0,673**
	Fachbuch	31	71	109	41	16	268											
	Facharti-kel	46	36	127	43	16	268											
	Masterar-beit	219	46	29	88	20	402											
	Wikipe-diaartikel	114	12	11	36	95	268											
SVM	Bachelor-arbeit	230	41	20	36	75	402	804	804	725	361	443	79	27,36	2,35	0,902	0,668	0,767
	Fachbuch	13	132	57	14	52	268											
	Facharti-kel	22	52	150	16	28	268											
	Masterar-beit	123	57	38	54	130	402											
	Wikipe-diaartikel	12	7	16	2	231	268											
C4.5	Bachelor-arbeit	182	46	36	77	61	402	804	804	628	360	444	176	33,33	2,22	0,781	0,636	**0,701**
	Fachbuch	26	100	63	28	51	268											
	Facharti-kel	36	64	107	35	26	268											
	Masterar-beit	94	64	42	91	111	402											
	Wikipe-diaartikel	24	23	12	27	182	268											

Anhang E: Ausführliche Ergebnisse der auf *8-Features* basierenden Modelle im Kontext der technischen Evaluation

Writeprints, S=10*, Trainingsdatensatz, *features

Character count, Average characters per word, Characters-{a}, Characters-{u}, Characters-{s}, Characters-{b}, Characters-{l}, Characters-{i}, Characters-{c}, Characters-{k}, Characters-{.}, Characters-{ }, Characters-{\n}, Characters-{d}, Characters-{e}, Characters-{r}, Characters-{g}, Characters-{o}, Characters-{t}, Characters-{h}, Characters-{m}, Characters-{v}, Characters-{w}, Characters-{n}, Characters-{z}, Characters-{p}, Characters-{y}, Characters-{f}, Characters-{ä}, Characters-{x}, Characters-{j}, Characters-{ö}, Characters-{ß}, Characters-{null}, Characters-{q}, Characters-{ü}, Characters-{?}, Characters-{!}, Top-character-bigrams{au}, Top-character-bigrams{ic}, Top-character-bigrams{. }, Top-character-bigrams{ \n}, Top-character-bigrams{de}, Top-character-bigrams{er}, Top-character-bigrams{r }, Top-character-bigrams{ a}, Top-character-bigrams{al}, Top-character-bigrams{it}, Top-character-bigrams{s }, Top-character-bigrams{ d}, Top-character-bigrams{ v}, Top-character-bigrams{en}, Top-character-bigrams{nd}, Top-character-bigrams{t }, Top-character-bigrams{ w}, Top-character-bigrams{n }, Top-character-bigrams{ s}, Top-character-bigrams{ u}, Top-character-bigrams{di}, Top-character-bigrams{ie}, Top-character-bigrams{e }, Top-character-bigrams{el}, Top-character-bigrams{re}, Top-character-bigrams{on}, Top-character-bigrams{ b}, Top-character-bigrams{be}, Top-character-bigrams{ch}, Top-character-bigrams{ne}, Top-character-bigrams{d }, Top-character-bigrams{ i}, Top-character-bigrams{in}, Top-character-bigrams{se}, Top-character-bigrams{ge}, Top-character-bigrams{es}, Top-character-bigrams{st}, Top-character-bigrams{te}, Top-character-bigrams{un}, Top-character-bigrams{an}, Top-character-bigrams{ng}, Top-character-bigrams{ei}, Top-character-bigrams{ti}, Top-character-bigrams{ e}, Top-character-bigrams{nt}, Top-character-bigrams{le}, Top-character-bigrams{he}, Top-character-bigrams{is}, Top-character-bigrams{sc}, Top-character-bigrams{nullnull}, Top-character-trigrams{. \n}, Top-character-trigrams{der}, Top-character-trigrams{er }, Top-character-trigrams{ de}, Top-character-trigrams{ver}, Top-character-trigrams{nde}, Top-character-trigrams{den}, Top-character-trigrams{en }, Top-character-trigrams{ di}, Top-character-trigrams{die}, Top-character-trigrams{ie }, Top-character-trigrams{ion}, Top-character-trigrams{on }, Top-character-trigrams{ zu}, Top-character-trigrams{ be}, Top-character-trigrams{ere}, Top-character-trigrams{ in}, Top-character-trigrams{in }, Top-character-trigrams{ da}, Top-character-trigrams{ste}, Top-character-trigrams{und}, Top-character-trigrams{gen}, Top-character-trigrams{n. }, Top-character-trigrams{n d}, Top-character-trigrams{ung}, Top-character-trigrams{ng }, Top-character-trigrams{ au}, Top-character-trigrams{cht}, Top-character-trigrams{ten}, Top-character-trigrams{eit}, Top-character-trigrams{te }, Top-character-trigrams{ vo}, Top-character-trigrams{ ei}, Top-character-trigrams{ein}, Top-character-trigrams{ un}, Top-character-trigrams{nd }, Top-character-trigrams{ ge}, Top-character-trigrams{ent}, Top-character-trigrams{nte}, Top-character-trigrams{ter}, Top-character-trigrams{ich}, Top-character-trigrams{ch }, Top-character-trigrams{nge}, Top-character-trigrams{hen}, Top-character-trigrams{ine}, Top-character-trigrams{sch}, Top-character-trigrams{che}, Top-character-trigrams{es }, Top-character-trigrams{nullnullnull}, Top-character-trigrams{ers}, Character Percentage, Uppercase Letters Percentage, Word-Lengths{8}, Word-Lengths{3}, Word-Lengths{11}, Word-Lengths{9}, Word-Lengths{6}, Word-Lengths{4}, Word-Lengths{2}, Word-Lengths{10}, Word-Lengths{5}, Word-Lengths{7}, Word-Lengths{12}, Word-Lengths{1}, Word-Lengths{21}, Word-Lengths{13}, Word-Lengths{18}, Word-Lengths{14}, Word-Lengths{29}, Word-Lengths{16}, Word-Lengths{15}, Word-Lengths{19}, Word-Lengths{17}, Word-Lengths{20}, Word-Lengths{24}, Word-Lengths{31}, Word-Lengths{23}, Word-Lengths{25}, Word-Lengths{22}, Word-Lengths{26}, Word-Lengths{28}, Word-Lengths{32}, Word-Lengths{27}, Word-Lengths{30}, Word-Lengths{33}, Word-Lengths{36}, Word-Lengths{42}, Word-Lengths{35}, Word-Lengths{34}, Word-Lengths{41}, Word-Lengths{57}, Word-Lengths{53}, Word-Lengths{38}, Word-Lengths{44}, Word-Lengths{43}, Word-Lengths{37}, Word-Lengths{56}, Word-Lengths{87}, Word-Lengths{65}, Word-Lengths{39}, Word-Lengths{58}, Word-Lengths{40}, Word-Lengths{54}, Word-Lengths{49}, Word-Lengths{51}, Word-Lengths{52}, Word-Lengths{60}, Word-Lengths{55}, Word-Lengths{98}, Word-Lengths{48}, Word-Lengths{47}, Word-Lengths{45}, Word-Lengths{103}, Word-Lengths{50}, Word-Lengths{46}, Word-Lengths{83}, Word-Lengths{59}, Word-Lengths{61}, Word-Lengths{79}, Word-Lengths{101}, Word-Lengths{66}, Word-Lengths{97}, Word-Lengths{75}, Word-Lengths{86}, Word-Lengths{62}, Word-Lengths{68}, Word-Lengths{73}, Word-Lengths{63}, Word-Lengths{85}, Word-Lengths{89}, Word-Lengths{67}, Word-Lengths{69}, Word-Lengths{71}, Word-Lengths{70}, Word-Lengths{107}, Word-Lengths{72}, Word-Lengths{64}, Function-Words{der}, Function-Words{werden}, Function-Words{soll}, Function-Words{um}, Function-Words{die}, Function-Words{zu}, Function-Words{wird}, Function-Words{in}, Function-Words{als}, Function-Words{damit}, Function-Words{kann}, Function-Words{auf}, Function-Words{von}, Function-Words{bei}, Function-Words{und}, Function-Words{lassen}, Function-Words{sich}, Function-Words{nicht}, Function-Words{im}, Function-Words{da}, Function-Words{das}, Function-Words{den}, Function-Words{vom}, Function-Words{du}, Function-Words{zum}, Function-Words{sie}, Function-Words{dies}, Function-Words{dann}, Function-Words{aber}, Function-Words{dadurch}, Function-Words{ist}, Function-Words{auch}, Function-Words{für}, Function-Words{durch}, Function-Words{mit}, Function-Words{machen}, Function-Words{eines}, Function-Words{jedes}, Function-Words{oder}, Function-Words{es}, Function-

