FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Intrinsisches Plagiatserkennungsverfahren auf Basis einer stilometrischen Clusteranalyse

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur/in

im Rahmen des Studiums

Medieninformatik

eingereicht von

Daniel Schneider
Matrikelnummer 0525640

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuer/in: Ao. Univ.-Prof. Mag. Dr. Horst Eidenberger

Wien, TT.MM.JJJJ

(Unterschrift Verfasser/in)

(Unterschrift Betreuer/in)

Daniel Schneider
Säulengasse 7/11
1090 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum) (Unterschrift)

Abstract

Plagiarism detection is the process of analysing a scientific text and to find potential plagiarised passages. In this context, non-automated procedures have proven to be time-consuming and subjective. Especially in the light of a steadily increasing number of scientific publications, automated software-aided approaches represent valuable instruments to effectively detect plagiarized text. Conventional plagiarism software compares text passages against potential original documents based on matching strings. In contrast, intrinsic plagiarism detection attempts to detect plagiarized sections based on stylometric features. Thus, this procedure enables to discover sudden changes in the writing style. The recognition of stylistic inconsistencies is closely associated with the field of Authorship Attribution, especially in the use of textual features.

The present thesis focuses on the development and implementation of a prototype of intrinsic plagiarism detection. The developed approach automatically extracts stylometric features from a given text and performs a multivariate cluster analysis. The respective clusters represent groups of text passages exhibiting similar stilometric properties and can therefore be associated with the respective number of authors. The input data (text) is represented by articles from the English-language edition of the online encyclopedia Wikipedia.

The evaluation results demonstrate that the conducted procedure enables to approximately distinguish between text passages originating form different authors. Furthermore, it was shown that the reliability of the results are strongly dependent on the number of authors. The approximation of the correct author class structure depends among others on the determination of the number of clusters. The resulting number is validated by an own developed quality measure.

Kurzfassung

Plagiatserkennung ist der Prozess, einen wissenschaftlichen Text zu analysieren und mögliche plagiierte Abschnitte zu finden. In diesem Zusammenhang haben sich nicht-automatisierte Vorgehensweisen als zeitaufwändig und subjektiv erwiesen. Insbesondere in Anbetracht des steigenden Publikationsvolumens wissenschaftlicher Arbeiten stellen automatisierte Verfahren wertvolle Instrumente zur Verfügung, die das Erkennen von plagiiertem Text effektiv unterstützen. Konventionelle Plagiatserkennungssoftware vergleicht Textpassagen mit möglichen Originaldokumenten basierend auf übereinstimmende Zeichenketten. Im Gegensatz dazu versucht intrinsische Plagiatserkennung plagiierte Abschnitte anhand stilistischer Merkmale zu erkennen. Auf diese Weise ist es möglich, plötzliche Änderung des Schreibstils zu erkennen. Die Erkennung von stilistischen Inkonsistenzen ist mit dem Gebiet der Authorship Attribution, vor allem in der Verwendung der textuellen Features, eng verbunden.

Die vorliegende Arbeit beschäftigt sich mit der Entwicklung und Implementierung eines Prototypen, der eine intrinsische Plagiatserkennung durchführt. Das entwickelte Verfahren extrahiert automatisch stilometrische Features aus einem Text und führt eine multivariate Clusteranalyse durch. Die jeweiligen Cluster repräsentieren Gruppen von Textpassagen, die ähnliche stilometrische Eigenschaften aufweisen und können daher mit der entsprechenden Anzahl von Autoren in Verbindung gesetzt werden. Die Eingabedaten (Text) werden durch Artikel aus der englischsprachigen Ausgabe des Onlinelexikons Wikipedia generiert.

Die Evaluierung der Ergebnisse zeigt, dass das durchgeführte Verfahren Textpassagen verschiedener Autoren unterscheiden kann. Des Weiteren wird gezeigt, dass die Zuverlässigkeit des Verfahrens stark von der Anzahl der Autoren abhängt. Die Annäherung der korrekten Klassenaufteilung hängt unter anderem von der Ermittlung der Clusteranzahl ab. Die resultierende Anzahl wird anhand eines eigens entwickelten Qualitätsmaßes bewertet.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Das Gebiet der automatisierten Plagiatserkennung wissenschaftlicher Arbeiten ist eine noch relativ junge Disziplin, die besonders in den letzten Jahren an Bedeutung gewonnen hat. Mit der wachsenden Anzahl an Akademikern steigt auch das Publikationsvolumen – etwa alle 20 Jahre verdoppelt sich die jährlich publizierte Menge wissenschaftlicher Arbeiten (vgl. Bauerlein et al. 2010). Aufgrund der Informationsvielfalt, die durch das Internet geboten wird, ist ein breiter Zugriff auf verschiedenste Quellen möglich, was das Plagiieren so attraktiv wie nie zuvor erscheinen lässt (vgl. Alzahrani et al. 2012). Dementsprechend stellen Errami & Garner (2008) fest, dass die Zahl der Plagiate in den letzten Jahren schneller gestiegen ist als die Zahl der Publikationen.

Im Rahmen einer Studie an der Universität Graz, die sich auf eine 2012 durchgeführte anonyme Umfrage von über 600 Studierenden bezieht, gab ca. ein Drittel der Befragten zu, bereits mindestens einmal plagiiert zu haben. Das Bewusstsein des Vergehens und der Plagiatsproblematik an sich ist jedoch grundsätzlich da (vgl. Reichmann 2013).

Nicht nur Studierende, sondern auch sogenannte *Personen öffentlichen Interesses*, geraten in Bedrängnis und müssen sich Plagiatsvorwürfen stellen. So traten in den letzten Jahren innerhalb kurzer Zeit zwei ranghohe Politiker Deutschlands – Karl-Theodor zu Guttenberg 2011 und Annette Schavan 2013 – aufgrund nachweislicher Plagiate in der Dissertationsarbeit von ihren politischen Ämtern zurück. Außerdem wurde ihnen ihr Abschlusstitel aberkannt.

Konsequenzen haben solche Aufdeckungen von Plagiaten nicht nur für die Autoren der Arbeit selbst, sondern im Grunde für all jene, die das Plagiat als solches erkennen hätten sollen. Der Ruf der betreffenden Hochschuleinrichtung, des Verlegers, des Journals etc. kann darunter leiden (vgl. IThenticate 2013). Inzwischen ist es auf sämtlichen Hochschulen und in Verlagen üblich, eine verpflichtende eidesstattliche Erklärung beizulegen, die bestätigt, dass die Arbeit selbstständig verfasst worden ist und die verwendeten Quellen vollständig angegeben sind (vgl. Reichmann 2013).

Auch der Einsatz von Plagiatserkennungssoftware ist zur gängigen Praxis geworden. Bevor Abschlussarbeiten an Hochschulen angenommen werden oder ein Verlag die Publikation von Werken freigibt, müssen sie einer Plagiatsprüfung standhalten. Die Funktionstüchtigkeit der derzeit gebräuchlichen Erkennungstools wird jedoch kritisch gesehen. Jones (2009) spricht von einem *changing face of plagiarism*: Die Art des Plagiierens hat sich von einem simplen Copy-and-Paste zu einer aufwändigen Umschreibung und Neuformulierung von Textpassagen fremder Arbeiten gewandelt.

Da klassische Plagiatserkennungssysteme vor allem mit Zeichenkettenvergleichen arbeiten, kann ihre Performance durch solche Umschreibungen beeinträchtigt werden (vgl. Chong et al. 2010). Ein neuer Ansatz, der dem intuitiven Prüfen eines Menschen viel eher entspricht, wird durch die Methodik der Stilometrie geboten. Einem geübten Leser fallen plötzliche Stilbrüche innerhalb eines Textes auf. Textsegmente, die von unterschiedlichen Schreibstilen geprägt sind, lassen auf unterschiedliche Autoren schließen. Dennoch müssen weitere Recherchen der verdächtigen Stellen angestrengt werden, um ein tatsächliches Plagiat nachzuweisen (vgl. Alzahrani et al. 2012).

Die Stilometrie soll das Messen des Schreibstils ermöglichen. Man versucht den Stil eines Autors in Zahlen zu fassen und Unterschiede zwischen Autoren zu erkennen. Auf diese Weise ist es auch möglich, dem Computer beizubringen, wie ein Schreibstil definiert bzw. durch welche Attribute er charakterisiert ist.

1.1 Zielsetzung

Es wird angenommen, dass Texte unterschiedlicher Autoren anhand ihres Schreibstils unterschieden werden können und dass der Schreibstil mittels einer Kombination aus verschiedenen textuellen Eigenschaften automatisiert erfasst werden kann.

Darüber hinaus wird die Annahme verfolgt, dass aus den extrahierten Eigenschaften durch den Einsatz von Clusterverfahren[1] bestimmt werden kann, von wievielen Autoren ein Text stammt und welche konkreten Stellen zu einer theoretischen Identität zusammengefasst werden können.

Es soll ein Prototyp entwickelt werden, der diese Form der intrinsischen Plagiatserkennung ermöglicht. Die Applikation soll sich vorerst auf die englische Sprache beschränken, da einige linguistische Maße nicht auf mehrere Sprachen anwendbar sind und die Qualität und Verfügbarkeit dafür notwendiger Werkzeuge für die englische Sprache höher sind. Als Testdaten bieten sich Artikel aus der englischsprachigen Ausgabe des Onlinelexikons *Wikipedia* an, die den realen Anwendungsfall gut nachbilden, da die Texte jeweils von verschiedenen Autoren stammen können, das Thema aber unverändert bleibt. Um eine erleichterte Interaktion und eine Visualisierung des Outputs zu gewährleisten, soll die Applikation außerdem eine simple grafische Oberfläche bieten.

[1] Clusterverfahren sind Methoden unüberwachten maschinellen Lernens und werden in Abschnitt 2.3 und Umsetzungen davon in Abschnitt 3.2 beschrieben.

1.2 Struktur der Arbeit

Diese Arbeit ist wie folgt strukturiert: Kapitel 2 beleuchtet den theoretischen Hintergrund mit historischen Überblicken und grundlegenden Vorgehensweisen der Plagiatserkennung und der Stilometrie, einem Einblick in die Thematik des maschinellen Lernens sowie einer Betrachtung verwandter Arbeiten. Kapitel 3 verfolgt einen praktischeren Ansatz und beschreibt in dieser Arbeit verwendete Methoden – von textuellen Features und deren Extraktion, über spezielle Clusterverfahren, hin zu Evaluierungsmethoden überwachter und unüberwachter Verfahren. In Kapitel 4 werden das Design und die Umsetzung der Applikation diskutiert. Die Beschreibung bestimmter Implementierungsschritte folgt einer allgemeinen Übersicht der Architektur und dem generellen Ablauf. Kapitel 5 präsentiert die Ergebnisse, die verwendete Evaluierungsmethodik und eine Diskussion der Ergebnisse. Abschließend folgt in Kapitel 6 eine Zusammenfassung der Arbeit und ein Ausblick auf mögliche aussichtsreiche Verfahren.[2]

[2] Der vorliegende Text ist auf Basis des LaTeX-Templates von Gockel (2008) erstellt.

Kapitel 2

Theoretischer Hintergrund

Dieses Kapitel soll als Einführung in die Thematik der stilometrischen Plagiatserkennung dienen und einen Einblick in die theoretischen Konzepte dahinter bieten. Der erste Unterabschnitt beschäftigt sich mit der Plagiatserkennung an sich. Aufbauend darauf werden die Grundlagen der Stilometrie untersucht. Der anschließende Abschnitt soll die prinzipielle Funktionsweise maschinellen Lernens beschreiben und die möglichen Verbindungen zur Anwendung in der Stilometrie aufzeigen, bevor abschließend ein Überblick über verwandte Arbeiten geboten wird.

2.1 Plagiatserkennung

Das computergestützte Erkennen von Plagiaten ist sehr eng mit dem Gebiet der Information Retrieval (IR)[1] verbunden. Dieser Abschnitt liefert einen kurzen Überblick über die geschichtliche Entwicklung und Funktionsweise aktuell verwendeter Systeme.

2.1.1 Geschichte & Klassische Vorgehensweise

Der Begriff des Plagiats geht auf die lateinische Bezeichnung *plagiārius* zurück, was so viel wie Entführer, Kidnapper heißt. Der Ausdruck wurde im 1. Jhd. n. Chr. vom Dichter Martial geprägt. Er war der Überzeugung, dass ein Dichterkollege seine Verse als dessen eigene wiedergegeben hatte und beschimpfte ihn in einem seiner Epigramme als plagiārius (vgl. Seo 2009).

Definitionen des Begriffes variieren in ihrem Fokus, unterschiedliche Aspekte des Vergehens hervorzuheben: Diebstahl geistigen Eigentums, Betrug und/oder Copyright-Verletzungen.

[1] IR ist ein Gebiet, das sich mit der Suche unstrukturierter Inhalte (häufig Textdokumente) auseinandersetzt (vgl. Manning et al. 2008, S. 1).

Fishman 2009, S. 5 versucht allen drei Aspekten gerecht zu werden und definiert den Begriff wie folgt:

> *Plagiarism occurs when someone*
>
> 1. *Uses words, ideas, or work products*
> 2. *Attributable to another identifiable person or source*
> 3. *Without attributing the work to the source from which it was obtained*
> 4. *In a situation in which there is a legitimate expectation of original authorship*
> 5. *In order to obtain some benefit, credit, or gain which need not be monetary*

Im Zusammenhang wissenschaftlicher Arbeiten geht es also darum, dass jemand Wörter, Abschnitte oder Ideen einer anderen Arbeit verwendet und sie als die eigenen präsentiert, ohne auf die ursprüngliche Quelle zu verweisen. Den Begriff der Plagiatserkennung definieren Alzahrani et al. (2012) als den Prozess, eine Arbeit zu analysieren, mögliche plagiierte Abschnitte zu kennzeichnen und ähnliche Quellen (von denen das Plagiat stammen könnte) zu liefern, sofern diese vorhanden sind. Nachdem dieser Prozess mit erheblichem Aufwand verbunden sein kann und sich Methoden des IR für das Finden identischer Stellen zweier Texte anbieten, scheint die Unterstützung automatisierter Software sinnvoll zu sein.

Obwohl computergestützte Plagiatserkennung natürlicher Sprache bereits in den 1990er Jahren begonnen hat (vgl. Alzahrani et al. 2012), wurde die erste Plagiatserkennungssoftware im Hochschulbereich erst 2001 an der University of Virginia eingesetzt. Dabei zeigte sich, dass die damals verfügbare Software noch nicht ausgereift war, Plagiate effektiv zu erkennen (vgl. Stappenbelt & Rowles 2009). In den folgenden Jahren wurde die Entwicklung kommerziell verfügbarer Software vorangetrieben und es entstanden Produkte wie Turnitin[2], iThenticate[3] oder Ephorus[4], die noch heute breite Anwendung finden. Die konkret verwendeten Algorithmen hinter dieser Software sind nicht bekannt.

> *iThenticate utilizes a proprietary algorithm that transforms each submitted manuscript into a "digital fingerprint", which is compared – just like a human fingerprint – to an extensive database where subtle shades of similarity can be detected.* (IThenticate 2013, S. 2)

Damit wird angedeutet, dass ein stilometrischer Ähnlichkeitsvergleich zum Einsatz kommt. Auf welche Art oder in welchem Ausmaß dies geschieht kann jedoch nicht genauer untersucht werden. Die generelle Funktionsweise korpusbasierter Systeme wird in Abbildung 2.1 dargestellt: Die zu prüfende Arbeit wird mit Arbeiten aus einem verfügbaren Korpus verglichen und auf Ähnlichkeiten untersucht. Jegliche Treffer werden

[2] `http://www.turnitin.com`, abgerufen am 25.08.2014
[3] `http://www.ithenticate.com`, abgerufen am 25.08.2014
[4] `http://www.ephorus.com`, abgerufen am 25.08.2014

anschließend als Output mit der entsprechenden Quelle zurückgeliefert. Zusätzliche Schritte wie die Vorverarbeitung der zu prüfenden Arbeit oder das Zusammenführen zusammenhängender Treffer sind notwendig. Im Mittelpunkt steht jedoch die Datenbank an Dokumenten, mit denen der Text verglichen wird.

Alzahrani et al. (2012) unterscheiden zwischen extrinsischer und intrinsischer Plagiatserkennungssoftware. Das beschriebene Verfahren fällt in die Kategorie der extrinsischen Plagiatserkennung. Extrinsische Systeme beruhen darauf, externe Dokumente zu untersuchen, die mit der verdächtigen Arbeit verglichen werden. Intrinsische Systeme beschränken sich hingegen auf die zu prüfende Arbeit selbst und verwenden keine Information von außen. Erstere suchen Übereinstimmungen zwischen Dokumenten, letztere suchen Stilunterschiede innerhalb eines Dokumentes.

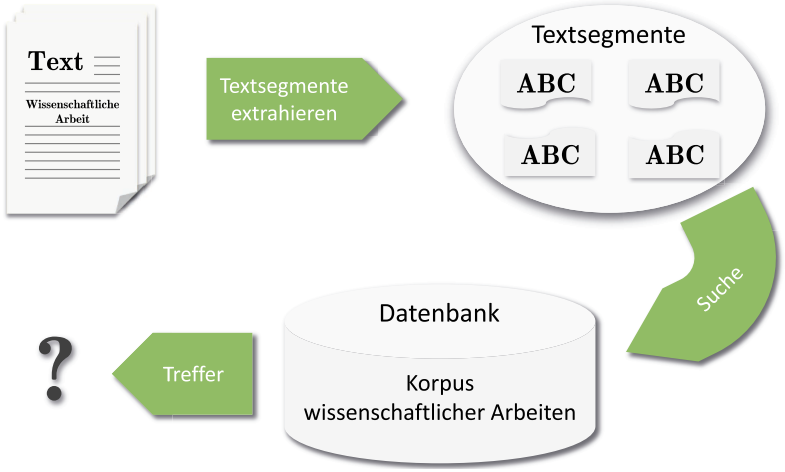


Abbildung 2.1: Ablauf korpusbasierter Plagiatserkennung.

2.1.2 Korpusbasiert vs. Stilometrie

In der Literatur werden verschiedene Formen des Plagiierens unterschieden. Dabei wird die verwendete Technik und der Grad der Verschleierung in Betracht gezogen: nahezu oder direktes Kopieren, Einfügen zusätzlicher Leerzeichen oder Buchstaben in weißer Schrift zwischen Buchstaben und Wörtern, Satzumstellung, komplette Neuformulierung, Zusammenfassung oder Zerstückelung mehrere Sätze, Rückübersetzung etc. (vgl. Alzahrani et al. 2012; IThenticate 2013; Jones 2009).

Extrinsische Systeme haben den offensichtlichen Vorteil für die potentiell plagiierten Stellen bereits einen Beweis – in Form der gefundenen Quelle – zu liefern. Da sie sich jedoch stark auf direkt kopierte Stellen im Text konzentrieren, sind sie nicht in der Lage, um- oder neugeschriebene Passagen effektiv zu erkennen (vgl. Alzahrani et al. 2012; Chong et al. 2010).

Intrinsische Systeme beziehen sich hingegen nicht direkt auf identische Wortfolgen, sondern versuchen Stilunterschiede zu erkennen. Stilometrische Verfahren nutzen Techniken des Natural Language Processing (NLP), welche grundlegend auch dazu verwendet werden könnten, die Einschränkungen rein korpusbasierter Verfahren zu umgehen.

In Lyon et al. (2004) wird die Frequenz von einzigartigen Wort-Tri-Grammen[5] gemessen. Es hat sich gezeigt, dass eine gewisse Wiederholungsrate der Tri-Gramme – selbst in Texten, die nur von einem Autor stammen – nicht überschritten wird. Werden zwei Arbeiten verglichen, deren Tri-Gramme über eine festgelegte Wiederholungsrate hinausgehen, liegt die Vermutung eines Plagiats nahe.

Chong et al. (2010) fassen diverse NLP-Techniken zusammen, welche dazu dienen könnten, korpusbasierte Systeme zu erweitern: die Entfernung von Satzzeichen, Stemming (das Reduzieren von Wörtern auf ihre lexikalische Grundform), das Filtern von Stoppwörtern, Part-of-Speech-Tagging[6] (POS) und eine Thesauruseinbindung. Durch die Verwendung dieser Methoden könnten Satzumstellungen oder ausgetauschte Synonyme sowie Übereinstimmungen in der grammatikalischen Struktur entdeckt werden. Es ist davon auszugehen, dass einige dieser Methoden in kommerziell vertriebener Plagiatserkennungssoftware bereits zur Anwendung kommen.

2.2 Stilometrie

Stilometrie versucht stilistische Aspekte eines Werkes zu messen. Es wird nicht nach einer offensichtlichen Übereinstimmung gesucht, sondern vielmehr versucht, eine übergreifende Konsistenz zu finden:

> *Questions of style are questions of consistency, rather than equality. "Is this one like the others?"asks the connoisseur, not "Is this one the same as any of the others?"* (Rockmore 2006)

2.2.1 Historisch

Erste Versuche einer mathematischen Analyse von Literatur gehen auf das 19. Jhd. zurück. 1851 strebte der britische Mathematiker Augustus de Morgan eine Charaktersierung des Schreibstils durch die Verwendung der durchschnittlichen Wortlänge an. 1887 sprach der amerikanische Pysiker Thomas C. Mendenhall in seinem Werk „The characteristic curves of composition“ (Mendenhall 1887) von einer charakteristischen Komposition eines Autors, die sich aufgrund der verwendeten Wortlängen bestimmen lässt. Er wollte die Werke der Autoren Charles Dickens und William Thackeray stilistisch unterscheiden.

[5] Wort-Tri-Gramme stellen Folgen von drei Wörtern dar. Die Wort-Tri-Gramme des nachfolgenden Satzes im Text sind: (Es, hat, sich), (hat, sich, gezeigt), (sich, gezeigt, dass) etc. Textuelle Features, darunter auch N-Gramme, werden in Abschnitt 3.1.1 erläutert.

[6] Kann als Wortartmarkierung übersetzt werden – wird in Abschnitt 3.1.2 näher erläutert.

Erfolgreicher war der Versuch des polnischen Philosophen Wincenty Lutoslawski etwa zehn Jahre später. Mit zusätzlichen numerischen Attributen (ca. 500) versuchte er Platos Dialoge chronologisch einzuordnen. In seinem Buch „Principes de stylometrie“ (1890) beschrieb er seinen Ansatz und führte damit den Begriff der Stilometrie ein (vgl. Lakshmi & Pateriya 2012; Rockmore 2006).

Auch in der russischen Literaturwissenschaft gab es intensive Untersuchungen auf dem Gebiet der Stilometrie. So versuchte man unter anderem Charakteristika von Autoren wie Puškin und Tolstoj statistisch zu erfassen (vgl. Grzybek & Kelih 2005).