Words{nun}, Function-Words{dieser}, Function-Words{ein}, Function-Words{aus}, Function-Words{müssen}, Function-Words{einen}, Function-Words{immer}, Function-Words{wenn}, Function-Words{eine}, Function-Words{dem}, Function-Words{des}, Function-Words{dieses}, Function-Words{nur}, Function-Words{nachdem}, Function-Words{mehr}, Function-Words{sind}, Function-Words{nach}, Function-Words{sein}, Function-Words{an}, Function-Words{zur}, Function-Words{jede}, Function-Words{einem}, Function-Words{andere}, Function-Words{wie}, Function-Words{vor}, Function-Words{können}, Function-Words{doch}, Function-Words{so}, Function-Words{ich}, Function-Words{jeder}, Function-Words{man}, Function-Words{was}, Function-Words{denn}, Function-Words{am}, Function-Words{dass}, Function-Words{mich}, Function-Words{weiter}, Function-Words{dessen}, Function-Words{daher}, Function-Words{unter}, Function-Words{noch}, Function-Words{einer}, Function-Words{haben}, Function-Words{sehr}, Function-Words{er}, Function-Words{jedem}, Function-Words{weil}, Function-Words{weitere}, Function-Words{über}, Function-Words{ihre}, Function-Words{hier}, Function-Words{wieder}, Function-Words{wir}, Function-Words{sehen}, Function-Words{dort}, Function-Words{mir}, Function-Words{bis}, Function-Words{selbst}, Function-Words{also}, Function-Words{sowie}, Function-Words{hatten}, Function-Words{zwei}, Function-Words{jeden}, Function-Words{seine}, Function-Words{sollen}, Function-Words{lange}, Function-Words{wissen}, Function-Words{oben}, Function-Words{ihr}, Function-Words{ganz}, Function-Words{kommen}, Function-Words{wo}, Function-Words{gut}, Function-Words{hatte}, Function-Words{schon}, Function-Words{stehen}, Function-Words{deshalb}, Function-Words{neu}, Function-Words{kein}, Function-Words{jetzt}, Function-Words{viel}, Function-Words{anderer}, Function-Words{uns}, Function-Words{gehen}, Function-Words{erste}, Function-Words{ihn}, Function-Words{ja}, Function-Words{geben}, Function-Words{ihm}, Function-Words{daß}, Punctuation{?}, Punctuation{!}, POS-Tags{NNP}, POS-Tags{.}, POS-Tags{FW}, POS-Tags{NN}, POS-Tags{VB}, POS-Tags{IN}, POS-Tags{VBZ}, POS-Tags{NNS}, POS-Tags{JJ}, POS-Tags{VBD}, POS-Tags{WP}, POS-Tags{RB}, POS-Tags{NNPS}, POS-Tags{VBP}, POS-Tags{CC}, POS-Tags{DT}, POS-Tags{TO}, POS-Tags{JJR}, POS-Tags{JJS}, POS-Tags{SYM}, POS-Tags{VBG}, POS-Tags{PRP}, POS-Tags{CD}, POS-Tags{MD}, POS-Tags{VBN}, POS-Tags{WDT}, POS-Tags{EX}, POS-Tags{WRB}, POS-Tags{PRP$}, POS-Tags{LS}, POS-Tags{RBR}, POS-Tags{UH}, POS-Tags{PDT}, POS-Tags{RP}, POS-Tags{$}, POS-Tags{RBS}, POS-Tags{WP$}, POS-Bigrams{(NNP)-(.)}, POS-Bigrams{(.)-(NNP)}, POS-Bigrams{(NNP)-(NNP)}, POS-Bigrams{(NNP)-(FW)}, POS-Bigrams{(FW)-(FW)}, POS-Bigrams{(FW)-(NN)}, POS-Bigrams{(NN)-(NN)}, POS-Bigrams{(NN)-(VB)}, POS-Bigrams{(VB)-(NNP)}, POS-Bigrams{(NNP)-(NN)}, POS-Bigrams{(NN)-(IN)}, POS-Bigrams{(IN)-(NNP)}, POS-Bigrams{(NNP)-(VBZ)}, POS-Bigrams{(VBZ)-(NN)}, POS-Bigrams{(NN)-(.)