Erste bedeutsame Erfolge konnten Mosteller & Wallace (1963) mit der Autorenzuordnung der umstrittenen „Federalist Papers“ verzeichnen. Im Gegensatz zu den bisherigen Versuchen beobachteten sie die Frequenz sehr häufig verwendeter Wörter der englischen Sprache, welche unabhängig vom Inhalt waren und eine stilistische Unterscheidung zwischen den drei Autoren Hamilton, Madison und Jay ermöglichten.

In den darauffolgenden Jahren (bis in die späten 1990er) wurden verschiedenste textuelle Eigenschaften definiert. Darunter fielen Maße, die sich auf die Satzlänge, Wortlänge, Wortfrequenzen, Zeichenfrequenzen und diverse Wortschatzmaße bezogen (vgl. Stamatatos 2009). Rudman (1998) kritisierte die vorgestellten Methoden und Ergebnisse in ihrer Gesamtheit. Er bemängelte, dass es an objektiver Evaluierung fehle und die Daten, mit denen gearbeitet wird, wären mangelhaft – es fehle an interdisziplinärer Expertise.

Ein Beispiel dafür ist die QSUM-Methode von Morton: QSUM ermittelt aufsummierte Frequenzen von Wörtern und Satzlängen. Anfang der 1990er Jahre fand die Methode große Akzeptanz. In Großbritannien und Australien wurde sie vor Gericht eingesetzt. Oft verwendete der verteidigende Anwalt die Methode, um festzustellen, ob das vom Angeklagten gelieferte Geständnis tatsächlich sein eigenes war oder von den Behörden manipuliert wurde. Das Verfahren wurde daraufhin stark kritisiert und die Funktionstüchtigkeit konnte mehrmals widerlegt werden (vgl. Tweedie 2005, S. 393f.).

Die Idee den Stil eines Werkes mathematisch zu analysieren fand nicht nur in der Literaturwissenschaft Zuspruch, sondern auch in den Gebieten der Musik und der bildenden Kunst. So setzte sich der Physiker Wilhelm Fucks gleichzeitig mit der Stilometrie in der Literaturwissenschaft und in der Musik auseinander (vgl. Fucks 1962). Er untersuchte Häufigkeitsverteilungen von Tonhöhen und Intervallen sowie Übergangswahrscheinlichkeiten benachbarter Töne und wie stark diese sich gegenseitig beeinflussen. Mit seinen Stilanalysen gelang es ihm, Musikstile verschiedener Epochen zu unterscheiden (vgl. Aichele 2005, S. 155ff.). Aktuellere Arbeiten auf dem Gebiet sind jene von Juola (2004) oder Stamatatos & Widmer (2002).

Der italienische Kunsthistoriker Giovanni Morelli beschäftigte sich bereits im 19. Jhd. mit der Stilanalyse in der bildenden Kunst. Er sprach von sogenannten unterbewussten *Grundformen*, die jeden Maler kennzeichnen. Durch die Untersuchung scheinbar unwichtiger Details (Abbildungen von Fingernägeln, Ohrläppchen oder Vorhängen im Hintergrund) schaffte er es mehrere Fälschungen und Falschzuordnungen aufzudecken (vgl. Rockmore 2006).

Der Hauptgegenstand der Stilometrie ist (bis dato) jedoch die Linguistik bzw. die Schriftsprache. Die Anwendung in der Musik und der bildenden Kunst wurden der Vollständigkeit halber angeführt und sollten dem allgemeinen Verständnis der Stilanalyse dienen.

Entwicklungen auf den Gebieten Information Retrieval, Machine Learning und Natural Language Processing ermöglichten es, größere Datenmengen zu verarbeiten, mit den gesammelten Daten effizienter umzugehen und Features effektiver zu extrahieren. Dadurch wurden die Möglichkeiten stilometrische Techniken zu nutzen erweitert und die Vorgehensweise verändert (vgl. Stamatatos 2009).

2.2.2 Vorgehensweise

Wie aus dem letzten Abschnitt hervorgeht, werden bestimmte Maße für die Analyse des Stils verwendet. Diese Maße sollen von nun an, wie in der Literatur üblich, als Features bezeichnet werden. Die grundlegende Vorgehensweise besteht darin, Features des zu untersuchenden Dokumentes zu extrahieren und mit diesen extrahierten Features weitere Berechnungen durchzuführen. Das tatsächliche Dokument wird also in numerisch darstellbare Werte transformiert und kann auf diese Weise maschinell weiterverarbeitet werden (siehe Abbildung 2.2).

> *There are no two humans, no matter what languages they use and howsimilar thoughts they have, [who] write exactly the same text.* (Alzahrani et al. 2012, S. 133)

Stilometrische Systeme teilen die Annahme, dass keine zwei Menschen denselben Text schreiben und dass der Text von verschiedenen Autoren – zu einem gewissen Grad – auch unterschiedlich ist (vgl. Alzahrani et al. 2012). In der Literatur bedient man sich deshalb oft auch der Analogie des stilistischen Fingerabdruckes eines Autors. Schulstad et al. (2012) kritisieren diese Ansichtsweise und argumentieren mit der weit höheren Präzision biometrischer Systeme.

Generell hängt die Vorgehensweise von der konkreten Aufgabenstellung ab. Es sind verschiedene Arten von Aufgaben denkbar, die mit Hilfe stilometrischer Mittel gelöst werden könnten (vgl. Stamatatos 2009):

Authorship Attribution / Author Identification Input: Dokumente mit jeweils bekanntem Autor und ein oder mehrere Dokumente mit fraglicher Autorenschaft. Output: Dokument(e) mit fraglicher Autorenschaft werden einem Autor aus einem Pool bekannter Autoren zugeordnet.

Author Verification Input: Ein Dokument mit fraglicher Autorenschaft soll anhand vorhandener Dokumente eines Autors verifiziert werden. Output: Stammt das Dokument von diesem Autor oder nicht?

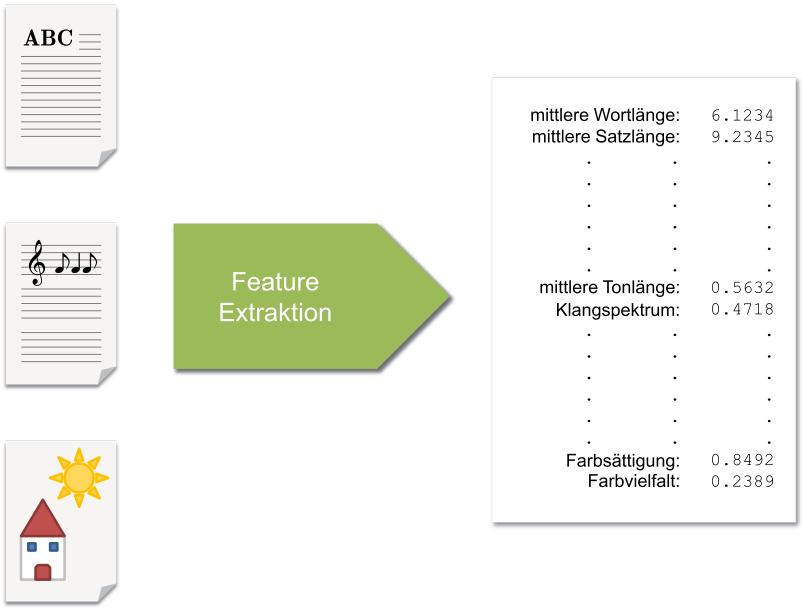


Abbildung 2.2: Stilometrische Features aus Dokumenten extrahieren.

Author Profiling Input: Ein Dokument, dessen Autor auf eine bestimmte Eigenschaft überprüft werden soll. Output: Eine Aussage über Eigenschaften des Autors (Geschlecht, Alter, psychischer Zustand etc.) wird getroffen. Als weiterer Input wird ein Ground Truth[7] benötigt, auf dessen Basis die jeweiligen Aussagen getroffen werden können.

Author Independent Characterization Input: Ein Dokument, das auf bestimmte Eigenschaften (vom Autor unabhängig) überprüft werden soll. Output: Eine Aussage über die Epoche in der das Dokument geschrieben wurde, welchem Genre es zugehörig ist, ob es eine Spam-Mail ist oder nicht etc. soll getroffen werden.

Plagiarism Detection (extrinsic) Input: Dokumente welche auf Übereinstimmungen überprüft werden. Output: Sind die Dokumente (oder Teile davon) identisch? Stilometrischer Einsatz: kopierte Stellen, die umgeschrieben worden sind, sollen trotzdem noch als Plagiat erkannt werden.

Plagiarism Detection (intrinsic) or Detection of Stylistic Incosistencies Input: Ein Dokument soll auf Ähnlichkeiten bzw. Inkonsistenzen innerhalb des Dokumentes (Vergleich von Textsegmenten) überprüft werden. Output: Stammt der Text von einem oder mehreren Autoren? Welche Stellen sind welchem Autor zuzuweisen?

[7] Ground Truth kann als Wissensbasis übersetzt werden (Details dazu siehe Abschnitt 2.3).

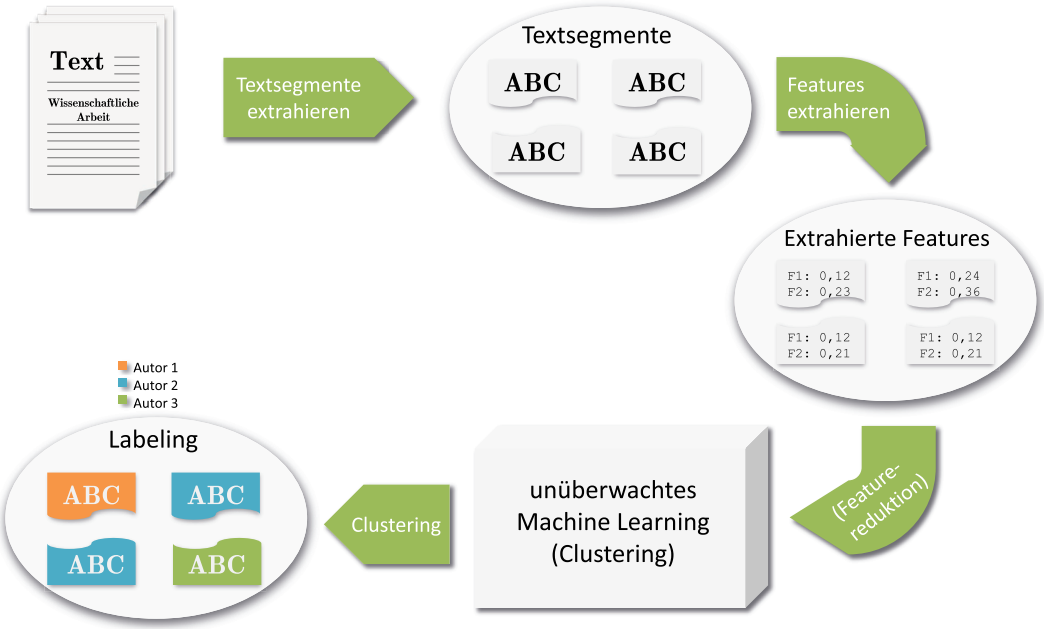


Abbildung 2.3: Möglicher Ablauf einer stilometrischen Plagiatserkennung.

Authorship Attribution und Author Verficiation vergleichen das in Frage stehende Dokument mit externen Dokumenten. Es geht darum, stilistische Übereinstimmungen zwischen Dokumenten zu finden.

Author Profiling und Author Independent Characterization bauen auf eine vorherige Wissensbasis auf, um Aussagen über bestimmte Eigenschaften des Autors treffen zu können. Verschiedenste Eigenschaften und Charakteristika wurden untersucht, die Ergebnisse auf diesem Gebiet sind deshalb durch starke Schwankungen gekennzeichnet (vgl. Juola 2007).

Plagiarism Detection (extrinsic) könnte stilometrische Mittel nutzen, um plagiierte Stellen zu finden, die neu geschrieben oder umformuliert wurden. Das Ziel besteht darin, Inhalte zu vergleichen und eventuelle Stilunterschiede zu eliminieren (z.B. verwendete Synonyme).

Plagiarism Detection (intrinsic) kann als Generalisierung von Authorship Attribution und Author Verification betrachtet werden und erfüllt dieselbe Aufgabe, mit der Einschränkung auf das zu prüfende Dokument selbst (vgl. Alzahrani et al. 2012). Eine mögliche Vorgehensweise für ein intrinsisches Plagiatserkennungssystem wird in Abbildung 2.3 beschrieben. Der offensichtliche Nachteil zu korpusbasierten Systemen besteht darin, dass für die Stellen, die sich stilistisch unterscheiden, kein Nachweis der tatsächlich verwendeten Quelle geliefert wird.

Betrachtet man die erläuterten Aufgabenstellungen, sind Anwendungen in verschiedenen Gebieten möglich.

2.2.3 Anwendungsgebiete

Durch die vorherrschende Anonymität im Internet scheinen Fragen über die dahinter liegenden Identitäten umso bedeutender zu werden. E-Mail-Nachrichten, Beiträge in Foren und Blog-Einträge haben das Anwendungsgebiet stilometrischer Verfahren in den letzten Jahren erweitert (vgl. Juola 2007).

Die folgende Auflistung fasst einige Anwendungsbeispiele zusammen (vgl. Stamatatos 2009):

Geheimdienstliche Tätigkeiten: Zuweisung und Verknüpfung von terroristischen Mitteilungen oder Ankündigungen zu bekannten Terroristen

Strafgesetz: Autorenidentifizierung von Drohbriefen, Authentizitätsprüfung von Abschiedsbriefen, Plagiatsprüfung

Zivilgesetz: Copyright-Verletzungen, Plagiatsprüfung

Computer-Forensik: Autorenidentifizierung schädlicher Software

Linguistik: kann die verwendete Sprache eine Aussage über Geschlecht, Alter, Herkunft, mentalen Zustand etc. des Autors treffen?

Literaturwissenschaft: Identifizierung/Zuweisung eines Autors von/zu umstrittenen Werken, Genre-Erkennung, Epochen-Zuweisung

Ist man sich bewusst, welche Daten zur Verfügung stehen und welches Ziel damit verfolgt werden soll, besteht der nächste Schritt darin, die Daten effizient aufzuarbeiten und die Stärke bestehender Machine-Learning-Techniken zu nutzen.

2.3 Machine Learning für Stilometrie

Machine Learning (ML) spielt in stilometrischen Systemen auf mehreren Ebenen eine zentrale Rolle. Wesentliche Fragen, die in diesem Abschnitt behandelt werden, sind: Was ist ML? Wie kann aus Daten gelernt werden? In welcher Form ist ML für stilometrische Systeme interessant?

2.3.1 Grundlagen

Machine Learning kann als Teildisziplin der künstlichen Intelligenz verstanden werden und umfasst eine Reihe von Methoden, die ein automatisches Erkennen von Mustern in Daten ermöglichen. Aus diesen Mustern sind Schlussfolgerungen über zukünftige Daten möglich (vgl. Murphy 2012, S. 1).

Vor der maschinellen Verarbeitung der Daten musste das Messen eines Features (z.B. Wortlänge) und das anschließende Prüfen, ob das Feature eine Diskriminierung der Daten

zulässt, händisch durchgeführt werden. Eine maschinelle Verarbeitung scheint schon allein wegen der ansonsten schwer vermeidbaren hohen Fehlerrate sinnvoll zu sein.

Die eigentliche Stärke, die ML bietet, äußert sich jedoch im Lernen und Erkennen von Mustern hochdimensionaler Strukturen, welche für den Menschen nur theoretisch vorstellbar sind. Die manuelle Verarbeitung solcher hochdimensionaler Strukturen ist im realen Anwendungsfall nicht praktikabel, da sie mit einem äußerst hohen Aufwand verbunden ist.

Um maschinelles Lernen zu ermöglichen, müssen messbare Eigenschaften (Features) der Daten extrahiert werden. Jedes Datenobjekt wird durch einen Feature-Vektor dargestellt, welcher die extrahierten Feature-Werte beinhaltet. Die Features sind im besten Fall so beschaffen, dass sie eine Unterscheidung zwischen Datenobjekten unterschiedlicher Klassen gestatten (vgl. Duda et al. 2001, S. 3f.).

ML-Methoden werden grundsätzlich in überwachte (supervised) und unüberwachte (unsupervised) Verfahren unterteilt.

2.3.2 Überwacht vs. Unüberwacht

Der Ausgangspunkt für überwachte und unüberwachte Verfahren ist derselbe: Nachdem die entsprechenden Features aus den Daten extrahiert wurden, werden Datenobjekte in Form von Featurevektoren dargestellt. In der weiteren Vorgehensweise unterscheiden sich die beiden Verfahren.

Überwachte Verfahren arbeiten neben den Featurevektoren mit der Information über die Klassenzugehörigkeit der Datenpunkte. Mit dem Wissen welche Punkte zu welcher Klasse gehören wird entweder ein Modell konstruiert, das diese Aufteilung beschreibt – oder eine Entscheidungsgrenze definiert, die die Daten dementsprechend separiert (siehe Abbildung 2.4). In höherdimensionalen Vektorräumen wird diese Grenze durch eine Hyperfläche dargestellt (vgl. Duda et al. 2001, S. 4f.).

Das Zuordnen der Datenpunkte zu der entsprechenden Klasse wird als Labeln beschrieben – die Datenpunkte werden beschriftet/gekennzeichnet, sie sind gelabelt. Das Wissen über die Klassenzugehörigkeit der Punkte wird als Ground Truth bezeichnet. Die Ground-Truth-Daten werden dafür verwendet ein Modell zu generieren, das ein ungesehenes neues Datenobjekt der richtigen Klasse zuordnen kann. Die Mächtigkeit überwachter Verfahren besteht also darin, aus gelabelten Daten Muster zu lernen, um neue und ungelabelte Daten der korrekten Klasse zuordnen zu können. Solche Verfahren werden deshalb auch als Klassifikatoren bezeichnet (vgl. Murphy 2012, S. 2).

Es existieren verschiedene Klassifikationsverfahren. Sie unterscheiden sich in der Verwendung unterschiedlicher Algorithmen für das Lernen bzw. Training der gelabelten Daten. Zwei Beispiele dafür, die sich grundlegend voneinander unterscheiden, sind Bayes-Klassifikatoren und Support Vector Machines (SVM): Bayes-Klassifikatoren

generieren ein Modell aufgrund der Featureausprägungen unter der Klassenzugehörigkeit. Ein ungelabeltes Datenobjekt wird dem Modell nach der wahrscheinlichsten Klasse zugeordnet. Eine SVM hingegen betrachtet die Featurevektoren im Vektorraum und sucht eine Entscheidungsgrenze, die die Daten am besten voneinander trennt. Ein ungelabeltes Datenobjekt wird einer Klasse je nach Position im Vektorraum zugeordnet.

Unüberwachte Verfahren funktionieren im Gegensatz zu überwachten Verfahren ohne gelabelte Daten. Das heißt, es gibt keinen Ground Truth und es gibt auch kein Training. Stattdessen wird versucht, in den Featurevektoren zusammenhängende, ähnliche oder sich abgrenzende Strukturen und Muster zu finden. Das Ziel besteht darin, die Daten nach bestimmten Kriterien zu kategorisieren, homogene Datenpunkte zusammenzufassen und inhomogene zu trennen. Die Stärke unüberwachter Verfahren liegt darin, ungelabelte Daten zu analysieren und daraus eine Struktur zu erkennen, die Aufschluss über die Zusammensetzung der ungeordneten Menge an Datenpunkten gibt (vgl. Webb 2002, S. 5f.).

Es existieren verschiedene Formen unüberwachter Verfahren. Einen wesentlichen Teil nehmen die sogenannten Clusterverfahren ein, die in Abschnitt 3.2 näher erläutert werden. Dimensionsreduktions-Verfahren und Self Organizing Maps (SOM) sind weitere Beispiele, die der Klasse der unüberwachten Verfahren zuzuschreiben sind.

Um die grundsätzliche Vorgehensweise zu verdeutlichen soll ein hypothetisches Beispiel betrachtet werden: Ein Artenforscher ist daran interessiert, eine automatisierte Aussage darüber zu treffen, ob ein Papageienvogel eher ein Kakadu oder ein Wellensittich ist (2 Klassen). Als Features werden die Länge des Schnabels und die Farbvielfalt des Gefieders gewählt. Nachdem die Features der untersuchten Vögel extrahiert worden sind, wird jeder Vogel durch einen Punkt im zwei-dimensionalen Vektorraum dargestellt, der die jeweiligen Werte der Schnabellänge und Farbvielfalt repräsentiert. Da dem Artenforscher bekannt ist, welcher dieser Vögel ein Wellensittich ist und welcher ein Kakadu, werden die Punkte dementsprechend gelabelt. Anhand der Ausprägungen und Klassenzugehörigkeiten wird ein überwachtes Verfahren dazu verwendet ein Modell daraus zu generieren. Aus dem Gelernten kann ein neuer Papageienvogel, dessen Klassenzugehörigkeit unbekannt ist, aufgrund seiner Schnabellänge und der Farbvielfalt seines Gefieders der Klasse der Kakadus oder der Klasse der Wellensittiche zugeordnet werden (Klassifikationsaufgabe) – siehe Abbildung 2.4.

Analog dazu könnte sich der Artenforscher dafür interessieren, ob es Unterarten von Kakadus gibt bzw. inwiefern sich diese unterscheiden. Wiederum werden Schnabellänge und Farbvielfalt extrahiert (in diesem Fall nur von Kakadus) und im Vektorraum dargestellt. Nun werden Strukturen gesucht, die eine eigene Klasse darstellen könnten (z.B. Punkte, die sehr nah aneinander liegen und sich von anderen Punkten abgrenzen) - also Gruppen mit ähnlichen Eigenschaften (Cluster). Lassen sich solche Cluster finden, könnte man sie als Unterart definieren (Clusteraufgabe).

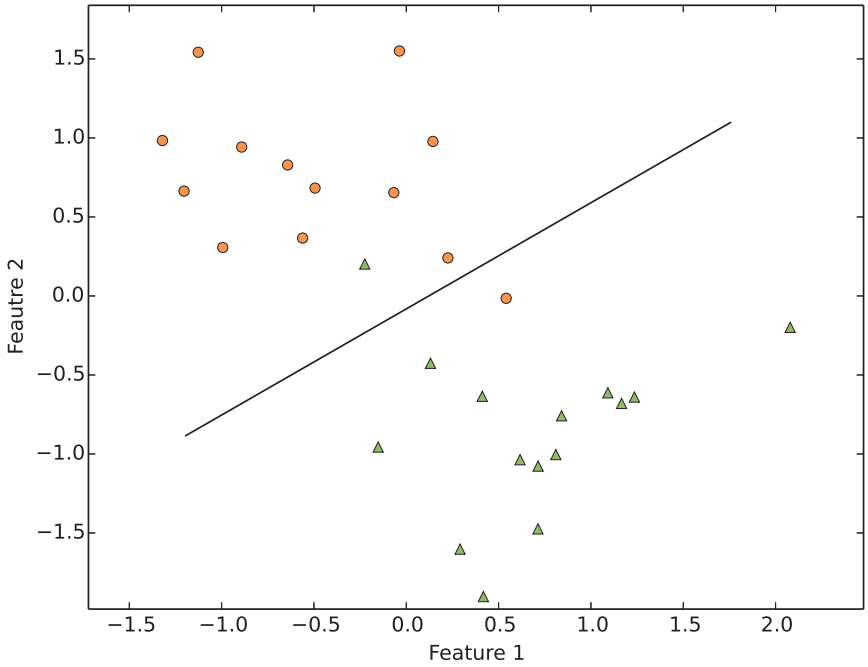


Abbildung 2.4: Datenpunkte zwei verschiedener Klassen in einem zwei-dimensionalen Vektorraum. Die schwarze Linie stellt eine mögliche Entscheidungsgrenze dar (jeweils ein Datenpunkt würde der falschen Klasse zugeordnet werden).