}, POS-Bigrams{(.)-(NN)}, POS-Bigrams{(NN)-(NNS)}, POS-Bigrams{(NNP)-(VB)}, POS-Bigrams{(NN)-(FW)}, POS-Bigrams{(NN)-(NNP)}, POS-Bigrams{(NNP)-(NNS)}, POS-Bigrams{(NNS)-(NN)}, POS-Bigrams{(.)-(JJ)}, POS-Bigrams{(JJ)-(NN)}, POS-Bigrams{(NNP)-(JJ)}, POS-Bigrams{(VB)-(NN)}, POS-Bigrams{(NN)-(JJ)}, POS-Bigrams{(FW)-(NNP)}, POS-Bigrams{(IN)-(FW)}, POS-Bigrams{(NNS)-(JJ)}, POS-Bigrams{(NNS)-(NNP)}, POS-Bigrams{(VBZ)-(NNP)}, POS-Bigrams{(VB)-(JJ)}, POS-Bigrams{(RB)-(JJ)}, POS-Bigrams{(NN)-(VBZ)}, POS-Bigrams{(NNP)-(VBD)}, POS-Bigrams{(NNS)-(VBP)}, POS-Bigrams{(VBP)-(NN)}, POS-Bigrams{(.)-(VB)}, POS-Bigrams{(JJ)-(NNP)}, POS-Bigrams{(NN)-(DT)}, POS-Bigrams{(DT)-(NN)}, POS-Bigrams{(NNP)-(IN)}, POS-Bigrams{(IN)-(NN)}, POS-Bigrams{(VBP)-(JJ)}, POS-Bigrams{(NNP)-(DT)}, POS-Bigrams{(DT)-(NNP)}, POS-Bigrams{(VBP)-(NNP)}, POS-Bigrams{(NN)-(VBP)}, POS-Bigrams{(.)-(IN)}, POS-Trigrams{(NNP)-(.)-(NNP)}, POS-Trigrams{(.)-(NNP)-(NNP)}, POS-Trigrams{(NNP)-(FW)-(FW)}, POS-Trigrams{(FW)-(FW)-(NN)}, POS-Trigrams{(FW)-(NN)-(NN)}, POS-Trigrams{(NN)-(NN)-(NN)}, POS-Trigrams{(NN)-(NN)-(VB)}, POS-Trigrams{(NN)-(VB)-(NNP)}, POS-Trigrams{(VB)-(NNP)-(NN)}, POS-Trigrams{(NNP)-(NN)-(NN)}, POS-Trigrams{(NN)-(NN)-(IN)}, POS-Trigrams{(VBZ)-(NN)-(NN)}, POS-Trigrams{(NN)-(NN)-(.)}, POS-Trigrams{(NN)-(.)-(NN)}, POS-Trigrams{(NN)-(FW)-(FW)}, POS-Trigrams{(FW)-(FW)-(FW)}, POS-Trigrams{(NN)-(NN)-(NNP)}, POS-Trigrams{(NN)-(.)-(NNP)}, POS-Trigrams{(.)-(NNP)-(NN)}, POS-Trigrams{(NNP)-(NNP)-(NNP)}, POS-Trigrams{(NNP)-(JJ)-(NN)}, POS-Trigrams{(JJ)-(NN)-(NN)}, POS-Trigrams{(NN)-(NNP)-(NN)}, POS-Trigrams{(NN)-(JJ)-(NN)}, POS-Trigrams{(.)-(NN)-(NN)}, POS-Trigrams{(NNP)-(NN)-(NNP)}, POS-Trigrams{(NN)-(NNP)-(FW)}, POS-Trigrams{(NNP)-(FW)-(NNP)}, POS-Trigrams{(FW)-(NNP)-(NN)}, POS-Trigrams{(NN)-(NNP)-(NNP)}, POS-Trigrams{(NNP)-(NNP)-(NN)}, POS-Trigrams{(IN)-(FW)-(FW)}, POS-Trigrams{(NN)-(NN)-(NNS)}, POS-Trigrams{(NNP)-(VB)-(NN)}, POS-Trigrams{(NNP)-(NN)-(.)}, POS-Trigrams{(NNP)-(NNP)-(.)}, POS-Trigrams{(VB)-(JJ)-(NN)}, POS-Trigrams{(NN)-(NN)-(FW)}, POS-Trigrams{(NN)-(NN)-(VBZ)}, POS-Trigrams{(NN)-(VBZ)-(NNP)}, POS-Trigrams{(NN)-(VB)-(JJ)}, POS-Trigrams{(VB)-(NNP)-(NNP)}, POS-Trigrams{(JJ)-(NN)-(NNP)}, POS-Trigrams{(IN)-(NN)-(NN)}, POS-Trigrams{(VB)-(NN)-(NN)}, POS-Trigrams{(VB)-(NNP)-(FW)}, POS-Trigrams{(NN)-(NN)-(JJ)}, POS-Trigrams{(NNP)-(IN)-(NNP)}, POS-Trigrams{(NN)-(IN)-(NN)}, POS-Trigrams{(NN)-(NNP)-(.)}, Words{der}, Words{werden}, Words{um}, Words{die}, Words{zu}, Words{wird}, Words{in}, Words{als}, Words{kann}, Words{auf}, Words{von}, Words{bei}, Words{und}, Words{sich}, Words{nicht}, Words{im}, Words{das}, Words{den}, Words{zum}, Words{sie}, Words{diese}, Words{ist}, Words{auch}, Words{für}, Words{durch}, Words{mit}, Words{oder}, Words{es}, Words{dieser}, Words{ein}, Words{aus}, Words{einen}, Words{eine}, Words{dem}, Words{des}, Words{nur},