Der grundlegende Unterschied zwischen Klassifikationsaufgaben und Clusteraufgaben kann so zusammengefasst werden, dass beim Klassifizieren aus gegebenen Strukturen gelernt wird, um zukünftige Daten korrekt zuzuordnen – und beim Clustern versucht wird, die Daten an sich zu analysieren und Strukturen innerhalb dieser hervorzuheben (vgl. Hastie et al. 2009, S. 2).

2.3.3 Anwendungsbeispiele in der Stilometrie

Nachdem die beiden Aufgaben der Klassifikation und des Clusterns beschrieben wurden, sollen jeweilige Anwendungsbeispiele in der Stilometrie aufgezeigt werden.

Die aufgelisteten Aufgabenstellungen von Stamatatos (2009) in Abschnitt 2.2.2 können den ML-Methoden laut Ansicht des Autors wie folgt zugewiesen werden:

Authorship Attribution fällt in die Sparte der überwachten Verfahren: mit den Dokumenten der n Autoren wird ein überwachtes Verfahren trainiert, welches ein Modell generiert und das in Frage stehende Dokument einem der n Autoren zuweist (ein Klassifikationsproblem).

Author Verification könnte theoretisch auch mit einem überwachten Verfahren gelöst werden[8], die Zuordnung zu den unüberwachten Verfahren scheint jedoch sinnvoll zu sein. Die vorhandenen Dokumente werden eingelesen und verarbeitet. Das in Frage stehende Dokument wird mit den verarbeiteten Dokumenten (Punkte im Vektorraum) verglichen und beobachtet, ob es den Eigenschaften der bisherigen Punktwolke entspricht oder nicht. Das Ziel besteht darin, das zu prüfende Dokument entweder als Außenseiter (Outlier-Detection) oder als Mitglied der bisherigen Punkte zu erkennen.

Author Profiling fällt in die Sparte der überwachten Verfahren. Wie bei der Authorship Attribution werden gelabelte Dokumente benötigt, die die jeweilige Eigenschaft widerspiegeln (z.B. Geschlecht, Alter etc.). Ein daraus generiertes Modell soll die Texte des zu prüfenden Autors einer Klasse (z.B. männlich/weiblich) zuordnen. Author Profiling ist in diesem Sinne ein Klassifikationsproblem.

Author Independent Characterization kann analog zu Authorship Attribution und Author Profiling betrachtet werden und ist somit auch ein Klassifikationsproblem.

Plagiarism Detection (extrinsic) ist im Grunde eine IR-Aufgabenstellung: Dokumente werden auf inhaltlich übereinstimmende Stellen geprüft. Ein Clustern von Textsegmenten ist prinzipiell zwar möglich, da die Datenmenge der zu vergleichenden Dokumente (der gesamte Korpus wissenschaftlicher Arbeiten) jedoch sehr groß ist, scheint dieser Ansatz nicht sehr praktikabel zu sein.

Plagiarism Detection (intrinsic) / Detection of Stylistic Incosistencies kann in die Sparte der unüberwachten Verfahren eingeordnet werden. Es sind keine Informationen außerhalb des zu prüfenden Dokumentes verfügbar. Es geht darum, im Text innerhalb des Dokumentes Strukturen zu erkennen, die unterschiedliche Schreibstile kennzeichnen. Im Sinne der Plagiatserkennung könnten unterschiedliche Schreibstile auf unterschiedliche Autoren hinweisen.

Neben der primären Aufgabe, ein Klassifikations- oder Clusterproblem zu lösen, werden ML-Methoden auch für weitere Zwecke in der Stilometrie eingesetzt. Beispiele dafür sind Methoden der Dimensionsreduktion und Featureselektion (z.B. Principal Component Analysis (PCA)), die das Arbeiten in einem niedriger-dimensionalen Vektorraum ermöglichen oder nicht relevante Features eliminieren. Auch für sämtliche NLP-Techniken (z.B. die Generierung eines POS-Taggers) werden ML-Methoden angewendet – mehr dazu wird in Abschnitt 3.1.2 erläutert.

In diesem Abschnitt wurden die grundlegenden ML-Konzepte beschrieben und eine Verknüpfung zur Stilometrie aufgebaut (eine detailliertere Beschreibung verschiedener

[8] Da es nur eine Klasse gibt (die Texte mit bekannter Autorenschaft) ist ein Training darauf nicht möglich. Eine überwachte Vorgehensweise könnte darauf abzielen, ein überwachtes Verfahren zu trainieren, das zwei Klassen als Input bekommt: Texte eines einzelnen Autors (1. Klasse) und Texte desselben Autors mit der Zugabe von Texten eines zweiten Autors (2. Klasse). Daraus extrahierte Features könnten eine Aussage über die Klassenzugehörigkeit ermöglichen.

Clusterverfahren folgt in Abschnitt 3.2). Abhängig von der Problemstellung kann ML auf verschiedene Arten eingesetzt werden. Mit welchen Problemen sich andere Arbeiten auseinandergesetzt haben, welche Methodiken dabei zum Einsatz gekommen sind und inwiefern sie sich von der vorliegenden Arbeit unterscheiden, wird im nächsten Abschnitt behandelt.

2.4 Verwandte Arbeiten

Dieser Abschnitt soll einen generellen Überblick über verwandte Arbeiten bieten. Ein beachtlicher Teil der Arbeiten, der sich mit stilometrischen Systemen auseinandersetzt, konzentriert sich auf die Problemstellung der Authorship Attribution/Verification. Deutlich weniger Arbeiten beschäftigen sich mit der intrinsischen Plagiatserkennung bzw. der Erkennung von stilistischen Inkonsistenzen. Die Gebiete sind jedoch, vor allem in der Verwendung der Features, relativ eng miteinander verbunden.

Einige Arbeiten setzen sich nicht mit einer konkreten Problemstellung auseinander, sondern bieten einen generellen Einblick in die Thematik. So liefern Alzahrani et al. (2012) einen guten Überblick über das Gebiet der automatisierten Plagiatserkennung einschließlich einer Taxonomie von Plagiatsvorgehensweisen, den dazugehörigen Erkennungsmethoden und möglichen Features. Der spezielle Bereich der Authorship Attribution wird von Stamatatos (2009) und Juola (2007) gut beschrieben. Vor allem Juola liefert eine äußerst ausführliche Arbeit mit einem Einblick in die Grundlagen stilometrischer Systeme und einem umfassenden Überblick verwandter Gebiete. Li et al. (2006) präsentieren die grundlegenden Schritte, die für stilometrische Analysen notwendig sind und betrachten dabei Unterschiede zwischen der englischen und chinesischen Sprache.

Für die folgende Auflistung untersuchter Arbeiten wird die Einteilung von Abbasi & Chen (2008) übernommen, die prinzipiell zwischen zwei Aufgabenstellungen unterscheidet. Bei der Identifizierung wird ein anonymer Text mit Texten bekannter Autorenschaft verglichen und einem dieser Autoren zugewiesen. Bei der Ähnlichkeitserkennung werden anonyme Texte mit anderen anonymen Texten verglichen und eine Aussage über die Ähnlichkeiten der Texte getroffen.

2.4.1 Identifizierungs-Task

Collins et al. (2004) schlagen die Verwendung von Features vor, die näher am menschlichen Verständnis eines Textes liegen als die gängigen abstrakten Features bisheriger Arbeiten. Sie testen ihren Ansatz an den bereits bekannten Federalist Papers (Mosteller & Wallace 1963) und filtern Wörter und Phrasen verschiedener Kategorien (Verwendung von positiven/negativen Ausdrücken, vergangenheitsorientiert/zukunftsorientiert), die einen Rückschluss auf den rhetorischen Stil eines Autors ermöglichen sollen. Es ergeben sich 18 verschiedene Features (Beispiele in Klammer): First Person (I'm, I have), Think

Positive (easy, a welcome change, a new relationship with), Think Ahead (will get into, will want to), Think Back (has made, may have pretended), Past Events (cought a case, got set up), Motion (open mail, tap peoples phones) und Time Interval (already, last time, during) sind einige davon. Mit einer Diskriminanzanalyse kommen sie auf dieselben Ergebnisse wie Mosteller & Wallace 1963.

Luyckx et al. (2008) kritisieren die verwendeten Daten bisheriger Arbeiten dahingehend, dass die Anzahl verschiedener Autoren zu gering sei und die Testdaten sich generell auf zu lange Texte beschränken würden. Außerdem wird bemängelt, dass viele Arbeiten den inhaltlichen Aspekt der Daten (Genre/Thema) außer Acht lassen. So sei eine erfolgreiche Klassifizierung oft auf unterschiedliche Inhalte der Texte und nicht auf den effektiven Schreibstil der Autoren zurückzuführen. Im Rahmen der Arbeit wird eine Autorenzuweisung niederländischer Buchtexte unternommen. Dabei wird die Textlänge und die Anzahl der Autoren variiert und mit einer SVM trainiert. Sie untersuchen die verwendeten Features (POS-N-Gramme, lexikalische N-Gramme, Funktionswörter und verschiedene Wortschatzmaße) getrennt voneinander (die Verwendung eines kombinierten Feature-Sets geht aus der Arbeit nicht hervor) und kommen zu dem Ergebnis, dass mit steigender Anzahl an Autoren und kürzeren Texten die Performance deutlich sinkt.

Juola (2009) stellt JGAAP vor – eine Applikation, welche den Task der Autorenzuweisung in Java implementiert und zum Ausprobieren gängiger ML-Verfahren dienen soll. Die Applikation ist mit einer grafischen Oberfläche ausgestattet, welche den Prozess schrittweise unterstützt: vor der Wahl des zu verwendenden ML-Verfahrens können Eingabetexte gefiltert und weiterverarbeitet (PDF, HTML etc.) und eine Auswahl der zu extrahierenden Features getroffen werden.

Ein neues Maß für linguistische Features wird von Koppel et al. (2006) vorgestellt: die Stabilität eines Wortes, ein Maß für die Ersetzbarkeit durch vorhandene Synonyme. Instabile Wörter könnten durch eine Vielzahl anderer Wörter ersetzt werden, ohne die Semantik des Satzes zu verändern. Solche Wörter sollen einen Aufschluss über den Schreibstil eines Autors liefern, da er sich speziell für dieses Wort entschieden hat. Stabile Wörter hingegen lassen kaum Spielraum. Als Features werden deshalb die Frequenzen von häufig verwendeten und instabilen Wörtern verwendet. Ein großer Teil dieser Wörter besteht aus Funktionswörtern, die sich seit der Verwendung in den Federalist Papers wiederholt als bewährt erwiesen haben. Auch Clark & Hannon (2007) verfolgen dieses Prinzip in ihrer Arbeit (scheinbar unabhängig[9] von Koppel et al. 2006). Sie implementieren ein System, das die Stabilität eines Wortes anhand diverser NLP-Methoden misst und eine Gewichtung anhand der auftretenden Häufigkeit innerhalb eines Textes trifft. Mit den Daten aus literarischen Werken vier bekannter Autoren erreichen sie eine positive Performance (F-Maß über 90%). Welches überwachte ML-Verfahren verwendet wird konnte nicht festgestellt werden. Obwohl das auf Synonymen basierende Feature für den Author Attribution Task verwendet wurde, könnte sich das Feature auch für unüberwachte Aufgaben eignen.

[9] Eine Referenz auf die Arbeit von Koppel et al. (2006) konnte nicht gefunden werden.

Savoy (2012) ist ein weiteres Beispiel für die Konzentration auf einen einzelnen Featuretyp. Für die Autorenzuweisung von Zeitungsartikeln (Zeitungen: *Glasgow Herald* und *La Stampa*) verwendet Savoy ein spezifisches Wortschatzmaß, welches sich auf die am häufigsten verwendeten Wörter pro Autor bezieht und mit einer Filterung auf bestimmte Wortarten (z.B. mit oder ohne Personalpronomen) durchgeführt wird. Für die Modellbildung wird ein naiver Bayes-Klassifikator verwendet. Erwähnenswert ist die unterschiedliche Performance der beiden Zeitschriften: die Autorenzuweisung der Artikel der italienischen *La Stampa* schnitt wesentlich besser ab als jene der englischen *Glasgow Herald*.

Brennan et al. (2012) verfolgen einen entgegengesetzten Ansatz. Um die Anonymität eines Autors zu bewahren und dessen Privatsphäre zu schützen, präsentieren sie *Adversarial Stylometry*, eine Methodik, die sich auf die Täuschung einer stilometrischen Autorenzuweisung konzentriert. Verschiedene Feature-Vektoren und Verfahren kommen zum Einsatz: darunter sind die bereits erwähnte synonym-basierte Variante von Collins et al. (2004) und die Writeprint-Methode von Abbasi & Chen (2008), die im Anschluss vorgestellt wird. Mit der Applikation Anonymouth[10] (basiert auf JGAAP) kann nachvollzogen werden, welche Stellen im Text ausschlaggebend für die Zuweisung zu einem bestimmten Autor sind. Dadurch wird der Benutzer unterstützt seinen Text so anzupassen, dass Authorship Attribution Tools getäuscht werden können. Brennan et al. kommen zu dem Schluss, dass aktuelle Methoden zwar gute Ergebnisse liefern, die Daten jedoch so verschleiert werden können, dass ihre Performance stark vermindert wird und sich jener einer zufälligen Zuweisung annähert.

2.4.2 Ähnlichkeitserkennung

Graham et al. (2005) zielt auf eine stilometrische Segmentierung von Dokumenten ab. Ein eigener Korpus aus Usenet-Postings wird erstellt und ein künstliches neuronales Netz wird auf die Erkennung von Stilübergängen zwischen Absätzen trainiert. Es gibt weder eine Zuordnung noch ein Clustering, welches die verschiedenen Textsegmente mit den jeweiligen Autoren verbindet. Man beschränkt sich auf die Aussage, ob zwischen Absätzen ein Stilbruch zu erkennen ist oder nicht. Vor allem syntaktische Features werden eingesetzt: Satzlänge, Wortlänge, Silbenlänge, Part-of-Speech (POS)-Frequenzen, 40 verschiedene Funktionswörter und Interpunktion. Ein Durchmischen bzw. eine zufällige Anordnung der Absätze wurde nicht getestet. Das wird damit begründet, dass in der Realität jene Absätze auch aufeinander folgen und man den realen Zustand so gut wie möglich nachbilden möchte. Außerdem wird eine fixierte Länge an Texteinheiten als Input für das Training verwendet, wobei die Struktur der verwendeten Daten einen wesentlichen Einfluss auf das Funktionieren dieser Vorgehensweise haben könnte (mögliche Überanpassung der Daten). Es wurde ein überwachtes Verfahren angewandt, wobei 90% der Daten für das Training und 10% für das Testen verwendet wurden.

[10] `https://psal.cs.drexel.edu/index.php/JStylo-Anonymouth`, abgerufen am 25.08.2014

Gruner & Naven (2005) stellen eine Applikation vor, die eine Aussage trifft, ob zwei Texte vom selben Autor stammen oder nicht. Dabei werden Textsegmente, die 5000 Wörter umspannen, auf 65 verschiedene Muster untersucht. Diese 65 Muster basieren vor allem auf Funktionswörtern und deren Kontext im Satz: Funktionswort + Nomen, Verb + Funktionswort + Adjektiv etc. Aus der Frequenz dieser Muster wird ein Score berechnet und einem Intervall zugeordnet. Aus der ursprünglichen Arbeit dieser Methodik geht hervor, dass die Texte eines Autors nicht mehr als drei verschiedene Intervalle besetzen. Gruner & Naven erhalten positive Ergebnisse um den Grenzwert von 8 unterschiedlichen Intervallen pro Autor. Interessant scheint der Ansatz Funktionswörter im syntaktischen Kontext des Satzes zu sehen. Nachteile dieser Methodik ergeben sich vor allem durch die vorgegebene, relativ hohe Mindestlänge der Texte und die Abhängigkeit des gewählten Grenzwertes für die Anzahl der Scores, wobei die Festlegung von Grenzwerten bei der Erkennung von Ähnlichkeiten durchaus üblich ist.

Meyer & Stein (2006) führen eine intrinsische Plagiatserkennung durch. Das eigens entwickelte Wortschatzmaß wird als *Averaged Word Frequency Class* bezeichnet: Wörter werden nach ihrer Häufigkeit in Klassen unterteilt. Die Word Frequency Class eines Wortes wird dadurch berechnet, dass die Frequenz des Wortes in Beziehung zur Frequenz des Klassenvertreters (das Wort mit der höchsten Frequenz derselben Klasse) gestellt wird. Eine Durchschnittsbildung dieser Werte soll den Stil eines Autors erkennbar machen. Es konnte nicht festgestellt werden, welche Methoden konkret zum Einsatz gekommen sind. Die Ergebnisse (ein F-Maß um die 80%) wurden anhand einer Diskriminanzanalyse erreicht und durch eine SVM bestätigt. Beide Verfahren gehören jedoch zu den überwachten Verfahren und sind für eine intrinsische Plagiatserkennung in dieser Form nicht aussagekräftig.

Zechner et al. (2009) präsentieren eine Lösung für eine extrinsische und intrinsische Plagiatserkennung. Für die extrinsische Plagiatserkennung werden Term-Frequenzen verwendet und eine Nearest-Neighbor-Suche angestellt. Für die intrinsische Plagiatserkennung werden folgende Features verwendet: POS-Frequenzen, Frequenzen von bestimmten Pronomen, Funktionswortfrequenzen, Interpunktion und ein Wortschatzmaß. Auf Satzebene werden die Features extrahiert und die Distanz zum Durchschnittsvektor des gesamten Textes gemessen. Anschließend werden Ausreißer gesucht (Outlier Detection), die eine hohe Distanz zum Durchschnittsvektor aufweisen. Im Gegensatz zur extrinsischen konnten sie mit der intrinsischen Ausreißer-Suche keine positive Performance erzielen (ein durchschnittliches F-Maß von 22%).

Iqbal et al. (2009) verwenden Abbasis & Chens Writeprint-Methode um E-Mails verschiedener Autoren zu clustern. Mit einem Feature-Set, das dem der Writeprint-Methode sehr ähnlich ist, werden E-Mails mithilfe des Expectation-Maximazation-Algorithmus (EM) und K-Means geclustert. Dabei werden Ergebnisse mit F-Maßen im Bereich zwischen 70% und 80% erreicht. Inwiefern die Parameter für den EM-Algorithmus bzw. das k für K-Means geschätzt oder festgelegt werden konnte aus der Arbeit nicht eruiert werden.

2.4.3 Abbasis & Chens Writeprint-Methode

Die Writeprint-Methode von Abbasi & Chen (2008) wird häufig zitiert und in verschiedenen Bereichen angewendet. Aufgrund ihrer hervorragenden Performance auch mit höherer Anzahl an Autoren gilt sie in den Bereichen der Authorship Attribution und Similarity Detection als Stand der Technik. Abbasi & Chen führen ein erweitertes Feature-Set ein, welches sich durch individuelle Features (getrennt pro Autor) charakterisiert. Die Features umspannen mehrere lexikalische, syntaktische, strukturelle und inhaltsbezogene Eigenschaften, wobei sich die Feature-Vektoren der jeweiligen Autoren vor allem durch die verschiedenen Ausprägungen der N-Gramme unterscheiden.

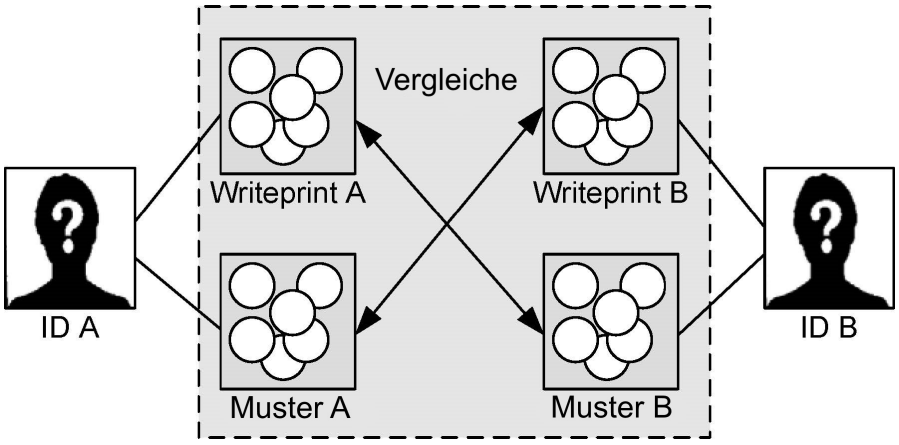


Abbildung 2.5: Ähnlichkeitsmaß der Writeprint-Methode von Abbasi & Chen (2008).

Um zwei Texte miteinander zu vergleichen werden die individuellen Features mit dem Sliding Window (umfasst ca. 250 Wörter) extrahiert und mit einer Karhunen-Loève-Transformation (KLT) transformiert – eine überwachte Variante der PCA, die einen Vergleich verschiedener Feature-Sets zulässt (vgl. Webb 2002). Die daraus resultierenden Hauptkomponenten bezeichnen Abbasi & Chen als Writeprints. Sowohl für die Autorenidentifizierung als auch für die Ähnlichkeitserkennung werden die Writeprints miteinander verglichen. Der Autor des Dokumentes, dessen Writeprint dem Writeprint des zu prüfenden Dokumentes am nächsten ist, wird dem fraglichen Dokument zugewiesen (Identifizierungs-Task); für die Ähnlichkeitserkennung werden jene Dokumente zu einer Identität zusammengefasst, deren Writeprint-Ähnlichkeit einen festgelegten Grenzwert überschreiten.

Das Vergleichsmaß wird in Abbildung 2.5 dargestellt. Segment A wird mit dem Feature-Vektor und der Transformations-Matrix von Segment B analysiert und mit den transformierten Features von Segment B verglichen und umgekehrt. Um die Nichtexistenz eines individuellen Features zu betonen werden sogenannte Disruption Values zur jeweiligen Transformations-Matrix hinzugefügt. Mit der beschriebenen Methode erreichen Abbasi & Chen F-Maße bis zu 94%.