Words{wurde}, Words{sind}, Words{nach}, Words{an}, Words{zur}, Words{einem}, Words{wie}, Words{so}, Words{dass}, Words{einer}, Words{er}, Words{über}, Words{a}, Words{s}, Word-Bigrams{(um)-(die)}, Word-Bigrams{(die)-(in)}, Word-Bigrams{(auf)-(der)}, Word-Bigrams{(sich)-(die)}, Word-Bigrams{(in)-(der)}, Word-Bigrams{(mit)-(der)}, Word-Bigrams{(werden)-(die)}, Word-Bigrams{(für)-(das)}, Word-Bigrams{(mit)-(den)}, Word-Bigrams{(wird)-(die)}, Word-Bigrams{(mit)-(dem)}, Word-Bigrams{(sich)-(in)}, Word-Bigrams{(in)-(den)}, Word-Bigrams{(durch)-(die)}, Word-Bigrams{(und)-(der)}, Word-Bigrams{(in)-(einem)}, Word-Bigrams{(von)-(den)}, Word-Bigrams{(ist)-(es)}, Word-Bigrams{(z)-(b)}, Word-Bigrams{(sind)-(die)}, Word-Bigrams{(auf)-(dem)}, Word-Bigrams{(auch)-(die)}, Word-Bigrams{(mit)-(einer)}, Word-Bigrams{(für)-(den)}, Word-Bigrams{(auf)-(die)}, Word-Bigrams{(aus)-(der)}, Word-Bigrams{(von)-(der)}, Word-Bigrams{(ist)-(der)}, Word-Bigrams{(für)-(die)}, Word-Bigrams{(zu)-(den)}, Word-Bigrams{(in)-(die)}, Word-Bigrams{(auf)-(den)}, Word-Bigrams{(bei)-(der)}, Word-Bigrams{(in)-(diesem)}, Word-Bigrams{(und)-(die)}, Word-Bigrams{(nach)-(dem)}, Word-Bigrams{(nach)-(der)}, Word-Bigrams{(aus)-(dem)}, Word-Bigrams{(ist)-(die)}, Word-Bigrams{(bei)-(den)}, Word-Bigrams{(in)-(einer)}, Word-Bigrams{(dass)-(die)}, Word-Bigrams{(ist)-(ein)}, Word-Bigrams{(zu)-(einer)}, Word-Bigrams{(an)-(den)}, Word-Bigrams{(mit)-(einem)}, Word-Bigrams{(über)-(die)}, Word-Bigrams{(an)-(der)}, Word-Bigrams{(und)-(in)}, Word-Bigrams{(vor)-(allem)}, Word-Bigrams{(an)-(die)}, Word-Trigrams{(aus)-(diesem)-(grund)}, Word-Trigrams{(handelt)-(es)-(sich)}, Word-Trigrams{(es)-(sich)-(um)}, Word-Trigrams{(auch)-(in)-(der)}, Word-Trigrams{(im)-(hinblick)-(auf)}, Word-Trigrams{(im)-(gegensatz)-(zu)}, Word-Trigrams{(im)-(rahmen)-(des)}, Word-Trigrams{(im)-(zusammenhang)-(mit)}, Word-Trigrams{(als)-(auch)-(die)}, Word-Trigrams{(im)-(bereich)-(der)}, Word-Trigrams{(der)-(in)-(der)}, Word-Trigrams{(die)-(in)-(der)}, Word-Trigrams{(an)-(der)-(universität)}, Word-Trigrams{(sich)-(in)-(den)}, Word-Trigrams{(in)-(diesem)-(fall)}, Word-Trigrams{(wird)-(in)-(der)}, Word-Trigrams{(in)-(den)-(letzten)}, Word-Trigrams{(an)-(dieser)-(stelle)}, Word-Trigrams{(die)-(ergebnisse)-(der)}, Word-Trigrams{(die)-(anzahl)-(der)}, Word-Trigrams{(proceedings)-(of)-(the)}, Word-Trigrams{(in)-(dieser)-(arbeit)}, Word-Trigrams{(im)-(vergleich)-(zu)}, Word-Trigrams{(einfluss)-(auf)-(die)}, Word-Trigrams{(dass)-(sich)-(die)}, Word-Trigrams{(in)-(der)-(lage)}, Word-Trigrams{(in)-(der)-(regel)}, Word-Trigrams{(die)-(frage)-(nach)}, Word-Trigrams{(auch)-(für)-(die)}, Word-Trigrams{(wie)-(z)-(b)}, Word-Trigrams{(sich)-(in)-(der)}, Word-Trigrams{(vor)-(allem)-(die)}, Word-Trigrams{(in)-(der)-(praxis)}, Word-Trigrams{(und)-(in)-(der)}, Word-Trigrams{(im)-(rahmen)-(der)}, Word-Trigrams{(frankfurt)-(am)-(main)}, Word-Trigrams{(d)-(h)-(die)}, Word-Trigrams{(auf)-(diese)-(weise)}, Word-Trigrams{(in)-(form)-(von)}, Word-Trigrams{(in)-(bezug)-(auf)}, Word-Trigrams{(in)-(den)-(er)}, Word-Trigrams{(den)-(er)-(jahren)}, Word-Trigrams{(es)-(handelt)-(sich)}, Word-Trigrams{(in)-(diesem)-(zusammenhang)}, Word-Trigrams{(vor)-(dem)-(hintergrund)}, Word-Trigrams{(der)-(er)-(jahre)}, Word-Trigrams{(schriftenreihe)-(des)-(lfulg)}, Word-Trigrams{(des)-(lfulg)-(heft)}, Word-Trigrams{(frankfurt)-(a)-(m)}, Word-Trigrams{(von)-(bis)-(war)}, Word-Trigrams{(war)-(ein)-(deutscher)}

Anhang F: Die bei einer Segmentierungsgröße von *S=10* durch *Writeprints* aus dem Trainingsdatensatz extrahierten Features

Literaturverzeichnis

Abbasi, A., Chen, H., Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace, in: ACM Transactions on Information Systems (TOIS), 26 (2008) 2, 7:1-7:29.

Afroz, S., Islam, A., Stolerman, A., Greenstadt, R., McCoy, D., Doppelgänger Finder: Taking Stylometry to the Underground, in: Proceedings of the 2014 IEEE Symposium on Security and Privacy, (2014), S. 212–226.

Aggarwal, C.C., Data classification, CRC Press, Boca Raton, FL, 2015.

Aggarwal, C.C., Zhai, C., Mining text data, Springer, New York, 2012.

Aggarwal, C.C., Zhai, C., Text Classification, in: Aggarwal, C.C. (Hrsg.), Data classification, CRC Press, Boca Raton, FL, 2015, S. 287–336.

Akiva, N., Using Clustering to Identify Outlier Chunks of Text, in: Notebook Papers of CLEF 2011 LABs and Workshops (2011).

Almishari, M., Oguz, E., Tsudik, G., Fighting authorship linkability with crowdsourcing, in: COSN '14 Proceedings of the second ACM conference on Online social networks (2014), S. 69–82.

Alpaydin, E., Introduction to machine learning, 2. Aufl., MIT Press, Cambridge, Mass, 2010.

Alzahrani, S.M., Salim, N., Abraham, A., Understanding Plagiarism Linguistic Patterns, Textual Features and Detection Methods, in: IEEE Transactions on Systems, Man, and Cybernetics, Part C, 42 (2012) 2, S. 133–149.

Argamon, S., Koppel, M., Pennebaker, J.W., Schler, J., Automatically profiling the author of an anonymous text, in: Communications of the ACM - Inspiring Women in Computing, 52 (2009) 2, S. 119–123.

Aumasson, J.-P., Meier, W., Phan, R.C.-W., Henzen, L., The Hash Function BLAKE, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

Barrón-Cedeño, A., Rosso, P., On Automatic Plagiarism Detection Based on n-Grams Comparison, in: ECIR 2009: Advances in Information Retrieval (2009), S. 696–700.

Beall, J., Criteria for Determining Predatory Open-Access Publishers, 2015, URL: https://beallslist.weebly.com/uploads/3/0/9/5/30958339/criteria-2015.pdf, gelesen am 16.07.2018.

Bedford, D., Evaluating classification schema and classification decisions, in: Bulletin of the American Society for Information Science and Technology, 39 (2012) 2, S. 13–21.

Bellinger, C., Sharma, S., Japkowicz, N., One-Class versus Binary Classification: Which and When?, in: 2012 11th International Conference on Machine Learning and Applications, 2 (2012).

Bock, H.H., Automatische Klassifikation, in: Walter, E. (Hrsg.), Statistische Methoden II, Springer, Berlin, Heidelberg, 1970, S. 36–80.

Boser, B.E., Guyon, I.M., Vapnik, V.N., A training algorithm for optimal margin classifiers, in: Haussler, D. (Hrsg.), Proceedings of the fifth annual workshop on Computational learning theory, ACM, New York, NY, 1992, S. 144–152.

Brennan, M., Afroz, S., Greenstadt, R., Adversarial stylometry, in: ACM Transactions on Information and System Security, 15 (2012) 3, S. 1–22.

Bretag, T., Mahmud, S., Self-Plagiarism or Appropriate Textual Re-use?, in: Journal of Academic Ethics, 7 (2009) 3, S. 193–205.

Caragea, C., Wu, J., Gollapalli, S., Giles, C., Document Type Classification in Online Digital Libraries, Proceedings of the thirtieth AAAI conference (AAAI-16), 2016, S. 3997–4002.

Ceska, Z., Fox, C., The Influence of Text Pre-processing on Plagiarism Detection, in: Proceedings of the International Conference RANLP 2009 (2009), S. 55–59.

Chandola, V., Kumar, V., Summarization – compressing data into an informative representation, in: Knowledge and Information Systems, 12 (2007) 3, S. 355–378.

Cios, K.J., Kurgan, L.A., Pedrycz, W., Swiniarski, R.W., Data Mining, Springer Science+Business Media LLC, Boston, MA, 2007.

Clough, P., Plagiarism in Natural and Programming Languages - An Overview of Current Tools and Technologies, University of Sheffield - Department of Computer Science, Sheffield, 2000.

Culotta, A., Sorensen, J., Dependency Tree Kernels for Relation Extraction, in: Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics, (2004).

Dewan, P., Kashyap, A., Kumaraguru, P., Analyzing social and stylometric features to identify spear phishing emails, in: Proceedings of the 2014 APWG Symposium on Electronic Crime Research, (2014), S. 1–13.

Dodd, B., Working with German corpora, Continuum, London, 2006.

Dolev, S., Lodha, S., Cyber security cryptography and machine learning, Springer, Cham, 2017.

Domingos, P., A Few Useful Things to Know About Machine Learning, in: Communications of the ACM, 55 (2012) 10, S. 78–87.

Dregvaite, G., Damasevicius, R., Information and Software Technologies, Springer International Publishing, Cham, s.l., 2015.

Eissen, S.M.z., Stein, B., Intrinsic Plagiarism Detection, in: Proceedings of the 28th European Conference on Advances in Information Retrieval, (2006), S. 565–569.

Elhadi, M., Al-Tobi, A., Use of text syntactical structures in detection of document duplicates, in: Proceedings of the third International Conference on Digital Information Management, (2008), S. 520–525.

Fishman, T., "We know it when we see it" is not good enough: toward a standard definition of plagiarism that transcends theft, fraud, and copyright, in: Proceedings of the 4th Asia Pacific Conference on Educational Integrity (2009).

Flesch, R., A new readability yardstick, in: Journal of Applied Psychology, 32 (1948) 3, S. 221–233.

Fobbe, E., Forensische Linguistik, Narr Francke Attempto, Tübingen, 2011.

Francillon, A., Rohatgi, P., Smart Card Research and Advanced Applications, Springer International Publishing, Basel, 2014.

Frohlich, H., Zell, A., Efficient parameter selection for support vector machines in classification and regression via model-based global optimization, in: Proceedings of the International Joint Conference on Neural Networks (IJCNN), (2005), S. 1431–1436.

Fröhlich, G., Plagiate und unethische Autorenschaften, in: Information - Wissenschaft & Praxis (2006) 57, S. 81–89.

Gasparyan, A.Y., Nurmashev, B., Seksenbayev, B., I. Trukhachev, V., I. Kostyukova, E., Kitas, G., Plagiarism in the Context of Education and Evolving Detection Strategies, in: Journal of Korean Medical Science, 32 (2017), S. 1220–1227.

Gavrilovski, A., Jimenez, H., Mavris, D.N., Rao, A.H., Shin, S., Hwang, I., Marais, K., Challenges and Opportunities in Flight Data Mining: A Review of the State of the Art, AIAA Infotech @ Aerospace, American Institute of Aeronautics and Astronautics, 2016.

Gavrilut, D., Cimpoesu, M., Anton, D., Ciortuz, L., Malware detection using machine learning, in: Proceedings of the International Multiconference on Computer Science and Information Technology, (2009), S. 735–741.