2.4.4 Zusammenfassung & Abgrenzung

Zusammenfassend kann festgestellt werden, dass sich die betrachteten Arbeiten besonders in der Wahl der Feature-Sets voneinander unterscheiden und im speziellen in der verwendeten Methodik. Die Wahl des Feature-Sets könnte dabei womöglich die wichtigere Rolle spielen: Brennan et al. (2012) haben für die Implementierung der Writeprint-Methode in Anonymouth eine SVM-Variante mit Abbasis & Chens umfangreichen Feature-Set getestet und kamen dabei auf eine Performance, die mit der speziellen Writeprint-Methode vergleichbar ist.

Es lässt sich beobachten, dass Authorship-Attribution-Aufgaben klarer definiert und dokumentiert sind als intrinsische Plagiatserkennungsaufgaben (oder generell Aufgaben die eine Ähnlichkeitserkennung verfolgen). Viele der Arbeiten, die sich auf die Ähnlichkeitserkennung konzentrieren, gehen außerdem davon aus, dass die zu analysierenden Texte abgegrenzte Dokumente sind (Bsp. Cluster von E-Mails), deren Ähnlichkeit zueinander verglichen werden kann. Auch die Writeprint-Methode geht von dieser Grundannahme aus, weshalb sie für die vorliegende Arbeit nicht anwendbar ist. Für die intrinsische Plagiatserkennung ist nicht definiert, wie lang die Texte der verschiedenen Autoren sind bzw. wo sie beginnen und wo sie enden. Zudem müssen die Segmente relativ kurz sein, um den Übergang von einem Autor zum nächsten erkennen zu können.

Die vorliegende Arbeit beschäftigt sich mit dieser Problemstellung der intrinsischen Plagiatserkennung. Textstellen innerhalb eines Dokumentes mit undefinierten Grenzen werden verglichen. Es wird versucht mit Hilfe von Clusterverfahren eine Struktur der Daten zu erkennen und stilistisch ähnliche Textstellen zusammenzufassen.

Mit dem Überblick über die theoretischen Konzepte stilometrischer Systeme wurden einige Methoden namentlich teilweise schon vorweg genommen. Die konkrete Extraktion textueller Features, verschiedene Clusterverfahren und deren Evaluierung werden im nächsten Kapitel erläutert.

Kapitel 3

Methodik

Dieses Kapitel befasst sich mit den Methoden, die für die vorliegende Aufgabenstellung von Bedeutung sind. Welche Verfahren konkret zum Einsatz gekommen sind und sich gegenüber anderen als effizient erwiesen haben, wird in den Kapiteln 4 und 5 erläutert. Die Feature-Extraktion, die jedem Clusterverfahren vorausgeht, stellt den einleitenden Abschnitt des Kapitels dar. Der Fokus liegt dabei auf der Beschreibung textueller Features und den Techniken, die hinter der Extraktion selbiger stehen. Den Hauptteil nimmt die Erläuterung einiger ausgewählter Clusterverfahren ein, mit denen versucht wird, eine intrinsische Plagiatserkennung durchzuführen. Deren Funktionsweise sowie ihre Stärken und Schwächen werden diskutiert. Abschließend folgt eine Auseinandersetzung mit der Evaluierung von Clustern.

3.1 Feature-Extraktion

Die Feature-Extraktion bildet den Schritt zwischen den tatsächlichen Daten und der Repräsentation gemessener relevanter Eigenschaften dieser Daten (vgl. Duda et al. 2001, S. 3). Die tatsächlichen Daten sind in diesem Fall Text – relevante Eigenschaften eines Textes sind textuelle Features. Welche textuellen Features es gibt, in welche Gruppen diese unterteilt werden können und mit Hilfe welcher Techniken sie extrahiert werden, wird in diesem Abschnitt behandelt.

3.1.1 Textuelle Features

Die Möglichkeiten, Textattribute numerisch darzustellen, sind vielseitig. Bis in die späten 1990er Jahre wurden bereits etwa 1000 verschiedene Features ausgearbeitet (vgl. Rudman 1998) und bis heute ist nicht eindeutig definiert, welche Features für welche Aufgaben am besten geeignet sind (vgl. Schulstad et al. 2012). Der Grund dafür könnte auch darin liegen, dass eine klare Abgrenzung mit den bisher verwendeten Features nicht möglich ist.

Übersicht und Taxonomie

Die Wahl der zu extrahierenden Features aus einem Text ist essentiell für das Erkennen von stilometrischen Unterschieden. Sie sind der Grundstein für jegliche weitere Verarbeitung eines Textes.

> *[...], the most complicated and accurate technique cannot aid the practitioner whose data is flawed.* (Tweedie 2005, S. 394)

Einige Features sind gut erkennbar und konkret mit dem Text verbunden (z.B. Satzlänge, Wortlänge), andere Features wiederum sind abstrakter und lassen sich oft erst durch komplexere statistische Berechnungen formulieren (z.B. Wortschatzmaße). Eine gängige Taxonomie textueller Features wird in Tabelle 3.1 dargestellt. Textuelle Features werden häufig in lexikalische, syntaktische, semantische, strukturelle/applikationsspezifische und idiosynkratische Features kategorisiert (vgl. Abbasi & Chen 2008; Alzahrani et al. 2012; Li et al. 2006; Stamatatos 2009). Die angeführten NLP-Techniken werden in Abschnitt 3.1.2 beschrieben.

Lexikalische Features arbeiten auf einer niedrigen Ebene. Ein Text wird als eine Sequenz von Tokens betrachtet. Jeder Token entspricht entweder einem Zeichen (Buchstaben, Ziffern, Satzzeichen, Sonderzeichen), einem Wort oder einer Zahl (vgl. Stamatatos 2009).

Beispiele: Wortschatzmaße, Frequenzen und N-Gramme von Buchstaben, Ziffern, Satzzeichen, Wörtern, Wortlängen, Satzlängen etc.

Ein Vorteil lexikalischer Features ist die Unabhängigkeit der Sprache (sie funktionieren für die englische Sprache genauso wie für die deutsche). Wobei Sprachen, deren Wort- oder Zeichentrennung nicht trivial ist, eine Ausnahme bilden (z.B. Chinesisch). Für die Extraktion wird lediglich ein Tokenisierer (engl. Tokenizer)[1] benötigt. Sollen für die Wortschatzmaße nicht die exakten Wörter verwendet werden, sondern die jeweilige Stammform, ist auch ein Stemmer[1] notwendig (vgl. Stamatatos 2009). Die Erläuterungen der Wortschatzmaße und des N-Gramm-Konzeptes erfolgen im Anschluss.

Syntaktische Features arbeiten auf einer höheren Ebene als lexikalische Features. Der Text wird nicht als reine Zeichenkette gesehen, vielmehr wird die Syntax betrachtet, die sich dahinter befindet. Man geht davon aus, dass Autoren dazu tendieren, syntaktisch ähnliche Muster zu verwenden (vgl. Stamatatos 2009).

Beispiele: Frequenzen und N-Gramme von Wortarten (POS), Funktionswörtern und Interpunktion.

Wortartfrequenzen und N-Gramme repräsentieren eine relativ simple Darstellung der Syntax eines Satzes. Jedes Wort wird durch seine Wortart (z.B. Nomen singular/plural, Adjektiv komparativ/superlativ etc.[1]) ersetzt bzw. gekennzeichnet

[1] Details siehe Abschnitt 3.1.2

Feature-Typ	Features	Beispiele
Lexikalisch	Wort-N-Gramme	Unigramme (Wortfrequenz/Bag-of-Words), Bigramme, Trigramme
	Wortlänge	mittlere Wortlänge, Wortlängenfrequenz, Wortlängen-N-Gramme
	Satzlänge	mittlere Satzlänge, Satzlängenfrequenz, Satzlängen-N-Gramme
	Buchstaben-N-Gramme	Unigramme (Buchstabenfrequenz), Bigramme, Trigramme, Filterung & Variation mit Vokalen & Konsonanten etc.
	Ziffern-N-Gramme	Unigramme (Ziffernfrequenz), Bigramme, Trigramme etc.
	Wortschatzmaße	Hapax Legomena, Dis Legomena, Yule's K etc.
Syntaktisch	Wortart (POS)	Wortartfrequenz, Wortart-N-Gramme etc.
	Interpunktion	Frequenz einzelner Satzzeichen, Verhätlnis von Satzzeichen zur gesamten Anzahl an Zeichen etc.
	Funktionswörter	Funktionswortfrequenzen
Semantisch	Synonyme	Stabilität eines Wortes, Wortklassen etc.
	Semantische Strukturen	N-Gramme aus Hauptsätzen, Nebensätzen, Nominalphrasen etc.
Strukturell/Applikationsspezifisch	Absatzlänge	Absatzlängenfrequenz, Absatzlängen-N-Gramme
	Formatierung	Schriftartformatierung, Layout
	Zitate	Zitatenanzahl, Zitatenlänge, indirekt/direkt zitiert
	Sprache	Anzahl verwendeter Sprachen, wo/wann/wie oft Sprachwechsel
Idiosynkratisch	Rechtschreib- & Grammatikfehler	beleive, beutiful
	amererikanische vs. britische Ausdrücke	color - colour

Tabelle 3.1: Textuelle Featurekategorien.

und der daraus resultierende syntaktische Aufbau analysiert. Die Ermittlung der Wortarten wird in der Regel automatisiert von einem POS-Tagger[1] übernommen, wobei eine gewisse Fehlerrate in Kauf genommen werden muss (vgl. Juola 2007, S. 265).

Funktionswörter sind jene Wörter, die für die Satzbildung notwendig sind, jedoch keine Aussage über den Inhalt machen (z.B. Artikel, Präpositionen, Pronomen etc.). Sie stellen eine teils unterbewusste Wahl dar, wie die wesentlichen Elemente eines Satzes miteinander verknüpft werden sollen. Auf diese Weise können sie Rückschlüsse über den Schreibstil eines Autors liefern (vgl. Juola 2007, S. 242). In Abbildung 3.1 werden Funktionswörter dargestellt, gewichtet nach der auftretenden Häufigkeit in englischen Wikipedia-Artikeln. Die Selektion der Funktionswörter kann entweder manuell erfolgen oder automatisiert durch die Selektion der n-häufigsten Wörter eines Korpus[2] mit anschließender manueller Filterung oder Erweiterung.

Die Interpunktion kann in die Gruppe der syntaktischen Features eingeordnet werden (vgl. Li et al. 2006), da sie zum Aufschluss der Syntax beiträgt (die Frequenz von Beistrichen lässt z.B. auf den Verschachtelungsgrad eines Satzes schließen). Sie könnte aber ebenso der Gruppe der lexikalischen Features zugeordnet werden (Frequenz von Satzzeichen).

Syntaktische Features haben den Nachteil, dass sie größtenteils abhängig von der Sprache sind und individuell angepasst werden müssen (Funktionswörter, Wortartbestimmung). Komplexere syntaktische Strukturen könnten mit der Erkennung von Nominal- und Verbalphrasen, Haupt- und Nebensätzen etc. realisiert werden, was jedoch bereits stark in die Richtung semantischer Features geht (vgl. Stamatatos 2009).

Semantische Features arbeiten auf der höchsten Ebene der bisherigen Featuregruppen. Der Text wird semantisch interpretiert, die Struktur und Bedeutung eines Satzes kann nachvollzogen werden (vgl. Stamatatos 2009).

Beispiele: Art von Nebensätzen (Ausführung, Erklärung, Erweiterung, Steigerung des vorhergehenden Satzes) und Strukturen selbiger (z.B. Anzahl der Nebensätze, die nötig war, eine Erklärung abzuschließen).

Da ein Verstehen der Semantik eines Satzes komplex ist und die Implementierung sich dementsprechend nicht einfach gestaltet, haben semantische Features in der Stilometrie noch keine breite Anwendung gefunden (vgl. Cambria & White 2014).

Synonymbasierte Features wie die von Koppel et al. (2006) und Clark & Hannon (2007) können auch zu den semantischen Features gezählt werden, da sie sich gewissermaßen mit der Semantik eines Wortes auseinandersetzen (Synonyme). Die Implementierung gestaltet sich jedoch wesentlich einfacher und ist mit dem semantischen Parsen eines Satzes kaum vergleichbar.

Strukturelle/applikationsspezifische Features sind stark von der Datendomäne abhängig. Es geht darum, strukturelle Eigenschaften eines Textes zu extrahieren (vgl. Alzahrani et al. 2012).

[2] Funktionswörter gehören zu den am häufigsten verwendeten Wörtern einer Sprache, deshalb ist die Selektion der n häufigsten Wörter eines Korpus eine gängige Methode (vgl. Stamatatos 2009).

Beispiele: Absatzlänge, Verwendung von Grußformeln, Länge und Anzahl von Zitaten, Schriftgröße, Schriftart und weitere Formatierungen.

Die Implementierung muss applikationsspezifisch angepasst werden.

Idiosynkratische Features sind jene Features, die die individuelle Wahl von Wörtern oder Schreibweisen eines Autors widerspiegeln, die sich aufgrund kultureller Einflüsse entwickelt haben oder einfach zu den gängigen Rechtschreibfehlern des Autors gehören (vgl. Juola 2007, S. 267f.).

Beispiele: Rechtschreib- und Grammatikfehler, amerikanische vs. britische Ausdrücke oder Schreibweisen etc.

Für die Implementierung solcher Features sind Rechtschreib- und Grammatikprüfer notwendig. Auch ein manuelles Selektieren von bestimmten Wörtern (z.B. center – centre, color – colour) ist möglich.

Abbildung 3.1: Funktionswörter der englischen Sprache. Die Größe entspricht der auftretenden Häufigkeit in Artikeln aus der englischsprachigen Ausgabe der Wikipedia (Damakos 2011).

Wortschatzmaße

Autoren weisen einen unterschiedlichen Wortschatz auf – manche greifen auf einen reicheren Wortschatz zurück und gebrauchen viele seltene Wörter, andere drücken sich mit einem geringeren Wortschatz aus und verwenden wenige seltene Wörter. Nach diesem Prinzip versucht man Autoren durch Wortschatzmaße zu unterscheiden (vgl. Hoover 2003). Tweedie & Baayen (1998) haben 17 verschiedene Maße analysiert und deren Funktionsweise beschrieben. Der nachfolgende Überblick orientiert sich daran.

Eine grundlegende Einheit für die simpelsten Wortschatzmaße ist die Größe des Vokabulars (vocabulary size) $V(N)$. Abhängig von der Textlänge (Anzahl der Wörter) N wird damit

die Anzahl an unterschiedlichen Wörtern eines Textes ausgedrückt. Ein Wortschatzmaß, das sich der Vokabulargröße bedient, ist die sogenannte Type-Token-Relation (TTR). Die Anzahl verschiedener Wörter wird der gesamten Anzahl gegenüberstellt, wodurch formuliert wird, wie verschiedenartig das verwendete Vokabular eines Textes ist:

$$TTR(N) = \frac{V(N)}{N}$$

Es wurden viele Maße definiert, die eine Variation der TTR darstellen. Unumstritten ist dabei jedoch die Abhängigkeit von der Textlänge N: je länger der Text ist, umso größer wird das Vokabular – wobei die Steigung des Vokabulars nichtlinear ist und mit zunehmender Länge abnimmt.

Eine andere Vorgehensweise stellen Maße dar, die sich auf die Frequenz unterschiedlicher Wörter beziehen. *Hapax Legomena* $V(1, N)$ sind nur einmal vorkommende Wörter, *Dis Legomena* $V(2, N)$ sind genau zweimal vorkommende Wörter etc. Verschiedene Wortschatzmaße wurden nach diesem Prinzip, jeweils in Relation zu N oder $V(N)$, definiert $(X = 1, 2, ..., N)$:

$$\frac{V(X, N)}{N}, \qquad \frac{V(X, N)}{V(N)}$$

Yule (1944) schloss alle Frequenzen $(X = 1, 2, ..., N)$ mit ein und behauptete, dass sein Maß (*Yules K*) unabhängig von der Textlänge sei:

$$K = 10^4 \left[-\frac{1}{N} + \sum_{i=1}^{N} V(i, N) \left(\frac{i}{N} \right)^2 \right]$$

Weitere Wortschatzmaße, die ebenfalls als unabhängig von der Textlänge gelten, sind Maße die sich auf das Zipfsche Gesetz beziehen, wonach die Frequenz eines Wortes umgekehrt proportional zum Häufigkeitsrang des Wortes steht. Das häufigste Wort tritt wesentlich häufiger auf als das zweithäufigste etc. Ein Wort des Ranges n weist eine Auftrittswahrscheinlichkeit von $p(n) \sim \frac{1}{n}$ auf (vgl. Manning & Schütze 1999, S. 24f.). Orlovs[3] generalisiertes Zipf Modell ist eine Funktion mit einem freien Parameter Z und p^*, das für die Frequenz des häufigsten Wortes dividiert durch die Textlänge N steht:

$$V(N) = \frac{Z}{\log(p^* Z)} \frac{N}{N - Z} \log(N/Z)$$

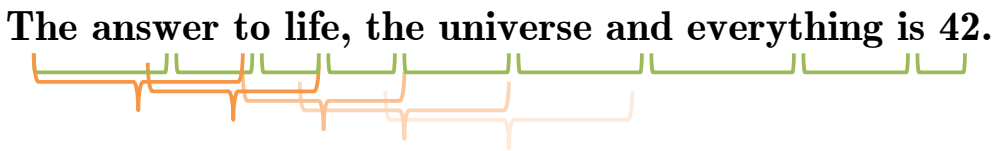
Tweedie & Baayen (1998) schließen aus ihrer Analyse, dass ein Großteil der definierten Wortschatzmaße stark von der Textlänge abhängt. Ausnahmen bilden dabei die Modelle

[3] Orlov (1983) zitiert in Tweedie & Baayen (1998).

von Yule und Zipf. Dennoch legen sie die Verwendung eines einzelnen Maßes nicht nahe und raten zu einer Kombination mit anderen textuellen Features. Hoover (2003) hat ähnliche Wortschatzmaße mit homogeneren Daten untersucht und kommt zu dem Schluss, dass sich Wortschatzmaße generell nicht zur Stilanalyse eignen.

N-Gramme

N-Gramme sind eine Sequenz von n Einheiten eines Textes. Diese Einheiten können entweder einzelne Zeichen oder Wörter, aber auch abstraktere Einheiten sein, wie Wortarten (POS), Wortlängen, Satzlängen etc. Durch N-Gramme wird der umliegende Kontext einer Texteinheit einbezogen, wobei der Inhalt des Textes einen starken Einfluss auf die Ausprägung der N-Gramme hat (vlg. Stamatatos 2009).



Bigramme: (the, answer), (answer, to), (to, life), (life, the), (the, universe), (universe, and), (and, everything), (everything, is), (is, 42)

Trigramme: (the, answer, to), (answer, to, life), (to, life, the), (the, universe, and), (universe, and, everything), (and, everything, is), (everyhting, is, 42)

Abbildung 3.2: Wort-Bigramme und -Trigramme (Beispielsatz aus Adamas 1979).

Der Text wird durch einen Tokenizer[1] in Einheiten zerlegt, aus welchen anschließend Sequenzen der Länge n gebildet werden. Zu beachten sind dabei die natürlichen Abgrenzungen des Textes wie Sätze und Absätze. Eine übergreifende Sequenzbildung zwischen ihnen könnte das Ergebnis verfälschen (die Wörter am Ende eines Satzes sind strukturell von den Wörtern des darauffolgenden Satzes abgetrennt), weshalb eine vorhergehende Zerlegung auf Satzebene sinnvoll sein kann (vgl. Stamatatos 2009). Wort-Bigramme und -Trigramme eines Beispielsatzes werden in Abbildung 3.2 dargestellt.

Gängige Features

Es scheint noch keine Einigkeit darüber zu herrschen, welche Features für welche Aufgabenstellung am nützlichsten sind. Laufend werden neue Maße und Methoden definiert, die eine Lösung für teilweise sehr eingegrenzte Spezialgebiete anbieten (vgl. Schulstad et al. 2012).

Syntaktische Features stellen eine zum Teil unterbewusste Wahl eines Autors dar und werden für die Stilanalyse grundsätzlich als zuverlässiger eingestuft als lexikalische

Features (vgl. Stamatatos 2009). Vor allem die Verwendung von Funktionswörtern hat sich als effizient erwiesen und seit Mosteller & Wallace 1963 bereits über einen längeren Zeitraum bewährt (vgl. Juola 2007). Obwohl lexikalische Features auf einer niedrigeren Ebene arbeiten, scheinen sie für die Stilanalyse dennoch nicht ungeeignet zu sein – allein mit der Verwendung von Zeichen-N-Grammen konnten bereits positive Ergebnisse erzielt werden. Wort-N-Gramme sind hingegen sehr inhaltsabhängig und kommen daher vermehrt in den Gebieten der Themen- und Genre-Erkennung zum Einsatz (vgl. Stamatatos 2009).

Viele Arbeiten gehen davon aus, dass eine Kombination verschiedener Feature-Typen zu besseren Ergebnissen führt als die Beschränkung auf einen einzelnen Typ (vgl. Abbasi & Chen 2008; Schulstad et al. 2012). Wiederholt werden jedoch singuläre Maße definiert, die ohne weitere Features zu guten Ergebnissen führen. Neben den Wortschatzmaßen, deren Funktionstüchtigkeit kritisch betrachtet wird, ist in diesem Zusammenhang die synonymbasierte Variante von Koppel et al. (2006) bzw. Clark & Hannon (2007) erwähnenswert.

Zusammenfassend kann festgestellt werden, dass die Wahl der zu extrahierenden Features stark von der Aufgabenstellung und den zugrunde liegenden Daten abhängt. Welche Features am besten geeignet sind, ist noch nicht eindeutig geklärt. Für die Extraktion textueller Features sind Techniken der NLP notwendig.

3.1.2 Natural Language Processing

Natural Language Processing (NLP) beschäftigt sich mit der automatisierten Verarbeitung natürlicher Sprache. Die Einsatzgebiete von NLP sind relativ breit gestreut: Von Satz- und Worttrennung, über maschinelle Übersetzung, Textzusammenfassung, Spracherkennung und Sentiment Analysis, bis hin zu Semantischem Verstehen eines Satzes sind die Möglichkeiten vielfältig (vgl. Cambria & White 2014). Von einer menschenähnlichen Sprachverarbeitung kann aber noch nicht gesprochen werden:

> *Today, even the most popular NLP technologies view text analysis as a word or pattern matching task. Trying to ascertain the meaning of a piece of text by processing it at word-level, however, is no different from attempting to understand a picture by analyzing it at pixel-level.* (Cambria & White 2014, S. 51)

Die Forschung auf dem Gebiet der NLP geht zunehmend in die Richtung semantischen Verstehens. Entwicklungen auf dem Gebiet der künstlichen Intelligenz könnten für NLP in Zukunft eine bedeutende Rolle spielen (vgl. Cambria & White 2014).