Gelbukh, A., Espinoza, F.C., Galicia-Haro, S.N., Molano, V., Cobos, C., Mendoza, M., Herrera-Viedma, E., Manic, M., Feature Selection Based on Sampling and C4.5 Algorithm to Improve the Quality of Text Classification Using Naïve Bayes, Springer International Publishing, 2014.

Gelbukh, A., Shrestha, P., Mukherjee, A., Solorio, T., Large Scale Authorship Attribution of Online Reviews, Springer International Publishing, 2018.

Ghahramani, Z., Unsupervised Learning, in: Bousquet, O., Luxburg, U. von, Rätsch, G. (Hrsg.), Advanced Lectures on Machine Learning, Springer Berlin Heidelberg, Berlin, Heidelberg, 2004, S. 72–112.

Gipp, B., Citation-based Plagiarism Detection, Springer Fachmedien Wiesbaden, Wiesbaden, s.l., 2014.

Grivas, A., Krithara, A., Giannakopoulos, G., Author Profiling using Stylometric and Structural Feature Groupings, in: Proceedings of the Conference on Multilingual and Multimodal Information Access Evaluation (CLEF), (2015).

Guillet, F.J., Hamilton, H.J., Quality Measures in Data Mining, Springer-Verlag Berlin Heidelberg, Berlin, Heidelberg, 2007.

Gunning, R., The Fog Index After Twenty Years, in: Journal of Business Communication, 6 (1969) 2, S. 3–13.

Gupta, P., Rao, S., Khushboo, S., Majumder, P., External & Intrinsic Plagiarism Detection: VSM & Discourse Markers based Approach, in: Notebook Papers of CLEF 2011 LABs and Workshops (2011).

H. Bast, C. Korzen, A Benchmark and Evaluation for Text Extraction from PDF, 2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL), 2017, S. 1–10.

Hadjidj, R., Debbabi, M., Lounis, H., Iqbal, F., Szporer, A., Benredjem, D., Towards an integrated e-mail forensic analysis framework, in: Digital Investigation: The International Journal of Digital Forensics & Incident Response, 5 (2009) 3-4, S. 124–137.

Han, J., Kamber, M., Pei, J., Data mining, 3. ed. Aufl., Elsevier/Morgan Kaufmann, Amsterdam, 2012.

Hariharan, S., Kamal, S., Faisal, A.V.M., Azharudheen, S.M., Raman, B., Detecting Plagiarism in Text Documents, in: Das, V.V., Vijayakumar, R., Debnath, N.C., Stephen, J., Meghanathan, N., Sankaranarayanan, S., Thankachan, P.M., Gaol, F.L., Thankachan, N. (Hrsg.), Information Processing and Management, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, S. 497–500.

He, H., Ma, Y., Imbalanced Learning: Foundations, Algorithms, and Applications, Wiley, 2013.

Heather, J., Turnitoff: identifying and fixing a hole in current plagiarism detection software, in: Assessment & Evaluation in Higher Education, 35 (2010) 6, S. 647–660.

Hoad, T.C., Zobel, J., Methods for identifying versioned and plagiarized documents, in: Journal of the American Society for Information Science and Technology, 54 (2003) 3, S. 203–215.

Holmes, I., The Evolution of Stylometry in Humanities Scholarship, in: Literary and Linguistic Computing, 13 (1998) 3, S. 111–117.

Hoorn, J., Frank, S., Kowalczyk, W., van der Ham, F., Neural network identification of poets using letter sequences, in: Literary and Linguistic Computing, 14 (1999) 3, S. 311–338.

Jain, R., Kasturi, R., Schunck, B.G., Machine vision, McGraw-Hill, New York, 1995.

Juola, P., Noecker, J., Stolerman, A., Ryan, M., Brennan, P., Greenstadt, R., Towards Active Linguistic Authentication, in: DigitalForensics 2013: Advances in Digital Forensics IX, 410 (2013), S. 385–398.

Kantardzic, M., Data mining, Wiley-Interscience IEEE Press; IEEE Xplore, Hoboken, New Jersey, Piscataway, Piscataway, New Jersey, 2009.

Kilgarriff, A., Reddy, S., Pomikálek, J., A Corpus Factory for Many Languages, in: Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10), (2010).

Kiss, T., Strunk, J., Unsupervised Multilingual Sentence Boundary Detection, in: Computational Linguistics, 32 (2006) 4, S. 485–525.

Köhler, R., Altmann, G., Piotrowski, R.G., Quantitative Linguistik, de Gruyter, Berlin, 2005.

Koppel, M., Schler, J., Argamon, S., Authorship attribution in the wild, in: Language Resources and Evaluation, 45 (2011) 1, S. 83–94.

Kraus, C., Plagiarism Detection - State-of-the-art systems (2016) and evaluation methods, Technische Universität Berlin, Berlin, 2016.

Kumar, E., Natural language processing, Reprint. Aufl., I K International Publishing House Pvt. Ltd, Neu-Delhi, 2012.

Kuznetsov, M., Motrenko, A., Kuznetsova, R., Strijov, V., Methods for Intrinsic Plagiarism Detection and Author Diarization, in: Proceedings of the Conference on Multilingual and Multimodal Information Access Evaluation (CLEF), (2016).

Lamas, D., Buitelaar, P., Hansen, N.D., Lioma, C., Larsen, B., Alstrup, S., Temporal Context for Authorship Attribution, Springer International Publishing, 2014.

Larose, D.T., Discovering knowledge in data, Second edition. Aufl., John Wiley & Sons, Hoboken, New Jersey, 2014.

Lipka, N., Stein, B., Shanahan, J.G., Estimating the Expected Effectiveness of Text Classification Solutions under Subclass Distribution Shifts, in: 2012 IEEE 12th International Conference on Data Mining, (2012), S. 972–977.

Liu, H., Motoda, H., Feature Extraction, Construction and Selection, Springer, Boston, MA, 1998.

Lorena, A.C., Carvalho, A.C.P.L.F. de, Gama, J.M.P., A review on the combination of binary classifiers in multiclass problems, in: Artificial Intelligence Review, 30 (2008) 1-4, S. 19–37.

Lu, Y., Automatic topic identification of health-related messages in online health community using text classification, in: SpringerPlus 2:309 (2013).

Luyckx, K., Kestemont, M., Daelemans, W., Intrinsic Plagiarism Detection Using Character Trigram Distance Scores, in: Notebook for PAN at CLEF 2011 (2011).

Ma, Y., Guo, G., Support Vector Machines Applications, Springer International Publishing, Cham, 2014.

Magerman, T., van Looy, B., Song, X., Data production methods for harmonised patent statistics, 2006 ed. Aufl., Office for Official Publications of the European Communities, Luxembourg, 2006.

Mahmoud, A., Niu, N., Xu, S., A semantic relatedness approach for traceability link recovery, in: Procceedings of the 20th IEEE International Conference on Program Comprehension (ICPC), (2012), S. 183–192.

Manar, H., Shameem, F., Attitude of Students Towards Cheating and Plagiarism: University Case Study, in: Journal of Applied Sciences, 14 (2014) 8, S. 748–757.

Manning, C.D., Raghavan, P., Schütze, H., Introduction to Information Retrieval, Cambridge University Press, 2008.

Maron, M.E., Kuhns, J.L., On Relevance, Probabilistic Indexing and Information Retrieval, in: Journal of the ACM, 7 (1960) 3, S. 216–244.

Maurer, H., Kappe, F., Zaka, B., Plagiarism - A Survey, in: Journal of Universal Computer Science, 12 (2006) 8, S. 1050–1084.

Mawson, C.O.S., Roget's Thesaurus of English Words and Phrases: Classified and Arranged So as to Facilitate the Expression of Ideas and Assist in Literary Composition, Thomas Y. Crowell Company, 1911.

Mccabe, D., Cheating among college and university students: A North American perspective, in: International Journal for Educational Integrity, 1 (2005), S. 1–11.

McDonald, A.W.E., Afroz, S., Caliskan, A., Stolerman, A., Greenstadt, R., Use Fewer Instances of the Letter "i": Toward Writing Style Anonymization, in: Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J.M., Mattern, F., Mitchell, J.C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M.Y., Weikum, G., Fischer-Hübner, S., Wright, M. (Hrsg.), Privacy Enhancing Technologies, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, S. 299–318.

Mikros, G., Argiri, E., Investigating Topic Influence in Authorship Attribution, in: Proceedings of the SIGIR 2007 International Workshop on Plagiarism Analysis (2007).

Mitkov, R., The Oxford handbook of computational linguistics, Reprinted. Aufl., Oxford Univ. Press, Oxford, 2009.

Mothe, J., Savoy, J., Kamps, J., Pinel-Sauvagnat, K., Jones, G., San Juan, E., Capellato, L., Ferro, N., Suchomel, Š., Brandejs, M., Determining Window Size from Plagiarism Corpus for Stylometric Features, Springer International Publishing, 2015.

Muhr, M., Zechner, M., Kern, R., External and Intrinsic Plagiarism Detection Using Vector Space Models, in: Proceedings of the SEPLN'09 Workshop on Uncovering Plagiarism, 502 (2009), S. 47–55.

Nanda, M., Seminar, K., Nandika, D., A Comparison Study of Kernel Functions in the Support Vector Machine and Its Application for Termite Detection, in: Information, 9 (2018).

Nédellec, C., Rouveirol, C., Lewis, D.D., Naive (Bayes) at forty: The independence assumption in information retrieval, Springer Berlin Heidelberg, 1998.

Niezgoda, S., Way, T.P., SNITCH, in: Baldwin, D., Tymann, P., Haller, S., Russell, I. (Hrsg.), Proceedings of the 37th SIGCSE technical symposium on Computer science education - SIGCSE '06, ACM Press, New York, New York, USA, 2006, S. 51–55.

o. V., Plagiarism Report, 2011, URL: http://turnitin.com/en_us/resources/blog/421-general/1660-plagiarism-report-infographic, gelesen am 16.07.2018.

o. V., Bereitstellen von Dokumenten in Repositorien, 2018a, URL: http://open-access.net/informationen-zu-open-access/rechtsfragen/bereitstellen-von-dokumenten-in-repositorien/, gelesen am 16.07.2018.

o. V., Grobid Documentation - Benchmarking, 2018b, URL: https://grobid.readthedocs.io/en/latest/Benchmarking/#fulltext-structures, gelesen am 16.07.2018.

o. V., Turnitin, 2018c, URL: http://turnitin.com/de/, gelesen am 16.07.2018.

o. V., Universitätsbibliothek Leipzig, 2018d, URL: https://www.ub.uni-leipzig.de, gelesen am 16.07.2018.

o. V., Was bedeutet Open Access?, 2018e, URL: http://open-access.net/informationen-zu-open-access/was-bedeutet-open-access/, gelesen am 16.07.2018.

o. V., Wikipedia Monolingual Corpora, 2018f, URL: http://linguatools.org/tools/corpora/wikipedia-monolingual-corpora/, gelesen am 16.07.2018.

Oberreuter, G., L'Huillier, G., Ríos, S., Velásquez, J., Approaches for Intrinsic and External Plagiarism Detection - Notebook for PAN at CLEF 2011, in: Proceedings of the Conference on Multilingual and Multimodal Information Access Evaluation (CLEF), (2011).

Osoba, O., Davis, P., An Artificial Intelligence/Machine Learning Perspective on Social Simulation: New Data and New Challenges, RAND Corporation, 2018.

Pak, A., Paroubek, P., Twitter as a Corpus for Sentiment Analysis and Opinion Mining, in: Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10), (2010), S. 1320–1326.

Paynter, G., Attribute-Relation File Format (ARFF), 2008, URL: https://www.cs.waikato.ac.nz/ml/weka/arff.html, gelesen am 16.07.2018.

Pereira, R., Cross Language Plagiarism Detection, Universidade Federal do Rio Grande do Sul, 2010.

Potthast, M., Overview of the 3rd International Competition on Plagiarism Detection, in: CEUR Workshop Proceedings, (2011).

Potthast, M., Stein, B., Barrón-Cedeño, A., Rosso, P., An evaluation framework for plagiarism detection, in: Proceedings of the 23rd International Conference on Computational Linguistics (2010).

Quinlan, J.R., C4.5: Programs for Machine Learning, Elsevier Science, 2014.

Raedt, L. de, Flach, P., Hoffmann, A., Kwok, R., Compton, P., Using Subclasses to Improve Classification Learning, Springer Berlin Heidelberg, 2001.

Ramnial, H., Panchoo, S., Pudaruth, S., Authorship Attribution Using Stylometry and Machine Learning Techniques, in: Berretti, S., Thampi, S.M., Srivastava, P.R. (Hrsg.), Intelligent Systems Technologies and Applications, 1st ed. 2016. Aufl., Springer International Publishing, Cham, 2016, S. 113–125.