Nachfolgend werden grundlegende NLP-Methoden beschrieben, die für diese Arbeit von Bedeutung sind und sich auf die lexikalische und syntaktische Struktur eines Satzes beschränken.

Tokenisierung

Die Tokenisierung bezeichnet die Tätigkeit, einen Text bzw. eine Zeichenkette in Einheiten (Token) zu zerlegen. Diese Token können einzelne Zeichen, Wörter, Sätze oder auch ganze Absätze sein (vgl. Manning et al. 2008, S. 22f.). Die Extraktion von Features, die auf Wort- oder Satzebene arbeiten, ist ohne eine vorherige Bestimmung der Wort- und Satzgrenzen nicht möglich. Die Tokenisierung ist also für die weitere Verarbeitung eines Textes von Bedeutung und kann daher als Vorverarbeitungsschritt betrachtet werden.

Wörter können aufgrund der Leerzeichen getrennt werden. Ausnahmen bilden Sprachen, die eine Schreibweise ohne Worttrennung aufweisen (z.B. Chinesisch oder Japanisch) (vgl. Stamatatos 2009). Auf Wortebene werden Satzzeichen entweder verworfen oder als eigene Token definiert. Für die Implementierung eignen sich reguläre Ausdrücke.[4]

Probleme bereiten Satz- und Sonderzeichen wie Apostrophe (can't, I'm), Bindestriche (26-year-old, text-based), zusammenhängende Wörter, die mit Leerzeichen getrennt sind (Los Angeles, San Francisco), Maß- und Währungseinheiten (1.050,36 €, 50cm) oder Punkte, die entweder ein Satzende (Punkt gehört nicht zum Wort) oder eine Abkürzung kennzeichnen (Punkt könnte dem Wort zugeordnet werden – als Kennzeichnung der Abkürzung). Es muss entschieden werden, ob und wie diese Wörter getrennt werden sollen (vgl. Manning & Schütze 1999, S. 124ff.).

Die Tokenisierung von Sätzen muss sich mit ähnlichen Problemen auseinandersetzen, stellt aber grundsätzlich eine größere Herausforderung dar. Ein Satz wird durch einen Punkt, ein Fragezeichen, ein Rufzeichen, evtl. aber auch durch einen Strich- oder Doppelpunkt abgeschlossen. Durch die wörtliche Rede ist das Abschließen eines Satzes auch durch Anführungszeichen möglich. Generell sind jene Satzzeichen problematisch, die einen Satz nicht abschließen, sondern eine Abkürzung, Zifferngruppierung oder spezielle Namen kennzeichnen (U.S.A, Mr. Freeman & Co., 1.050,36, Yahoo!). Abkürzungen wie *etc.* können sowohl mitten im Satz als auch am Ende stehen, wobei der Punkt gleichzeitig die Abkürzung und das Ende des Satzes darstellt (vgl. Manning & Schütze 1999, S. 134ff.).

Kiss et al. (2006) haben den sogenannten *Punkt Sequence Tokenizer* entwickelt, der breite Anwendung findet. Die grundlegende Vorgehensweise ist, in einem ersten Schritt bekannte Abkürzungen, Anführungszeichen und Satzzeichen zu markieren, die einen Satz abschließen könnten. In einem zweiten Schritt werden diese Elemente darauf untersucht, ob sie ein Satzende darstellen oder nicht:

- Abkürzungen am Satzende beenden den Satz (obwohl der Punkt einer Abkürzung ansonsten kein Satzende signalisiert).
- Anführungszeichen am Satzende beenden den Satz (obwohl sie nicht zu den satzabschließenden Zeichen gehören).

[4] Reguläre Ausdrücke können Muster beschreiben, die einer regulären Sprache folgen. Es kann definiert werden, wie ein Wort aufgebaut ist: eine Buchstabenfolge abgegrenzt durch Leerzeichen, eine Buchstabenfolge kombiniert mit einem Apostroph und einer weiteren Buchstabenfolge (z.B. *shouldn't*) etc. (vgl. Manning & Schütze 1999, S. 120f.)

- Erkennung von Initialen: Punkte, die im ersten Schritt als satzabschließend markiert wurden, werden aufgehoben, sofern das Initial den Satz nicht abschließt.
- Erkennung von Zifferngruppierungen: Die satzabschließende Markierung von Punkten, die Teil einer Zifferngruppierung sind, wird aufgehoben.

Die Erkennung im zweiten Schritt wird aufgrund der umliegenden Zeichen getroffen (anschließende Leerzeichen, Groß- oder Kleinschreibung, vorhergehende und anschließende Ziffernfolge etc.). Das Tokenisieren von Sätzen kann auch als Klassifikationsaufgabe betrachtet werden (vgl. Bird et al. 2009, S.233 ff.).

Stemming

Stemming bezeichnet das Reduzieren eines Wortes auf seinen Wortstamm. Die vier Wörter *laughing*, *laugh*, *laughs* und *laughed* werden z.B. alle zur Stammform *laugh* reduziert (vgl. Manning & Schütze 1999, S. 534). Für die vorliegende Arbeit ist Stemming bei der Extraktion der Wortschatzmaße von Bedeutung. Wenn ein Autor ein bestimmtes Wort verwendet, kann davon ausgegangen werden, dass er auch sämtliche Konjugationen bzw. Wörter desselben Wortstammes kennt – weshalb es sinnvoll erscheint, Wörter desselben Stammes nicht mehrfach zu berücksichtigen (vgl. Juola 2007, S. 266).

Eine weitere Anwendung findet Stemming in IR-Systemen wie Suchmaschinen. Dabei wird der Suchbegriff modifiziert, um bessere Resultate zu liefern (vgl. Büttcher et al. 2010, S. 86f.). Es kann jedoch argumentiert werden, dass Stemming die Performance von IR-Anwendungen nur bedingt verbessern kann und generell eine höhere Anzahl an Treffern liefert, die Präzision jedoch senkt (vgl. Manning et al. 2008, S. 34). Anstatt den Such-String direkt zu modifizieren, wird Stemming eher als eine Erweiterung der Suchanfrage (query expansion) verwendet, wobei die Verwendung von Synonymen in diesem Zusammenhang eine bedeutendere Rolle spielt (vgl. Manning et al. 2008, S. 177ff.).

Regel	**Beispiel**
SSES $\rightarrow$ SS	caresses $\rightarrow$ caress
IES $\rightarrow$ I	ponies $\rightarrow$ poni
S $\rightarrow$	cats $\rightarrow$ cat
$(m > 1)$ EMENT $\rightarrow$	replacement $\rightarrow$ replac
	cement $\rightarrow$ cement

Tabelle 3.2: Wortkürzungsregeln des Porter Stemmers (Manning et al. 2008, S. 33).

Es existieren verschiedene Vorgehensweisen, die sich hauptsächlich darin unterscheiden, wie und wo ein Wort getrennt wird. Der wohl bekannteste Vertreter davon ist der Porter-Stemmer-Algorithmus von Porter (1980), der in erweiterter Form noch heute zum Einsatz kommt. Der Algorithmus besteht aus fünf sequentiellen Phasen der Wortreduktion. In

jeder Phase werden verschiedene Regeln angewendet, um das Wort weiter zu reduzieren. Beispielregeln werden in Tabelle 3.2 dargestellt (vgl. Manning et al. 2008, S. 33).

Das Hauptproblem von Stemming besteht darin, dass Wörter, die nicht zur selben Stammform gehören oder in verschiedenen Kontexten stehen, auf dieselbe Stammform reduziert werden (*business*, *busy*; *operate*, *operating*). Abhängig von der Sprache kann sich Stemming unterschiedlich gut eignen. So können morphologisch reichere Sprachen (wie Deutsch, Finnisch und Spanisch) wesentlich mehr vom Einsatz eines Stemmers profitieren als die englische Sprache (vgl. Manning et al. 2008, S. 34).

POS-Tagging

Part-of-Speech-Tagging (POS) bezeichnet die automatisierte Wortartmarkierung eines Satzes. Durch die Betrachtung der Wortarten kann die syntaktische Struktur eines Satzes offengelegt und analysiert werden (vgl. Bird et al. 2009, S. 179).

Die grundlegenden Wortarten eines POS-Taggers sind (Penn Treebank[5] POS-Tag in Klammer): Verben (VB), Nomen (NN), Pronomen (PR+DT), Adjektive (JJ), Adverben (RB), Präpositionen (IN), Konjunktionen (CC) und Interjektionen (UH). Darüber hinaus wird eine detailliertere Unterscheidung von Wortarten unternommen (vgl. Alzahrani et al. 2012). Die nachfolgende Tabelle zeigt ein erweitertes Tag-Set für Verben. Ein erweitertes Tag-Set für Verben wird in Tabelle 3.3 gezeigt.

Die Wortartbestimmung eines Wortes ist nicht trivial, da ein und dasselbe Wort je nach Kontext verschiedenen Wortarten zugewiesen werden kann bzw. muss (vgl. Santorini 1990). Zwei Beispielsätze, die das englische Wort *play* einmal als Verb (spielen) und einmal als Nomen (Theaterstück) verwenden[6]:

> *To play poker you need cards.*
>
> To/TO play/VB poker/NN you/PRP need/VBP cards/NNS ./.

> *The play of Shakespeare was nice.*
>
> The/DT play/NN of/IN Shakespeare/NNP was/VBD nice/JJ ./.

Es gibt verschiedene Vorgehensweisen, die sich an erster Stelle darin unterscheiden, ob sie auf einen manuell getaggten Korpus zurückgreifen (überwacht) oder das POS-Tagging automatisiert anstreben (unüberwacht).

[5] Ein Projekt, das sich mit textueller Annotation natürlicher Sprache auseinandersetzt. Online verfügbar unter `http://www.cis.upenn.edu/~treebank/`, abgerufen am 04.09.2014

[6] Mit einem Brill-Tagger getaggt. Online verfügbar unter `http://cst.dk/online/pos_tagger/uk/`, abgerufen am 04.09.2014

Tag	Beschreibung	Beispiel
VB	verb, base form	play
VBZ	verb, 3rd person singular present	he plays
VBP	verb, non-3rd person singular present	I play
VBD	verb, past tense	she played
VBN	verb, past participle	a forgotten game
VBG	verb, gerund or present participle	playing is fun

Tabelle 3.3: Erweitertes Penn Treebank POS-Tagset für Verben (vgl. Santorini 1990).

Beispiele für überwachte Varianten sind Hidden Markov Models (HMM) und Brill Tagger. HMM werden dafür eingesetzt, die Übergangswahrscheinlichkeiten von POS-N-Grammen eines getaggten Korpus zu verarbeiten. Einem Wort wird jene Wortart zugewiesen, die aufgrund der vorangegangenen Wortarten am wahrscheinlichsten folgt. Ein Brill Tagger bezieht sich hingegen nicht auf Übergangswahrscheinlichkeiten, sondern weist in einem ersten Schritt jedem Wort jene Wortart zu, die diesem am häufigsten zugewiesen wurde (z.B. das Wort *play* wird als Nomen getaggt) und erstellt in einem zweiten Schritt Regeln, anhand welcher falsch getaggte Wörter ausgebessert werden (z.B. sofern *to* vor einem als Nomen getaggten Wort steht, wird es in ein Verb umgewandelt: *to play poker*). Der Ground Truth für beide Schritte ist wiederum ein vorher getaggter Korpus (vgl. Manning & Schütze 1999, S. 341ff.).

Vor allem aufgrund von Wörtern, die innerhalb desselben Satzes zwei verschiedenen Wortarten zugeordnet werden können, lässt sich eine bestimmte Fehlerrate nicht vermeiden (vgl. Juola 2007, S. 265). Es folgt ein Beispielsatz, dessen Wort *entertaining* sowohl als Verb (die Herzogin unterhielt) als auch als Adjektiv interpretiert werden kann (die Herzogin war unterhaltsam)[7]:

> *The duchess was entertaining last night.*
>
> The/DT duchess/NN was/VBD entertaining/**VBG** last/JJ night/NN ./.
>
> The/DT duchess/NN was/VBD entertaining/**JJ** last/JJ night/NN ./.

Nachdem ein Überblick über mögliche textuelle Features und deren Extraktion geboten wurde, werden im nächsten Abschnitt Clusterverfahren vorgestellt, die die Daten anhand der extrahierten Features analysieren.

[7] Beispiel aus Santorini 1990, S. 32.

3.2 Clusterverfahren

Wie in Abschnitt 2.3 bereits angeführt, gehören Clusterverfahren zu den unüberwachten ML-Methoden. Das Ziel besteht darin, Strukturen innerhalb der Daten zu erkennen und ähnliche Datenpunkte zu Clustern zusammenzufassen. Daten innerhalb eines Clusters sind demnach ähnlicher zueinander als Daten die nicht demselben Cluster angehören. Es existieren verschiedene Verfahren, die sich darin unterscheiden, nach welchen Kriterien und mit welchen Methoden Datenpunkte zusammengefasst werden. In diesem Abschnitt werden einige ausgewählte Verfahren beschrieben und deren Stärken und Schwächen aufgezeigt. Die jeweiligen Abbildungen (Abb. 3.3, 3.4 und 3.5) beziehen sich auf eine Datenmenge im zweidimensionalen Raum, die aus drei Normalverteilungen mit den Mittelpunktvektoren $\mu_{(-12,25)}$, $\mu_{(0,0)}$ und $\mu_{(13,5)}$ generiert wurde.

3.2.1 Hierarchisches Clustering

Das Ergebnis eines flachen Clusterings ist eine Aufteilung der Datenmenge in k verschiedene Cluster. Jedes Element x_i aus der Datenmenge wird genau einem Cluster aus der Clustermenge zugeordnet. Hierarchische Clusterverfahren sind dadurch gekennzeichnet, dass sie keine flachen Cluster, sondern eine hierarchische Struktur bilden (vgl. Duda et al. 2001, S. 550f.).

Die schrittweise Bildung der Cluster wird festgehalten und kann in Form eines Dendrogrammes (binärer Baum) dargestellt werden, wobei zwischen divisiven (top-down) und agglomerativen (bottom-up) Verfahren unterschieden wird. Divisive Verfahren gehen von einem einzigen Cluster aus, der die gesamte Datenmenge zusammenfasst, und unterteilen die Daten schrittweise in weitere Cluster, bis jedes Datenelement einen eigenen Cluster darstellt.

Agglomerative Verfahren hingegen teilen jedem Datenelement einen eigenen Cluster zu und fassen die Cluster schrittweise zusammen, bis die gesamte Datenmenge demselben Cluster angehört (vgl. Tan et al. 2005, S. 515f.).

Divisive Verfahren gelten als aufwändiger und ineffizienter, weshalb agglomerativen Verfahren mehr Beachtung geschenkt wird und sich auch die vorliegende Arbeit auf diese Vorgehensweise beschränkt (vgl. Webb 2002, S. 363).

Agglomerative Verfahren fassen jene zwei Cluster zusammen, die sich nach einer bestimmten Metrik (z.B. Euklidische Distanz) am nächsten sind. Es existieren verschiedene Fusionierungsalgorithmen (Linkage-Typen), nach denen die Distanz zwischen Clustern und damit die Art der Fusionierung definiert wird. Häufig erwähnte Methoden sind Single-, Complete- und Average-Linkage sowie die Methode von Ward (vgl. Hastie et al. 2009, S. 523; Witten et al. 2011, S. 275f.).

Single Linkage definiert die Distanz als die minimale Distanz zwischen den Elementen der Cluster. Die Ähnlichkeit der Cluster A, B ist demnach abhängig von den zwei Punkten i, j, die sich am nächsten sind und unabhängig davon, wie die restlichen Punkte der Cluster verteilt sind:

$$d_{AB} = \min_{i \in A, j \in B} d_{ij}$$

Complete Linkage hingegen verwendet die maximale Distanz zwischen den Clusterelementen. Wiederum nur abhängig von zwei Punkten werden nach dieser Vorgehensweise jene Cluster fusioniert, deren entferntesten Elemente sich am nächsten sind:

$$d_{AB} = \max_{i \in A, j \in B} d_{ij}$$

Average Linkage berücksichtigt alle Elemente der Cluster und definiert die Distanz als die durchschnittliche Distanz selbiger. Jene Cluster, deren Elemente zusammen die geringste durchschnittliche Distanz zueinander aufweisen, werden fusioniert:

$$d_{AB} = \frac{1}{n} \sum_{i,j \in A \cup B, i \neq j}^{n} d_{ij}$$

Zu den ursprünglichen Varianten schlägt Ward (1963) die Minimierung einer sogenannten *Objective Function* vor. Diese Funktion könnte z.B. die Summe der quadratischen Abweichung oder – wie in der nachfolgenden Berechnung gezeigt – die Varianz sein. Es werden jene zwei Cluster fusioniert, deren vereinte Elemente die geringste Varianz aufweisen:

$$d_{AB} = \frac{1}{n} \sum_{i=1, x \in A \cup B}^{n} (x_i - \mu)^2, \qquad \text{wobei } \mu = \frac{1}{n} \sum_{i=1, x \in A \cup B}^{n} x_i$$

Sofern die Datenmenge kompakte und gut abgetrennte Gruppen aufweist, produzieren alle Fusionierungsalgorithmen ähnliche Cluster – ist dies nicht der Fall, können die verschiedenen Verfahren auf sehr unterschiedliche Ergebnisse kommen (vgl. Hastie et al. 2009, S. 523f.; Witten et al. 2011, S. 276).

Abbildung 3.3 stellt die hierarchische Struktur in Form eines Dendrogramms dar. Die Datenelemente auf der x-Achse werden der Fusionierungsdistanz auf der y-Achse gegenübergestellt. Abbildung 3.4 und 3.5 zeigen die zu Grunde liegenden Daten im zweidimensionalen Raum und erlauben eine bessere Interpretation der hierarchischen Struktur des Dendrogramms. Es sind drei relativ gut abgetrennte Cluster erkennbar, deren Untergruppen am Beginn noch sehr nah aneinander liegen und sich zunehmend voneinander entfernen. An der Struktur lässt sich erkennen, dass sich die zwei linken Cluster vom Aufbau sehr ähneln (beide weisen eine maximale Distanz von etwa 3 auf) und sich vom

dritten Cluster unterscheiden. Bevor alle drei Punktwolken zu einem Cluster fusioniert werden (Distanz 30), werden die beiden Cluster rechts an der Distanz 15 zusammengefasst.

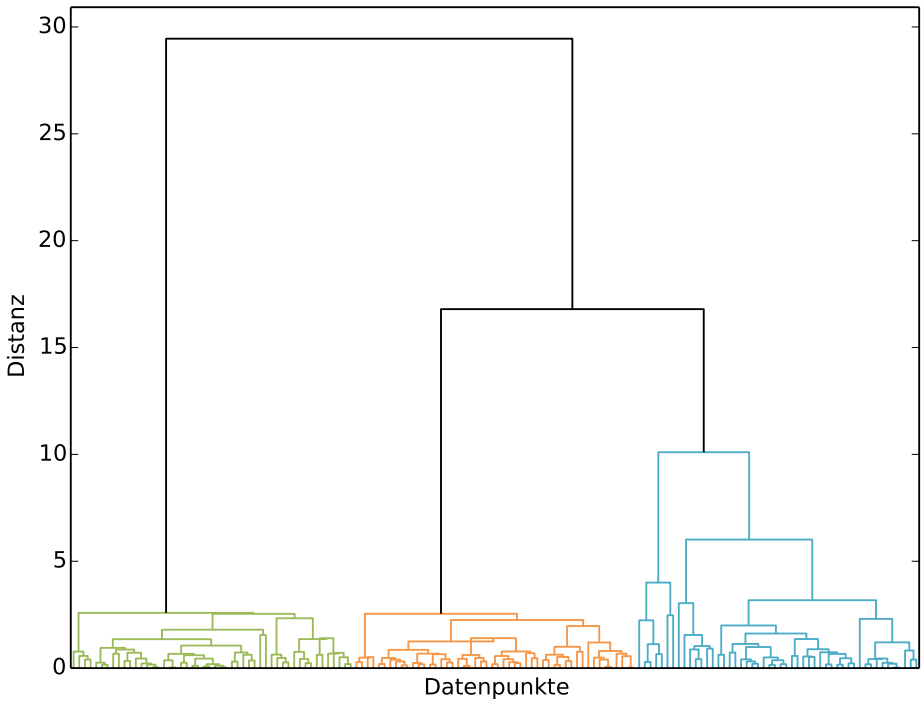


Abbildung 3.3: Dendrogram eines hierarchischen Clusterings mit Average Linkage: die drei eingefärbten Unterbäume stellen eine mögliche Clusteraufteilung dar.

Die Wahl der Parameter gestaltet sich je nach Vorgehensweise verschieden. Nachdem die Wahl der Metrik und des Fusionierungsalgorithmus getroffen wurde, kann entweder der gesamte Baum berechnet werden oder ein Grenzwert für die Distanz definiert werden, ab der die Agglomeration stoppen soll. Die Berechnung des gesamten Baumes kann dafür geeignet sein, Aufschluss über die Struktur der Daten zu gewinnen und sich aufgrund dessen für ein sinnvoll erscheinendes Clustering zu entscheiden.

Probleme und Schwächen ergeben sich vor allem aufgrund des gewählten Linkage-Verfahrens. Single- und Complete-Linkage sind sehr anfällig gegen Ausreißer (vgl. Duda et al. 2001, S. 554f.). So kann Single-Linkage zum sogenannten Chain-Effekt führen und an sich relativ gut separierte Datenmengen durch die Existenz eines einzelnen Punktes, der als Brücke zwischen den beiden Clustern dient, zu einem Cluster verbinden. Complete-Linkage kann die Fusionierung jener Cluster bevorzugen, deren entfernteste Punkte zwar die geringste Distanz aufweisen, in der gesamten Menge jedoch weiter voneinander entfernt sind als alternative Paarungen. Average-Linkage und Wards Methode neigen dazu, sphärische Clusterformen zu generieren (vgl. Tufféry 2011, S. 257).

Insgesamt kann festgehalten werden, dass hierarchische Verfahren nicht abhängig von einer Start-Initialisierung oder einem Start-Seeding sind[8] (vgl. Hastie et al. 2009, S 520). Die Frage nach der Anzahl der zu generierenden Cluster wird schlichtweg umgangen. Stattdessen wird eine hierarchische Struktur der Daten gebildet, aus der zusätzliche Informationen gewonnen werden können – z.B. Clustering von Krebsarten und Gen-Clustering in der Medizin (vgl. Hastie et al. 2009, S. 525f.), Clustering von Tierarten in der Artenforschung (vgl. Witten et al. 2011, S. 276f.) etc.