Reese, R.M., Reese, J.L., Grigorev, A., Java - Data science made easy, Packt Publishing, Birmingham, UK, 2017.

Reginald D. Smith, Distinct word length frequencies: distributions and symbol entropies, in: Glottometrics, abs/1207.2334 (2012), S. 7–22.

Rindflesch, T.C., Fiszman, M., The interaction of domain knowledge and linguistic structure in natural language processing: interpreting hypernymic propositions in biomedical text, in: Journal of biomedical informatics, 36 (2003) 6, S. 462–477.

Roiger, R.J., Data Mining, 2nd ed. Aufl., CRC Press, Boca Raton, 2017.

Rudman, J., The State of Authorship Attribution Studies: Some Problems and Solutions, in: Computers and the Humanities, 31 (1997) 4, S. 351–365.

Russell, S.J., Norvig, P., Artificial intelligence, 3. Aufl., Pearson, Boston, Columbus, Indianapolis, New York, San Francisco, 2016.

Sandig, B., Textstilistik des Deutschen, 2., völlig neu bearb. und erw. Aufl. Aufl., de Gruyter, Berlin, 2006.

Schiller, A., Teufel, S., Stöckert, C., Thielen, C., Guidelines für das Tagging deutscher Textcorpora mit STTS, Unitersität Stuttgart - Institut für maschinelle Sprachverarbeitung, 1999.

Shearer, C., The CRISP-DM Model: The New Blueprint for Data Mining, in: Journal of Data Warehousing, 5 (2000) 4, S. 13–22.

Simpson, E.H., Measurement of Diversity, in: Nature, 163 (1949) 4148, S. 688.

Sraka, D., Kaucic, B., Source code plagiarism, in: Proceedings of the ITI 31st International Conference on Information Technology Interfaces, (2009), S. 461–466.

Stamatatos, E., A survey of modern authorship attribution methods, in: Journal of the American Society for Information Science and Technology, 60 (2008) 3, S. 538–556.

Stamatatos, E., Fakotakis, N., Kokkinakis, G., Computer-Based Authorship Attribution Without Lexical Measures, in: Computers and the Humanities, 35 (2001) 2, S. 193–214.

Stein, B., Eissen, S.M.z., Potthast, M., Strategies for retrieving plagiarized documents, in: Kraaij, W., Vries, A.P. de, Clarke, C.L.A., Fuhr, N., Kando, N. (Hrsg.), Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '07, ACM Press, New York, New York, USA, 2007, S. 825–826.

Stein, B., Lipka, N., Prettenhofer, P., Intrinsic plagiarism analysis, in: Language Resources and Evaluation, 45 (2011) 1, S. 63–82.

Stolerman, A., Authorship Attribution Using Writeprints, Drexel University, Philadelphia, 2012.

Stolerman, A., Authorship Verification, Drexel University, Philadelphia, 2015.

Suárez, P., Gonzalez-Cristobal, J., Villena-Román, J., A plagiarism detector for intrinsic plagiarism Lab Report for PAN at CLEF 2010, in: CLEF 2010 LABs and Workshops, Notebook Papers, 1176 (2010), S. 22–23.

Taghva, K., Vergara, J., Feature Selection for Document Type Classification, in: Proceedings of the fifth International Conference on Information Technology: New Generations, (2008), S. 179–182.

Tan, P.-N., Steinbach, M., Kumar, V., Introduction to data mining, Pearson Addison-Wesley, Boston, 2009.

Tiwari, M., Dixit, R., Kesharwani, A., Data Mining Principles, Process Model and Applications, eBooks2go Incorporated, 2017.

Tschuggnall, M., Intrinsic plagiarism detection and author analysis by utilizing grammar, Computer Science and Physics of the University of Innsbruck, 2014.

Tuldava, J., Stylistics, author identification, in: Köhler, R., Altmann, G., Piotrowski, R.G. (Hrsg.), Quantitative Linguistik, de Gruyter, Berlin, 2005, S. 368–387.

Uzuner, Ö., Katz, B., Nahnsen, T., Using syntactic information to identify plagiarism, in: Proc. 2nd Workshop on Building Educational Applications using NLP (2005), S. 37–44.

Wang, H., Huang, Y., Bondec-A Sentence Boundary Detector, in: CS224N Project (2003).

Wang, L., Support Vector Machines: Theory and Applications, Springer-Verlag GmbH, Berlin Heidelberg, 2005.

Wang, P., Lin, C., Support Vector Machines, in: Aggarwal, C.C. (Hrsg.), Data classification, CRC Press, Boca Raton, FL, 2015, S. 187–204.

Weber-Wulff, D., Test Cases for Plagiarism Detection Software, in: Proceedings of the 4th International Plagiarism Conference (2010).

Witten, I.H., Pal, C.J., Frank, E., Hall, M.A., Data mining, 4. Aufl., Morgan Kaufmann, Cambridge, MA, 2017a.

Witten, I.H., Pal, C.J., Frank, E., Hall, M.A., Data mining - Online Appendix (The WEKA Workbench), 4. Aufl., Morgan Kaufmann, Cambridge, MA, 2017b.

Wu, X., Kumar, V., Ross Quinlan, J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G.J., Ng, A., Liu, B., Yu, P.S., Zhou, Z.-H., Steinbach, M., Hand, D.J., Steinberg, D., Top 10 algorithms in data mining, in: Knowledge and Information Systems, 14 (2008) 1, S. 1–37.

Yangqiu Song, Dan Roth, Machine Learning with World Knowledge: The Position and Survey, in: CoRR, abs/1705.02908 (2017).

Yule, G.U., On sentence length as a statistical characteristic of style in prose: With application to two cases of disputed authorship, in: Biometrika (1938) 30, S. 363–390.

Zheng, R., Li, J., Chen, H., Huang, Z., A framework for authorship identification of online messages: Writing-style features and classification techniques, in: Journal of the American Society for Information Science and Technology, 57 (2006) 3, S. 378–393.

Ehrenwörtliche Erklärung

Ich versichere, dass ich die Masterarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Darüber hinaus versichere ich, dass die elektronische Version der Masterarbeit mit der gedruckten Version übereinstimmt.

Leipzig, den 24.07.2018

Ort, Datum Unterschrift