3.2.2 Gaußsche-Mixture-Models

Die Grundidee verteilungsbasierter Clusterverfahren besteht darin, die Datenmenge anhand einer Verteilungsfunktion bzw. einer Summe gewichteter Verteilungsfunktionen (sog. Mischverteilungen oder Mixture Models) zu modellieren. Im Sinne der Clusterbildung beschreiben Gaußsche-Mixture-Models (GMM) die Datenmenge als eine Menge von Clustern, die jeweils einer multivariaten Normalverteilung mit bestimmtem Mittelpunktvektor μ_i und Kovarianzmatrix Σ_i zugrunde liegen (vgl. Hastie et al. 2009, S. 463). Die Verteilungen eines GMM werden als Komponenten bezeichnet, deren Anzahl von vornherein festgelegt (z.B. Expectation Maximization) oder als obere Schranke definiert werden kann (z.B. Dirichlet-Prozess).

Die Wahrscheinlichkeit eines D-dimensionalen Datenvektors x unter einem GMM kann also als gewichtete Summe von M normalverteilten Komponenten wie folgt definiert werden (vgl. Reynolds 2008):

$$p(x|\lambda) = \sum_{i=1}^{M} w_i g(x|\mu_i, \Sigma_i),$$

wobei w_i das Gewicht und $g(x|\mu_i, \Sigma_i)$ die Dichtefunktion der Gauß-Komponente ($i = 1, ..., M$) ist:

$$g(x|\mu_i, \Sigma_i) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} e^{-\frac{1}{2\Sigma_i}(x-\mu_i)^T(x-\mu_i)}$$

Die Gewichte ergeben in Summe 1 ($\sum_{i=1}^{M} w_i{=}1$) und die Parameter eines GMM werden mit λ zusammengefasst:

$$\lambda = w_i, \mu_i, \Sigma_i \qquad i = 1, ..., M$$

[8] Sofern es keine identischen Distanzen zwischen den Datenelementen gibt, gilt das auch für die Sortierung. Ansonsten wird eine von n gleichwertigen Fusionierungen gewählt, wodurch das Verfahren abhängig von der Sortierung der Daten ist (vgl. Hastie et al. 2009, S. 525).

Das GMM wird oft als weiches Clusterverfahren bezeichnet, da die Zuordnung der Datenelemente zu einem Cluster nicht exklusiv ist. Stattdessen gehört ein Datenpunkt mit einer gewissen Wahrscheinlichkeit zu jeder Komponente des Mixture-Models. Um die Daten eindeutig zu labeln, wird ein Datenpunkt jener Komponente (Cluster) zugewiesen, die die höchste Wahrscheinlichkeit für diesen Punkt aufweist (vgl. Hastie et al. 2009, S. 463; McLachlan et al. 2003).

In Abbildung 3.4 wird ein GMM mit drei Komponenten und deren Gewichte bzw. Dichteverteilungen in farbigen Kurven dargestellt. Ein theoretischer Datenpunkt an der Position $x_{(0,25)}$ stammt laut den geschätzten Dichteverteilungskurven der drei Gauß-Komponenten mit höherer Wahrscheinlichkeit von der Komponente oben links ($\mu_{-12,25}$) und mit jeweils niedrigerer Wahrscheinlichkeit von den unteren zwei Komponenten. Ein theoretischer Datenpunkt an der Position $x_{(0,10)}$ könnte hingegen mit einer etwa gleich hohen Wahrscheinlichkeit zu allen drei Komponenten zugeordnet werden.

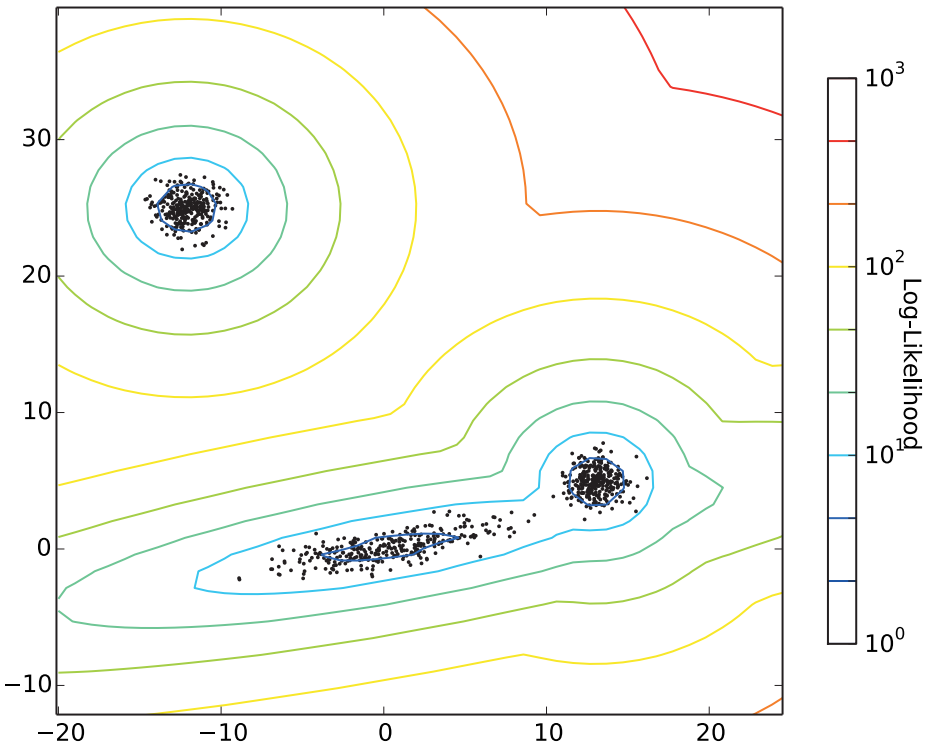


Abbildung 3.4: Dichteverteilungskurven eines GMM mit drei Komponenten.

Die Parameter der Normalverteilungen (μ_i, Σ_i) sind unbekannt und können anhand des Expectation-Maximization-Algorithmus (EM) (Dempster et al. 1977) iterativ ermittelt werden. Nachdem die Anzahl der Komponenten festgelegt wurde, werden die Parameter der Gauß-Komponenten initialisiert. Für die Mittelpunktvektoren μ_i werden zufällige

oder heuristisch ermittelte Punkte aus der Datenmenge gewählt. Die Kovarianzmatrizen Σ_i können mit Einheitsmatrizen initialisiert werden (vgl. Pedregosa et al. 2011).

Der EM-Algorithmus besteht anschließend aus zwei Schritten, die iteriert werden. Im E-Schritt (Expectation) werden den Datenpunkten Gewichte zugeordnet, die auf den Dichteverteilungen der Gauß-Komponenten beruhen und in Summe 1 ergeben. Datenpunkte, die nahe am Mittelpunktvektor μ_i einer Komponente liegen, bekommen für diese Komponente ein höheres Gewicht (gegen 1 strebend) als für alle anderen Komponenten (gegen 0 strebend). Im M-Schritt (Maximization) werden die Parameter (μ_i, Σ_i) aufgrund der vorher ermittelten Gewichte neu berechnet. Es wird versucht, die Verteilungen so zu modellieren, dass die Wahrscheinlichkeit in Bezug auf die Datenpunkte maximiert wird (vgl. Hastie et al. 2009, S. 463). Diese beiden Schritte werden iteriert, bis eine Konvergenz von $p(X|\lambda)$ erreicht wird. Die Wahrscheinlichkeit einer Datenmenge unter einem GMM kann als Produkt der Wahrscheinlichkeiten der Datenpunkte $p(x_i|\lambda)$ definiert werden, wobei X die Menge von T Datenpunkten darstellt (vgl. Reynolds 2008):

$$p(X|\lambda) = \prod_{t=1}^{T} p(x_n|\lambda)$$

Der EM-Algorithmus kann unter bestimmten Voraussetzungen (abhängig von den Daten und der Initialisierung) an sogenannten Sattelpunkten festsitzen, die weder einem lokalen noch dem globalen Optimum entsprechen (vgl. Collins 1997). Sofern das Verfahren konvergiert, wird nur ein lokales Maximum garantiert. Deshalb wird der Prozess mit verschiedenen Start-Parametern wiederholt, um die Chancen für das Erreichen des globalen Optimums zu erhöhen (vgl. Witten et al. 2011, S. 288).

Der Dirichlet-Prozess (DP) geht auf Ferguson (1973) zurück und stellt die Generierung einer Verteilung von Verteilungen dar, die durch einen Konzentrationsparameter α gesteuert wird (vgl. Blei & Jordan 2006). Da die Anzahl der Komponenten grundsätzlich unbeschränkt ist, wird das Modell auch als Infinite Mixture Model bezeichnet. Implementierungen wie Pedregosa et al. (2011) grenzen die Komponentenanzahl jedoch durch eine obere Schranke ab.

Der DP wird oft durch ein modifiziertes Pólya-Urnenmodell erläutert. Eine Urne ist mit α schwarzen Kugeln gefüllt (α ist eine positive reelle Zahl). Bei jeder Iteration wird eine Kugel aus der Urne genommen. Sofern es eine schwarze Kugel ist, wird eine Kugel einer neuen Farbe zusammen mit der schwarzen Kugel zurück in die Urne gelegt. Wird eine nicht-schwarze Kugel gezogen, wird die gezogene Kugel mit einer weiteren Kugel derselben Farbe zurück in die Urne gelegt. Die Anzahl der auf diese Weise generierten Farben ist abhängig vom Konzentrationsparameter α und von der Anzahl der Ziehungen (Iterationen). Ein kleines α (z.B. ≤ 1) resultiert in einer niedrigen Anzahl an Farben, ein größeres α (z.B. 100) in einer höheren Anzahl. Durch weitere Ziehungen können zusätzliche Farben (unbegrenzt viele) generiert werden (vgl. Frigyik et al. 2010).

Die verschiedenen Farben im beschriebenen Urnen-Beispiel sind im Sinne eines GMM Normalverteilungen mit unterschiedlichen Parametern (μ_i, Σ_i). Eine wesentlich ausführlichere und formalere Erläuterung des DP wird in Frigyik et al. (2010) geboten. Blei & Jordan (2006) und Rasmussen (2000) stellen eine Verbindung zwischen DP und Mixture Models bzw. GMM her.

Abgesehen von der Entscheidung, die Clusteranzahl festzulegen oder nicht, richten sich die Parameter eines GMM nach der gewünschten Clusterform. Abhängig von der Art der Kovarianzmatrizen (voll oder diagonal bzw. getrennt oder einheitlich für alle Komponenten) sind verschiedene Clusterformen möglich (vgl. Murphy 1998; Reynolds 2008). Zudem kann in Implementierungen wie Pedregosa et al. (2011) festgelegt werden, welche Parameter (Gewichte, Mittelpunktvektoren und Kovarianzmatrizen) iterativ angepasst werden sollen.

Eine offensichtliche Voraussetzung für das Clustern mit GMM ist die Form der Cluster, die einer Normalverteilung entsprechen muss. GMM beschreiben die Daten anhand eines Modells (eine Kombinationen von Normalverteilungen). Aufgrund dieses Modellcharakters ist das direkte Vergleichen unterschiedlicher Modelle möglich (vgl. Manning et al. 2008, S. 368): Wie hoch ist die Wahrscheinlichkeit, dass eine Datenmenge von einem Modell repräsentiert wird? Welches Modell liefert eine bessere Beschreibung der Daten? Zu beachten ist dabei das Problem der Überanpassung der Daten (overfitting). Mit steigender Anzahl an Komponenten werden einzelne Datenpunkte besser repräsentiert (im Extremfall für jeden Datenpunkt eine Komponente). Das bedeutet jedoch nicht zwangsläufig, dass auch das Clustering der Daten entsprechend verbessert wird (vgl. Witten et al. 2011, S. 290f.).

3.2.3 DBSCAN

DBSCAN verfolgt einen dichtebasierten Ansatz und kann in Anbetracht gängiger Clusterverfahren als relativ aktuell bezeichnet werden. Das Verfahren geht auf Ester et al. (1996) zurück, deren Erläuterungen für die folgende Beschreibung herangezogen wurden.

Die Grundannahme besteht darin, dass Cluster Regionen sind, die im Vergleich zu den umliegenden Gebieten eine höhere Dichte an Datenpunkten aufweisen. Zusätzlich werden sogenannte Rausch-Bereiche (noise areas) in der Modellierung berücksichtigt. Die Dichte solcher Rausch-Bereiche ist demnach geringer als die Dichte in den Cluster-Regionen.

Um die Daten nach dieser Grundannahme zu modellieren, werden Dichteregionen definiert. Für jeden Punkt eines Clusters muss sich innerhalb eines vorgegebenen Radius ϵ eine minimale Anzahl an Punkten *minPts* befinden. Die Dichte in der umliegenden Region eines Punktes muss also einen bestimmten Grenzwert überschreiten.

In Abbildung 3.5 wird das Ergebnis eines Clusterings mit DBSCAN dargestellt. Die Anzahl der Cluster wird automatisch bestimmt bzw. hängt von den Parametern ϵ und *minPts* ab. Mit niedrigerem ϵ und höheren *minPts* sind dichtere Punktwolken für die Clusterbildung

nötig. Die beiden unteren Punktwolken würden mit $\epsilon = 2.0$ und $minPts = 5$ zu einem Cluster vereint werden.

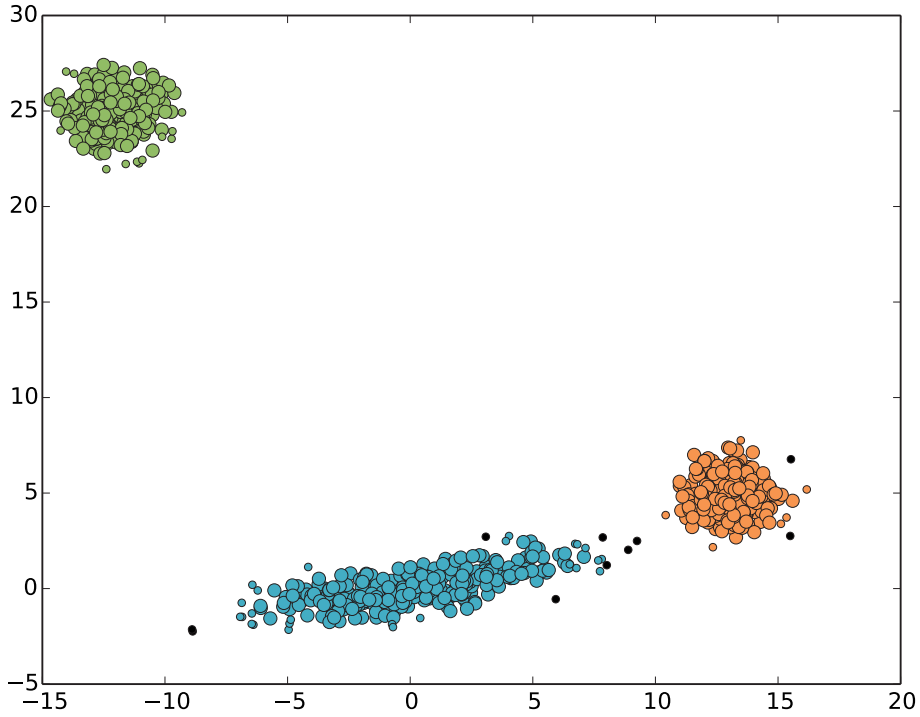


Abbildung 3.5: DBSCAN ($\epsilon = 1.0$, $minPts = 10$): drei Cluster gefunden. Randpunkte werden im Vergleich zu den Kernpunkten kleiner dargestellt. Punkte in schwarz sind Rauschpunkte.

Konkret wird zwischen drei Arten von Punkten unterschieden: Kernpunkte, Randpunkte und Rauschpunkte. Kernpunkte sind jene Punkte, die das Dichtekriterium (Anzahl der Punkte im Radius $\epsilon \geq minPts$) erfüllen. Randpunkte sind Punkte, die im ϵ-Bereich eines Kernpunktes liegen, die minimale Anzahl an Punkten im eigenen ϵ-Bereich jedoch nicht erreichen. Rauschpunkte gehören zu jenen Punkten, die weder das Kriterium eines Kernpunktes (Dichtekriterium) noch das eines Randpunktes (Erreichbarkeit durch Kernpunkte) erfüllen.

Folgende Relationen werden zwischen den Punkten definiert:

- direkt dichte-erreichbar: Ein Punkt p ist von einem Punkt q direkt dichte-erreichbar, sofern p sich im ϵ-Bereich von Punkt q befindet und q ein Kernpunkt ist.
- dichte-erreichbar: Ein Punkt p ist von einem Punkt q dichte-erreichbar, sofern eine Folge von Punkten $p_1, ..., p_n$, $p_1 = q$, $p_n = p$ existiert, sodass p_{i+1} direkt dichte-erreichbar von p_i ist.
- dichte-verbunden: Ein Punkt p ist mit einem Punkt q dichte-verbunden, sofern ein Punkt o existiert, sodass p und q dichte-erreichbar von o sind.

Eine Menge dichte-erreichbarer und dichte-verbundener Punkte wird jeweils zu einem Cluster zusammengefasst. Dabei deckt die Dichte-Verbundenheit die Relation zwischen Randpunkten eines Clusters ab, die an sich nicht dichte-erreichbar sind. Der Algorithmus beginnt mit einem zufällig gewählten Punkt p und verarbeitet alle dichte-erreichbaren Punkte von p. Sofern p ein Kernpunkt ist, wird ein neuer Cluster gebildet und allen dichte-verbundenen Punkten rekursiv zugewiesen. Wenn p kein Kernpunkt ist, wird ein weiterer Punkt zufällig gewählt. Punkte, die am Ende keinem Cluster zugewiesen werden konnten, sind Rauschpunkte.

Die Stärke von DBSCAN besteht darin, auch nicht-konvexe Clusterformen ohne Vorgabe der Clusteranzahl modellieren zu können. Ein weiteres Merkmal ist die Berücksichtigung von Punkten, die keinem Cluster zugewiesen werden können, da sie das Dichtekriterium nicht erfüllen (vgl. Tan et al. 2005, S. 530). Schwierigkeiten ergeben sich bei der Bestimmung der Dichteparameter, vor allem wenn die Datenmenge oder einzelne Cluster aus sehr unterschiedlich dichten Regionen bestehen (vgl. Kriegel et al. 2011).

3.2.4 Sonstige Verfahren

Zu den bereits vorgestellten Clusterverfahren wird die Funktionsweise folgender drei Verfahren umrissen: K-Means, Spectral Clustering und Affinity Propagation.

K-Means

Der K-Means-Algorithmus (Lloyd 1957) gehört zu den populärsten flachen Clusterverfahren (vgl. Hastie et al. 2009, S.509). Das Ziel besteht darin, mit vorgegebener Clusteranzahl k die Varianz innerhalb der Cluster iterativ zu minimieren. Ähnlich dem EM-Algorithmus werden zufällige Startpunkte gewählt (Mittelpunktvektoren μ_i), jeder Datenpunkt den am nächsten gelegenen μ_i zugeordnet (k-Schritt) und aus den daraus entstehenden Clustern die Mittelpunktvektoren μ_i neu berechnet (M-Schritt). Diese zwei Schritte werden iterativ durchgeführt, bis die Mittelpunktvektoren konvergieren (vgl. Manning et al. 2008, S.360).

Im Unterschied zu EM tendiert K-Means dazu, gleich große Cluster zu generieren. Da das Verfahren stark von Ausreißern beeinflusst werden kann, und nur das Erreichen eines lokalen Optimums garantiert wird, empfiehlt es sich, K-Means mehrmals mit unterschiedlichen Startpunkten durchzuführen und die Lösung mit der kleinsten Varianz zu wählen (vgl. Hastie et al. 2009, S. 510).

Spectral Clustering

Spectral Clustering geht auf Donath & Hoffman (1973) zurück und wird als Graphen-Partitionierungsproblem formuliert (vgl. Luxburg 2007). Um die Datenpunkte in

Graphen-Form zu bringen, wird eine Ähnlichkeitsmatrix berechnet (z.B. mittels Gauß-Kernel). Anhand der Eigenvektoren, die den n-kleinsten Eigenwerten entsprechen, wird anschließend eine Dimensionsreduktion durchgeführt und ein Clustering im reduzierten Vektorraum unternommen (z.B. mit K-Means). Das Ähnlichkeitsmaß, die Anzahl der zu verwendenden Eigenvektoren n und die Clusteranzahl k müssen abhängig von der Datenmenge bestimmt werden (vgl. Hastie et al. 2009, S. 544ff.).

Ein Vorteil von Spectral Clustering liegt darin, dass keine starke Annahme über die Clusterform getroffen wird. Nicht-konvexe oder in sich verschachtelte Cluster (im 2D-Raum z.B. Punkte, die einen Ring innerhalb eines weiteren Ringes mit demselben Mittelpunkt bilden) sind möglich (vgl. Luxburg 2007).

Affinity Propagation

Einen anderen Ansatz verfolgen Frey & Dueck (2007) mit Affinity Propagation. Zwischen Datenpunkt-Paarungen werden Mitteilungen ausgetauscht, die eine Aussage über die Ähnlichkeit der beiden Punkte treffen. Die Datenmenge wird iterativ in sogenannte Exemplar-Punkte unterteilt, die eine Menge von Nichtexemplaren repräsentieren. Ein Cluster besteht aus einem Exemplarpunkt und der Gesamtheit der Punkte, die von ihm vertreten wird. Zu Beginn werden alle Punkte mit einem negativen Präferenz-Wert initialisiert: Punkte mit einem höheren Präferenz-Wert werden mit einer höheren Wahrscheinlichkeit zu Exemplaren gewählt (um keinen Punkt zu bevorzugen kann ein einheitlicher Wert vergeben werden).

Zwei Mitteilungen werden zwischen den Punkten ausgetauscht:

- Zuständigkeit (responsibility) *r(i,k)* von Punkt i nach k: Wie gut ist k geeignet, ein Exemplar von i zu sein (unter Berücksichtigung aller anderen möglichen Exemplare für i)?
- Verfügbarkeit (availability) *a(i,k)* von Punkt k nach i: Wie gut ist i geeignet, von k repräsentiert zu werden (unter Berücksichtigung aller anderen Punkte, die k repräsentieren könnte)?

Die Zuständigkeit wird wie folgt berechnet:

$$r(i,k) = s(i,k) - \max_{k' \neq k}[a(i,k') + s(i,k')]$$

Bei der ersten Iteration wird die Verfügbarkeit mit 0 initialisiert, wodurch $r(i,k)$ die Ähnlichkeit der beiden Punkte $s(i,k)$, vermindert um die maximale Ähnlichkeit zu jedem anderen Punkt, ergibt. Anschließend wird $a(i,k)$ aktualisiert:

$$a(i,k) = \min \left[0, r(k,k) + \sum_{i' \notin (i,k)} \max[0, r(i',k)] \right]$$

Die Verfügbarkeit ist die Summe aller positiven Zuständigkeitswerte (die dem Punkt k zugewiesen wurden) und dem eigenen Zuständigkeitswert $r(k, k)$, der dem initialen Präferenz-Parameter (negativer Wert) für diesen Punkt entspricht (begrenzt durch 0).

Das optimale Set von Exemplaren maximiert die Summe $a(i, k) + r(i, k)$ der gesamten Datenmenge (dabei wird die Summe eines Exemplarpunktes maximiert, wenn $i = k$, er also von sich selbst repräsentiert wird). Das Verfahren wird so lange durchgeführt, bis die Unterteilung der Datenpunkte konvergiert. Die Anzahl der Exemplar-Punkte entspricht der Anzahl der Cluster und ist vom Präferenz-Parameter und der Zusammensetzung der Daten abhängig.

Im Unterschied zu Verfahren, die mit zufällig gewählten Start-Punkten arbeiten und diese Punkte iterativ anpassen, betrachtet Affinity Propagation alle Datenpunkte bei jeder Iteration als potenzielle Exemplar-Kandidaten. Vorteile von Affinity Propagation ergeben sich durch die Möglichkeit, auch nicht-metrische Ähnlichkeitsmaße zu verwenden, die nicht symmetrisch $s(i, k) \neq s(k, i)$ sein müssen. Das Verfahren kann auch mit unvollständigen Daten arbeiten (fehlende Ähnlichkeiten einzelner Datenpunkte zueinander).

Nachdem die Funktionsweise verschiedener Clusterverfahren beschrieben wurde, geht der folgende Abschnitt auf die Evaluierung von Clustern über.

3.3 Clusterevaluierung

Wie gut eine Datenmenge in Cluster aufgeteilt wurde, hängt von der Definition eines Clusters ab. Es ist nicht eindeutig bestimmt, wie sich ein Cluster zusammensetzt oder abgrenzt (vgl. Webb 2002, S. 415):

> *Different clustering methods may produce different classifications. How do we know whether the structure is a property of the data set and not imposed by the particular method that we have chosen?* (Webb, 2002, S. 396)

Einige Kriterien bzw. Maße werden vorgestellt. Zunächst wird die Evaluierung von Klassifikatoren beschrieben.

3.3.1 Evaluierung von Klassifikatoren

Das gelernte Modell eines Klassifikators kann auf Basis der Ground-Truth-Daten evaluiert werden – die tatsächlichen Klassenlabels sind bekannt. Häufig benutzte Evaluierungsmaße sind Präzision (Precision), Trefferquote (Recall) und eine Kombination der beiden: das F-Maß (vgl. Manning & Schütze 1999, S. 536f.).

Die Präzision eines Klassifikators entspricht dem Verhältnis von korrekt zugeordneten Datenpunkten (*true positive* t_p) zur Gesamtheit zugeordneter Datenpunkte einer Klasse (einschließlich jener Datenpunkte, die dieser Klasse zugeordnet wurden, aber nicht

dieser Klasse angehören: *false positive* f_p). Wie viele der Datenpunkte, die einer Klasse zugewiesen wurden, stammen tatsächlich von dieser Klasse?

$$\text{Präzision } P = \frac{t_p}{t_p + f_p}$$

Die Trefferquote entspricht dem Verhältnis von korrekt zugeordneten Datenpunkten (t_p) zur tatsächlichen Gesamtheit zugehöriger Punkte einer Klasse (einschließlich jener Datenpunkte, die dieser Klasse angehören, fälschlicherweise jedoch einer anderen Klasse zugeordnet wurden: *false negative* f_n). Wie viele der tatsächlichen Datenpunkte einer Klasse wurden gefunden?

$$\text{Trefferquote } T = \frac{t_p}{t_p + f_n}$$

Eine Kombination von Präzision und Trefferquote wird durch das F-Maß ausgedrückt. Der Parameter α steuert die Gewichtung von Präzision und Trefferquote (mit $\alpha = 0.5$ sind beide Maße gleich gewichtet).

$$\text{F-Maß } F_\alpha = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{T}}$$

Die Maße werden anhand einer Wahrheitsmatrix (auch Kontingenztabelle) verständlicher. Tabelle 3.4 zeigt das Ergebnis einer fiktiven Klassifizierung drei verschiedener Vogelarten (Kakadus, Wellensittiche und Tukane). Die Menge besteht aus jeweils 10 Exemplaren jeder Vogelart, wovon 11 als Kakadu, 9 als Wellensittich und 10 als Tukan klassifiziert wurden. Die Präzision der Kakadu-Klasse berechnet sich durch $G = 8/(8+3) = 0.727$, die Trefferquote durch $T = 8/(8+2) = 0.8$ und das F-Maß durch $F_{0.5} = \frac{1}{0.5\frac{1}{0.727} + 0.5\frac{1}{0.8}} = 0.762$. Die Maße der gesamten Klassifizierung können durch das gewichtete arithmetische Mittel der entsprechenden Klassen-Werte berechnet werden.

		klassifiziert als		
		Kakadu	Wellensittich	Tukan
tatsächliche Klasse	Kakadu	8	2	0
	Wellensittich	3	7	0
	Tukan	0	0	10

Tabelle 3.4: Kontingenztabelle einer fiktiven Klassifizierung mit drei Klassen. Die Diagonale zeigt die Anzahl der tatsächlich korrekt klassifizierten Tiere.

Für eine aussagekräftige Evaluierung werden die Daten getrennt, mit denen ein Klassifikator trainiert und getestet wird. Sofern die Datenmenge limitiert ist, bietet sich die stratifizierte Kreuzvalidierung an. Die Daten werden dabei in k-Teilmengen unterteilt

(z.B. $k = 10$), die jeweils ungefähr der Klassenverteilung der gesamten Menge entsprechen. Anschließend dient jede dieser Teilmengen als Test-Set und die restliche Menge als Trainings-Set. Um die Präzision der Kreuzvalidierung zu erhöhen, wird dieses Verfahren k-mal wiederholt und die Ergebnisse daraus gemittelt (vgl. Witten et al. 2011, S. 152ff.).

Im Gegensatz zum Klassifizieren wird die Information der Klassenlabels beim Clustern nicht verwendet. Welche Kriterien es für die Evaluierung von Clustern gibt, wird im nächsten Abschnitt behandelt.

3.3.2 Clusterevaluierungskriterien

Die Clusteraufteilung einer Datenmenge ist vom verwendeten Verfahren abhängig – unterschiedliche Clusterverfahren können dieselben Daten unterschiedlich strukturieren. Nahezu alle Clusteralgorithmen finden Cluster in einer Datenmenge, auch wenn keine natürlichen Clusterstrukturen vorhanden sind. Aus diesem Grund ist es sinnvoll, die Güte einer Clusteraufteilung zu evaluieren (vgl. Tan et al. 2005, S. 532).

Tan et al. (2005) unterscheiden zwischen den folgenden drei Arten von Evaluierungsmaßen.

Extern: Es wird bewertet, inwiefern die gefundene Clusterstruktur eines Clusterverfahrens mit einer externen Struktur übereinstimmt. Beispielsweise ist die tatsächliche Klassenaufteilung gegeben und anhand des bereits vorgestellten F-Maßes wird die ermittelte Clusteraufteilung bewertet.

Intern: Die Güte einer Clusterstruktur wird anhand interner Kriterien bewertet. Ein Beispiel dafür ist die Summe der quadratischen Abweichung (SQA), die beschreibt, inwieweit die Elemente eines Clusters von ihrem Mittelwert abweichen. Interne Evaluierungsmaße werden oft in zwei weitere Gruppen unterteilt: Cluster-Kohäsion, die feststellt, wie stark ein Cluster zusammenhängt (kleine Distanzen innerhalb der Cluster führen zu einer starken Kohäsion) und Cluster-Separation, die feststellt, wie gut die Cluster voneinander getrennt sind (große Distanzen zwischen den Clustern resultieren in einer hohen Separation).

Relativ: Externe oder interne Maße werden dafür verwendet, die Ergebnisse von Clusterverfahren zu vergleichen. Aus diesem Grund sind relative Evaluierungsmaße keine tatsächlichen Maße, sondern eher eine spezielle Art der Anwendung von Maßen. Beispielsweise werden zwei K-Means-Aufteilungen anhand der SQA oder dem F-Maß verglichen.

Die offensichtliche Voraussetzung für die Anwendung externer Maße ist das Vorhandensein externer Informationen. Diese Informationen sind aber nicht auf die tatsächliche Klassenstruktur beschränkt. Im Sinne eines relativen Evaluierungsmaßes kann auch das Labeling eines alternativen Clusterverfahrens als Referenz herangezogen werden (vgl. Tan et al. 2005, S. 549).

Interne Maße hängen stark von der Art der Clusterbildung ab. So ist die SQA für Verfahren wie K-Means besser geeignet als für dichtebasierte Verfahren wie DBSCAN: K-Means neigt dazu sphärische Cluster zu produzieren, die die Varianz innerhalb der Cluster minimieren (vgl. Webb 2002, S. 415).

Relative Maße können unter anderem dazu verwendet werden, die Clusteranzahl zu bestimmen. Indem Clusterings mit unterschiedlicher Clusteranzahl verglichen werden, entscheidet man sich für die Clusteranzahl, die ein Optimum darstellt. Beispielsweise wird die SQA mit der Clusteranzahl k verglichen: Es wird jenes k gewählt, das die SQA minimiert und ab dem der Qualitätsunterschied zu höheren Werten von k einen festgelegten Grenzwert nicht überschreitet – die Qualität sich also nicht mehr stark verbessert (vgl. Tan et al. 2005, S. 546f.).

Nachdem grundsätzliche Evaluierungskriterien vorgestellt wurden, werden im nächsten Abschnitt konkrete Evaluierungsmaße beschrieben.

3.3.3 Clusterevaluierungsmaße

Viele der internen Evaluierungsmaße sind Formen der Summe der quadratischen Abweichung:

$$\text{SQA} = \sum_{i=1}^{c} \sum_{x \in C_i} (x - \mu_i)^2$$

Der quadratische Abstand der Datenpunkte x zu dem Mittelwertvektor μ_i ihres Clusters wird aufsummiert. Der Wert der SQA hängt von der Clusteranzahl und von der Art der Clusterbildung ab. Die optimale Aufteilung ist durch die Minimierung der SQA definiert und wird erreicht, wenn die Datenmenge in kompakte Cluster aufgeteilt wird. Weitere Maße, die die Kompaktheit der Cluster beschreiben, sind die Varianz innerhalb der Cluster, der Abstand zum Cluster-Median und der maximale Abstand zwischen Punkten eines Clusters (vgl. Duda et al. 2001, S. 542ff.).

Die Separation der Cluster kann mit dem Abstand der Clustermittelpunkte (z.B. Mittelwertvektoren) zum gesamten Mittelwertvektor der Datenmenge gemessen werden (vgl. Tan et al. 2005, S. 539). Eine Kombination aus Cluster-Kohäsion und -Separation bildet der sogenannte Silhouettenkoeffizient von Rousseeuw (1987). Der Silhouettenkoeffizient eines Datenpunktes x_i ist wie folgt definiert:

$$s(i) = \frac{b(i) - a(i)}{\max[a(i), b(i)]},$$

wobei $a(i)$ die durchschnittliche Distanz des Punktes x_i zu allen anderen Punkten desselben Clusters darstellt und $b(i)$ für die durchschnittliche Distanz zu dem Cluster

steht, der dem Punkt x_i am nächsten ist und nicht dem eigenen Cluster entspricht. Der Wert von $s(i)$ kann zwischen -1 und 1 liegen. Negative Werte bedeuten, dass der Punkt näher an einem anderen als dem eigenen Cluster liegt, und positive Werte sagen dementsprechend aus, dass der Punkt dem Cluster zugeordnet ist, der ihm auch am nächsten ist. Um den Silhouettenkoeffizient eines gesamten Clusters oder einer gesamten Clusteraufteilung zu berechnen, werden die Werte der entsprechenden Punkte gemittelt (vgl. Rousseeuw 1987).

Ähnliche Maße, die ebenfalls beide Kriterien (Separation und Kohäsion) kombinieren, sind der Dunn-Index, der die kleinste Distanz zwischen den Clustern mit der größten Distanz innerhalb der Cluster vergleicht (vgl. Dunn 1973), und der Davies-Bouldin-Index, der die durchschnittliche Distanz zwischen den Clustern und ihren nächstgelegenen Nachbar-Clustern berechnet (vgl. Davis & Bouldin 1979).

Neben der Präzision, der Trefferquote und dem F-Maß gehören der Rand-Index (Rand 1971) und der Jaccard-Koeffizient (Jaccard 1901) zu den meistverwendeten externen Clusterevaluierungsmaßen. Zwei Aufteilungen A, B (z.B. Klassenlabels und Clusterlabels oder Labels zwei verschiedener Clusteraufteilungen) werden verglichen, indem folgende Mengen berechnet werden (vgl. Tan et al. 2005, S. 551):

> f_{00} = Anzahl der Datenpunkt-Paare, die sich in unterschiedlichen Mengen in A und in unterschiedlichen Mengen in B befinden.
>
> f_{01} = Anzahl der Datenpunkt-Paare, die sich in unterschiedlichen Mengen in A aber derselben Menge in B befinden.
>
> f_{10} = Anzahl der Datenpunkt-Paare, die sich in derselben Menge in A aber in unterschiedlichen Mengen in B befinden.
>
> f_{11} = Anzahl der Datenpunkt-Paare, die sich in derselben Menge in A und in derselben Menge in B befinden.

Der Rand-Index und der Jaccard-Koeffizient setzen diese Mengen wie folgt in Relation:

$$\text{Rand-Index} = \frac{f_{00} + f_{11}}{f_{00} + f_{01} + f_{10} + f_{11}}$$

$$\text{Jaccard-Koeffizient} = \frac{f_{11}}{f_{01} + f_{10} + f_{11}}$$

Die Ähnlichkeit bzw. Güte einer Clusteraufteilung in Bezug auf eine Referenzaufteilung wird daran gemessen, ob Datenpunkte, die sich im selben Cluster befinden sollen, auch im selben Cluster sind und ob Datenpunkte, die sich in unterschiedlichen Clustern befinden sollen, auch in unterschiedlichen Clustern sind.

Nachdem einige der für diese Arbeit relevanten Methoden diskutiert wurden, befasst sich das nächste Kapitel mit der Architektur und Implementierung eines Prototypen, der eine intrinsische Plagiatserkennung durchführt.

Kapitel 4

Umsetzung der Applikation

Dieses Kapitel setzt sich mit der praktischen Umsetzung der Diplomarbeit auseinander. *Das Ziel besteht darin, einen Prototypen zu entwickeln, der mit Hilfe von Clusterverfahren eine intrinsische Plagiatserkennung durchführt.* Hierbei sollen stilistisch ähnliche Textsegmente zusammengefasst werden. Anhand der entstehenden Cluster wird geschätzt, von wievielen Autoren ein Text stammt und welche Segmente zu welchem Autor (Cluster) zugeordnet werden können. Der erste Abschnitt bezieht sich auf die Architektur der Applikation und beschreibt den Ablauf der Implementierung und die verwendeten Software-Pakete. Im darauffolgenden Abschnitt werden die einzelnen Prozesse im Detail betrachtet: Erstellung der Testdaten, Featureextraktion, Dimensionsreduktion und das Clustern der Daten.

4.1 Architektur

Dieser Abschnitt beschreibt die Architektur des Prototypen und soll den Ablauf der Implementierungsschritte aufzeigen, deren detaillierte Beschreibung im anschließenden Abschnitt folgt.

4.1.1 Ablauf

Die Implementierung gliedert sich in zwei wesentliche Schritte: die Erstellung der Testdaten und das Clustering anhand dieser Daten. Der Ablauf wird in Abbildung 4.1 dargestellt.

Textkorpora für verschiedene Domänen (literarische Texte, Zeitungsartikel, E-Mail-Nachrichten, Forum-Beiträge, Blog-Einträge etc.) und verschiedene Anwendungen (Genre-Erkennung, Themen-Erkennung etc.) wurden konstruiert (vgl. Stamatatos 2009). Ein

Textkorpus für das Testen einer intrinsischen Plagiatserkennung besteht im optimalen Fall aus wissenschaftlichen Arbeiten, deren plagiierte Textsegmente gekennzeichnet sind.

Testdaten, die diese Eigenschaften annähern, können aus der Online-Enzyklopädie Wikipedia erzeugt werden. Wikipedia-Artikel haben eine Versionsgeschichte, anhand jener bestimmt werden kann, welche Inhalte von welchem Autor stammen. Auf diese Weise kann ein Ground-Truth erstellt werden, der folgende Punkte erfüllt:

- wissenschaftliche Domäne
- Text verschiedener Autoren zum selben Thema
- variierende Aufteilung der Textlängen unterschiedlicher Autoren

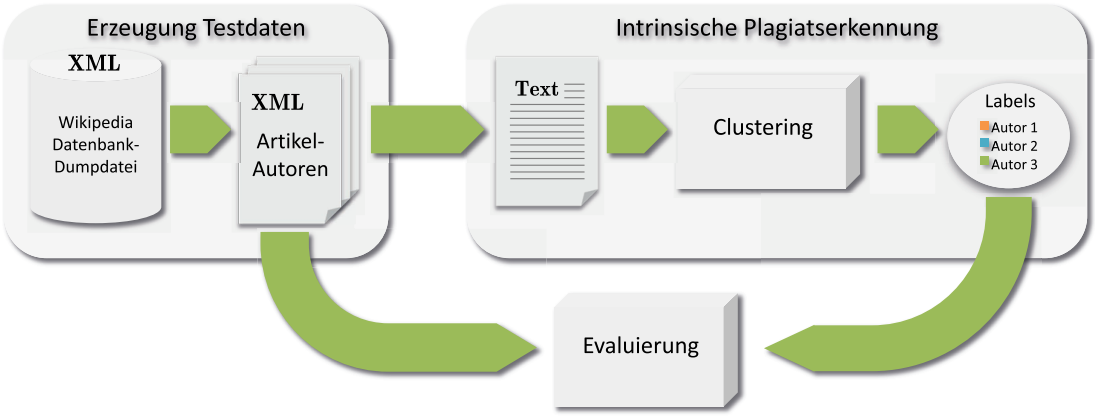


Abbildung 4.1: Gesamter Ablauf: Erstellung Testdaten, intrinsische Plagiatserkennung und Evaluierung.

Aus den Datenbank-Dumpdateien werden Wikipedia-Artikel im XML-Format extrahiert (mit der Information, welches Textsegment von welchem Autor stammt). Aus einem Artikel wird eine Textdatei erstellt, die den Input einer wissenschaftlichen Arbeit unterschiedlicher Autoren simuliert. Anschließend werden textuelle Features aus dieser Datei extrahiert, die als Input für ein Clusterverfahren dienen. Die resultierenden Cluster werden mit der tatsächlichen Klassenaufteilung, die aus dem Artikel im XML-Format hervorgeht, verglichen und evaluiert.

Der Ablauf der intrinsischen Plagiatsprüfung beschränkt sich auf die Schritte vom Einlesen einer Textdatei bis zum Labeln der Daten. Die konkreten Implementierungsdetails der einzelnen Prozesse folgen in Abschnitt 4.2.

Der nächste Abschnitt erläutert die verwendeten Programmbibliotheken und beschreibt, für welche Implementierungsschritte diese jeweils zum Einsatz gekommen sind.

4.1.2 Abhängigkeiten

Der Prototyp und die Organisation der Testdaten sind vollständig in Python geschrieben. Die verwendeten externen Bibliotheken und Implementierungen werden in Tabelle 4.1 zusammengefasst.

Bibliothek	**Implementierung**
lxml	etree (ElementTree-Klasse zur Verarbeitung der hierarchischen XML-Struktur)
NLTK	PunktSentenceTokenizer, RegexpTokenizer, PorterStemmer, ngrams
scikit-learn	PCA, AffinityPropagation, AgglomerativeClustering, DBSCAN, DPGMM, GMM, KMeans, SpectralClustering, metrics
scipy-cluster	dendrogram (Visualisierung der hierarchischen Clusteraufteilung)
NumPy	array (Datenstruktur zur numerischen Verarbeitung n-dimensionaler Datenobjekte)
matplotlib	pyplot (Framework zum Plotten mathematischer Darstellungen – angelehnt an MATLAB)

Tabelle 4.1: Verwendete externe Bibliotheken und ihre Implementierungen.

Für die Verarbeitung der Datenbank-Dumpdateien wird die XML-Bibliothek *lxml*[1] verwendet, die eine iterative Verarbeitung der XML-Struktur ermöglicht, ohne den XML-Baum vorher vollständig durcharbeiten zu müssen (aufgrund der Größe der Datenbank-Dumpdateien von Vorteil).

Das Natural Language Toolkit (NLTK)[2] wird für die Erstellung der Testdaten (Satzsegmentierung für das Filtern von Änderungen innerhalb eines Satzes und der Berücksichtigung einer Mindestanzahl an Sätzen pro Autor) und für die Feature-Extraktion (Wortsegmentierung, Erzeugung von N-Grammen, Stemming) eingesetzt.

Für die Dimensionsreduktion sowie das Clustern und die Evaluierung wird die Machine-Learning-Bibliothek *scikit-learn*[3] eingesetzt. Weitere externe Bibliotheken, die zur Anwendung kommen, sind *scipy-cluster*[4] (visuelle Inspektion des Dendrogramms für hierarchisches Clustering), *NumPy*[5] (effiziente Verarbeitung der extrahierten Features: hochdimensionale Matrizen) und *matplotlib*[6] (Visualisierung der Ergebnisse, für den Prototypen nicht notwendig).

Nachdem eine Übersicht über die Architektur der Applikation geboten wurde, folgt im nächsten Abschnitt ein Einblick in die konkreten Implementierungsschritte.

[1] `http://www.lxml.de`, online abgerufen am 15.10.2014.
[2] Bird et al. (2009): `http://www.nltk.org/`, abgerufen am 15.10.2014.
[3] Pedregosa et al. (2011): `http://scikit-learn.org`, abgerufen am 15.10.2014.
[4] `https://code.google.com/p/scipy-cluster/`, abgerufen am 16.10.2014.
[5] van der Walt et al. (2011): `http://www.numpy.org/`, abgerufen am 15.10.2014.
[6] Hunter (2007): `http://matplotlib.org/`, abgerufen am 15.10.2014.

4.2 Implementierung

Die aufgezeigten Implementierungsschritte befassen sich mit unterschiedlichen Problemstellungen. In den folgenden Abschnitten wird der Ablauf der wesentlichen Schritte diskutiert.

4.2.1 Textkorpus Wikipedia

Für das Testen der intrinsischen Plagiatserkennung wurde ein Textkorpus aus der englischsprachigen Ausgabe der Wikipedia erstellt. Dafür wurden Datenbank-Dumpdateien verarbeitet, die von Wikipedia zur Verfügung gestellt werden[7]. Die Dateien befinden sich im XML-Format, umfassen jeweils Tausende von Artikeln mit vollständiger Versionsgeschichte und sind sehr stark komprimiert. Um die Dumpdateien (im *bzip2*-Format) effizienter zu verarbeiten, wurden sie nicht dekomprimiert, sondern sequentiell (Artikel für Artikel) abgearbeitet. Der Ablauf wird in Abbildung 4.2 veranschaulicht.

Beginnend mit dem Einlesen der Meta-Daten einer Wikipedia-Seite (ID, Titel, Seiten-Typ) wird entschieden, ob die Seite weiter verarbeitet werden soll. Wikipedia unterscheidet mehr als zehn verschiedene Typen: Artikel-Seiten, Benutzer-Seiten, Diskussion-Seiten, Hilfe-Seiten etc. Alle bis auf die tatsächlichen Artikel-Seiten, die keine Weiterleitungen auf andere Artikel sind, werden übersprungen.

Anschließend folgt die Verarbeitung der Versionen eines Artikels. Die Versionen befinden sich in Form der *Wiki-Syntax*, einer Auszeichnungssprache (*markup language*) zur Formatierung von Wikipedia-Beiträgen. Es existieren Werkzeuge, die das Parsen von Wiki-Syntax in verschiedene Formate ermöglichen. Der Großteil der gefundenen Werkzeuge beschränkt sich auf das Verarbeiten von Wikipedia-Datenbank-Dumpdateien, die nur die aktuelle Version der Artikel beinhalten. Für die vorliegende Aufgabe ist außerdem kein komplettes Parsen notwendig, vielmehr muss ein großer Teil des Inhalts herausgefiltert werden: Überschriften, Aufzählungen, Verlinkungen, Tabellen, Formatierungen, Bilder etc. Der eigens entwickelte Parser filtert den Text mit Hilfe regulärer Ausdrücke.

Die Versionen sind chronologisch angeordnet, beginnend mit der ältesten Version. Jede Version repräsentiert den vollständigen Inhalt des Artikels zu einem bestimmten Zeitpunkt. Um die Änderung einer Version zu bestimmen, muss daher jede Version mit der vorherigen verglichen werden. Zudem wird zwischen gültigen und ungültigen Versionen unterschieden. Gültige Versionen erfüllen folgende Kriterien:

- von eingeloggtem Benutzer verfasst (derselbe Autor innerhalb eines Artikels sonst nicht erkennbar)
- Version ist keine Zurücksetzung auf eine ältere Version (Änderungen stammen vom Verfasser der Version, auf die zurückgesetzt wird)

[7] `http://dumps.wikimedia.org/enwiki/`, abgerufen am 17.10.2014.

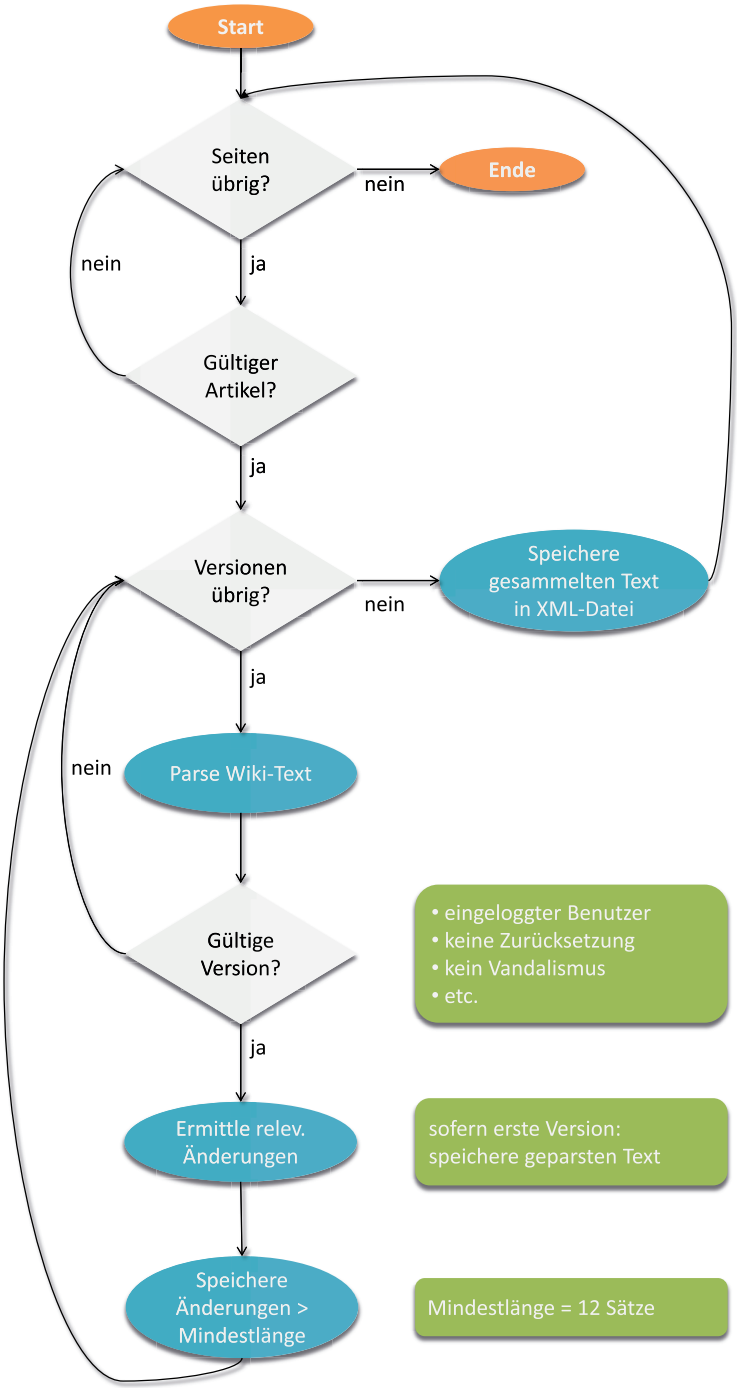


Abbildung 4.2: Verarbeitungsablauf einer Wikipedia-Datenbank-Dumpdatei.

- vom Verfasser nicht als kleine Änderung (*minor changes*) markiert (Filterung trivialer Versionen)
- Änderungen im Vergleich zur letzten Version sind gemessen in Bytes groß genug (Filterung trivialer Versionen)
- kein Vandalismus (zu lange Zeichenketten oder sonstige Probleme beim Parsen des Textes)

Sofern die Version laut diesen Punkten gültig ist, wird sie mit der vorherigen Version verglichen. Vor dem Vergleich werden beide Versionen einer Satzsegmentierung unterzogen (*PunktSentenceTokenizer* des NLTK-Paketes). Mit der Standard-Python-Bibliothek *diff* werden die Sätze beider Versionen verglichen. Änderungen innerhalb eines Satzes und gelöschte Inhalte werden ignoriert. Nur vollständig hinzugefügte Sätze werden beachtet, die einen Absatz von mindestens drei aufeinanderfolgenden Sätzen bilden (für absatzbasierte Features) und insgesamt die Anzahl von zwölf Sätzen nicht unterschreiten. Um eine effektive Stilanalyse durchführen zu können, wird in der Literatur oft die Mindestlänge hervorgehoben, die ein Text haben sollte (vgl. Forsyth & Holmes 1996; Stamatatos 2009).

Beinhaltet eine Version genügend neue vollständige Sätze, wird der Text in die XML-Datei des Artikels aufgenommen. Meta-Daten wie die ID des Autors und die entsprechende Anzahl der Sätze werden für die abschließende Evaluierung gespeichert. Sobald alle Versionen eines Artikels verarbeitet wurden, wird der nächste gültige Artikel gesucht und nach demselben Schema iteriert. Aus Effizienzgründen wird die Verarbeitung nach einer festgelegten Anzahl an Versionen (derzeit 3000) gestoppt und die verbleibenden Versionen übersprungen. Die meisten Inhalte stammen von den ersten Versionen eines Artikels und die Anzahl hinzugefügter vollständiger Sätze nimmt mit höherer Versionsnummer ab.

Schwierigkeiten ergeben sich vorwiegend beim Parsen der Texte. Wikipedia führt vor dem Speichern einer Version keine Validitätsprüfung durch: Fehler in der Syntax (nicht abgeschlossene Markierungen, überflüssige Markierungszeichen etc.) müssen deshalb korrekt gefiltert werden, sodass im exportierten Text unnötige Formatierungszeichen vermieden werden.

Ein weiteres Problem stellen Versionen dar, die eine vorherige Version rückgängig machen und auf eine ältere Version zurücksetzen. Die Versionen sind mit eindeutigen Hash-Werten gekennzeichnet. Das Zurücksetzen auf eine ältere Version kann mit dem Vergleich der Hash-Werte festgestellt werden. Werden beim Zurücksetzen (entgegen gängiger Praxis) gleichzeitig noch zusätzliche Änderungen vorgenommen, kann das Zurücksetzen über die Hash-Werte nicht mehr nachvollzogen werden. So besteht die Möglichkeit, dass aus einer Version Änderungen extrahiert werden, die von einer älteren Version stammen und nicht dem Verfasser der aktuellen Version zuzuschreiben sind. Jede Version kann mit einem Kommentar versehen werden, der die vorgenommenen Änderungen beschreibt. Dieser Kommentar wird bei der Gültigkeitsprüfung einer Version mitberücksichtigt.[8]

[8] Das Zurücksetzen auf eine ältere Version wird üblicherweise mit den Worten *revert, rev* oder ähnlichen Ausdrücken gekennzeichnet.

Um die Qualität der Testdaten zu erhöhen, werden die exportierten Texte manuell überprüft und unbrauchbare Daten entfernt. Darunter fallen Vandalismus-Beiträge (z.B. eine Folge sich wiederholender Wörter/Sätze/Absätze) und Fehler beim Parsen.

Die so gewonnenen Testdaten sind Dateien im XML-Format, beinhalten Text aus jeweils einem Wikipedia-Artikel und sind von n unterschiedlichen Autoren verfasst. Jede Zeile besteht genau aus einem Satz, und der Text eines Autors umspannt mindestens zwölf vollständige Sätze. Aus der XML-Datei wird eine Textdatei erzeugt, welche als Input für ein Clusterverfahren fungiert. Die Struktur des Artikels (welcher Teil von welchem Autor geschrieben worden ist) dient zur Evaluierung der Clusteraufteilung.

Die Beschreibung der Testdaten ist damit abgeschlossen. Der nächste Abschnitt erläutert die Extraktion textueller Features.

4.2.2 Feature-Extraktion

Textuelle Features wurden bereits in Abschnitt 3.1 diskutiert. Im folgenden Abschnitt wird beschrieben, welche Features effektiv für den Prototyp zur Anwendung kommen und wie sie extrahiert werden.

Es wurde versucht, Features zu verwenden, die nicht zu stark vom Inhalt des Textes abhängig sind. *Hierbei haben sich keine eindeutigen Eigenschaften herausgebildet, welche die Autoren global am besten unterscheiden können.* Je nach Testdaten weisen unterschiedliche Features die höchste Varianz auf. Dabei heben sich Funktionswortfrequenzen und N-Gramm-Features auffallend oft von den übrigen ab.

Der verwendete Feature-Vektor setzt sich aus folgenden Eigenschaften zusammen:

Lexikalische Features: mittlere Wortlänge, Wortlängenfrequenz, kurze Wörter (Buchstabenanzahl ≤ 3), mittellange Wörter (≤ 8), lange Wörter (> 8), Wortlängen-Bi-Gramme, Wortlängen-Tri-Gramme, mittlere Satzlänge, kurze Sätze (Wortanzahl ≤ 6), mittellange Sätze (≤ 20), lange Sätze (≤ 30), extralange Sätze (> 31), Satzlängen-Bi-Gramme, Satzlängen-Tri-Gramme, Satzlängen-4-Gramme, spezielle Buchstaben-Bi-Gramme, Buchstaben-Tri-Gramme, meistgebrauchte Konsonantengruppen, Wortschatzmaß Yule's K, Wortschatzmaß Hapax Legomena, Wortschatzmaß Dis Legomena.

Syntaktische Features: Funktionswortfrequenzen, Wortartfrequenzen, Wortart-Bi-Gramme, primäre Verben, Hilfsverben, Satzzeichenfrequenz, Sonderzeichenfrequenz.

Die Bezeichnungen der Features sollten selbsterklärend sein. Wenn möglich wird ein Verhältnis zur gesamten Menge berechnet (z.B. kurze Wörter: das Verhältnis von kurzen Wörtern zur Gesamtanzahl der Wörter). Zusätzliche Erläuterungen einzelner Features werden in Tabelle 4.2 zusammengefasst.

Die Bestimmung der Feature-Parameter (z.B. Länge kurzer Sätze ≤ 6 Wörter) wurde von anderen Arbeiten übernommen oder heuristisch bestimmt[9] ohne weitere Untersuchungen anzustellen. Einige Features wurden anhand manuell erstellter Listen erfasst: für die Funktionswortfrequenzen wurden 343 Funktionswörter aus drei verschiedenen Listen aufgenommen.[10,11,12] Ein Vorteil, der gegenüber der Verwendung der n häufigsten Wörter eines Korpus entsteht (siehe Abschnitt 3.1.1), ist die Berücksichtigung zusammengesetzter Funktionswörter (Bi-, Tri- und 4-Gramme): z.B. *due to, in spite of, by the time of* etc.

Feature-Typ	**Feature**	**Bemerkung**
Lexikalisch	Wortlängenfeatures	an der Buchstabenanzahl gemessen (eine andere Möglichkeit wäre z.B. Silbenanzahl)
	Satzlängenfeatures	an Wortanzahl gemessen
	Satzlängen-N-Gramme	nicht tatsächliche Satzlänge, sondern Kombinationen wie: kurzer Satz + kurzer Satz, langer Satz + kurzer Satz + langer Satz etc.
	spezielle Buchstaben-Bi-Gramme	Frequenz meistgebrauchter Buchstaben-Bi-Gramme (t+h, s+t, a+n, i+n, e+r, r+e, t+i, e+a etc.) und Frequenz von Vokal+Vokal, Vokal+Konsonant, Konsonant+Vokal, Konsonant+Konsonant[13]
	meistgebrauchte Konsonantengruppen	Konsonantenfrequenz folgender Gruppen im Verhältnis zu den gesamten Buchstaben: tnsrh, ldcpf, mwybg, jkqvxz[13]
Syntaktisch	primäre Verben	Frequenzen primärer Verben und Formen davon: to be, to do, to have[13]
	Hilfsverben	Frequenzen von Hilfsverben und Formen davon: can, could, may, might, must, shall, should, will, would, ought, dare, need, had better, used to[13]

Tabelle 4.2: Details zu einzelnen Features.

Der Ablauf der Feature-Extraktion wird in Abbildung 4.3 veranschaulicht. Der Text wird mit einem sogenannten Sliding-Window schrittweise verarbeitet (vgl. Abbasi & Chen 2008). Das Sliding-Window hat eine Länge n und eine Schrittweite k. Nachdem die ersten n

[9] Beispiel: Die Intervalle für kurze, mittellange und lange Wörter wurden aufgrund der durchschnittlichen Wortlänge (von 5,1) der englischen Sprache gewählt: `http://www.wolframalpha.com/input/?i=average+word+length+in+English`, abgerufen am 22.10.2014.

[10] `http://www.sequencepublishing.com/academic.html`, abgerufen am 22.10.2014.

[11] `https://web.archive.org/web/20130911024236/http://www.flesl.net/Vocabulary/Single-word_Lists/function_word_list.php`, Internet-Archive-Version vom 11.09.2013, abgerufen am 22.10.2014.

[12] `http://grammar.yourdictionary.com/parts-of-speech/conjunctions/conjunctions.html`, abgerufen am 22.10.2014.

[13] Übernommen von Schulstad et al. (2012).

Sätze des Textes betrachtet wurden, wird das Fenster um k Sätze weitergeschoben und die Features der nächsten n Sätze extrahiert, bis das Fenster am Ende des Textes angelangt ist. Dieses Verfahren hat den Zweck, eine gewisse Textlänge für die stilometrische Analyse zu garantieren und den Übergang zwischen Textsegmenten verschiedener Autoren besser bzw. überhaupt erfassen zu können. Es wurden verschieden Parameter für n und k versucht. In der derzeitigen Version wird eine Länge von sechs Sätzen und eine Schrittweite von einem Satz verwendet.

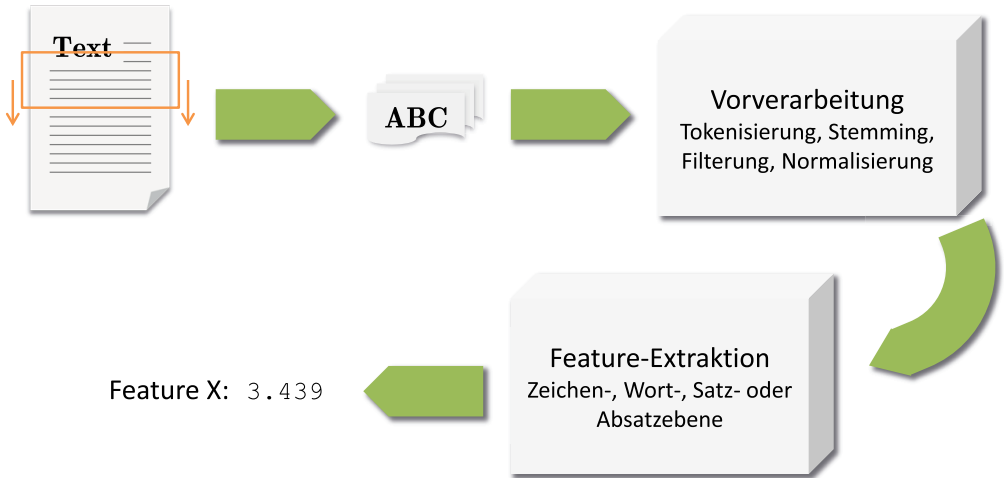


Abbildung 4.3: Ablauf Feature-Extraktion – wird pro Textsegment n mal durchgeführt (n Features).

Für jedes Text-Segment (Sliding-Window) werden d Features berechnet. Je nach Feature sind unterschiedliche Vorverarbeitungsschritte nötig. Dazu gehören die Tokenisierung (für nahezu alle Features), die Filterung von bestimmten Zeichen (z.B. Satzzeichen oder Ziffern für wortbasierte Features), das Stemming (für Wortschatzmaße) und Normalisierungen (z.B. Umwandlung in Kleinbuchstaben: für Funktionswortfrequenzen, Wortschatzmaße etc.).

Bei der anschließenden Extraktion der Features muss noch beachtet werden, in welchem Zusammenhang sie erfasst werden (Zeichen-, Wort-, Satz- oder Absatzebene). Abhängig davon werden die Features extrahiert:

- Wortart-Bi-Gramme werden nicht über die Grenze eines Satzes hinaus gebildet.
- Absatzabhängige Features (Satzlängen-N-Gramme) werden nicht über die Grenze eines Absatzes hinaus gebildet.
- Buchstaben-N-Gramme werden nicht über die Grenze eines Wortes hinaus gebildet etc.

Die extrahierten Features der Textsegmente unterscheiden sich voneinander (z.B. kann ein Textsegment ein Wortart-N-Gramm aufweisen, das in keinem anderen Textsegment vorkommt). Deshalb werden abschließend alle extrahierten Features vereint und jedem Textsegment zugeordnet (sofern nicht vorhanden: Null-Wert).

Die auf diese Weise gewonnenen Features ergeben eine Zahl im vierstelligen Bereich (abhängig von der Textlänge ca. 2000 bis 6000 Features). Vor allem die N-Gramm-Features sind verantwortlich für diese hohe Feature-Anzahl. Welche Probleme sich dadurch ergeben und wie die Dimensionalität reduziert werden kann, folgt im nächsten Abschnitt.

4.2.3 Dimensionsreduktion

Ein Clustern im hochdimensionalen Raum kann verschiedene Schwierigkeiten mit sich bringen. Clusteralgorithmen sind unterschiedlich gut skalierbar: Da eine Verarbeitung im hochdimensionalen Raum komplexer ist, kann dies zu impraktikablen Laufzeiten führen (vgl. Duda et al. 2001, S. 169f.).

Ein weiteres Problem ergibt sich durch das Volumen des Vektorraumes, das exponentiell zur Dimensionalität steigt. Es benötigt exponentiell mehr Datenpunkte, um dieselbe Dichte an Punkten in höher-dimensionalen Räumen zu halten (vgl. Bishop 2006, S. 35).

Die Konzentration der Distanzwerte zwischen den Datenpunkten in höheren Dimensionen ist auch problematisch. Sofern die Varianz gegen Null strebt ($\sigma^2 \to 0$), ist der Unterschied zwischen der kleinsten und größten Distanz von Punkten einer Datenmenge so gering, dass Distanzmaße ihre Funktionalität im Sinne der Unterscheidungskraft verlieren (vgl. Houle et al. 2010).

Die Einschränkungen, die sich mit zunehmender Dimensionalität eines Vektorraumes (oder steigender Parameteranzahl) ergeben, werden oft unter dem Begriff des *Fluches der Dimensionalität (curse of dimensionality)* zusammengefasst (vgl. Bishop 2006, S. 36). Eine Möglichkeit, der Problematik entgegenzuwirken, ist das Einbeziehen von Ground-Truth-Daten: sofern die Klassenlabels der Daten bekannt sind, ist das Erkennen *unnötiger* Dimensionen leichter. Deshalb sind Klassifikatoren für die Verarbeitung hochdimensionaler Daten besser geeignet als Clusterverfahren (vgl. Duda et al. 2001, S. 170).

Ein Ansatz, dem Problem auszuweichen, besteht in der Dimensionsreduktion der Datenmenge. Ein häufig verwendetes Verfahren dafür ist die Hauptkomponentenanalyse oder englisch Principal Component Analysis (PCA). Die PCA berechnet eine orthogonale Projektion der Daten, sodass die Achsen den Richtungen der stärksten Varianzen entsprechen: die erste Achse zeigt in Richtung der stärksten Varianz, die zweite in Richtung der zweitstärksten etc. Werden nur die ersten n Dimensionen berücksichtigt, kann auf diese Weise eine Dimensionsreduktion durchgeführt werden, welche die Varianz der Daten maximiert bzw. den Informationsverlust minimiert (vgl. Bishop 2006, S. 561).

Die Richtungen der stärksten Varianzen werden durch die Eigenvektoren der Kovarianz-Matrix berechnet. Die Eigenvektoren mit den niedrigsten Eigenwerten entsprechen den Richtungen der geringsten Varianz und werden bei einer Dimensionsreduktion ignoriert (vgl. Smith 2002).

Die in Abschnitt 4.2.2 beschriebene Feautre-Extraktion ergibt eine hohe Feature-Anzahl. Mit einer PCA kann die Dimensionalität stark reduziert werden, ohne dabei viel der Varianz der Daten zu verlieren: die Anzahl wird so gewählt, dass 95% der Varianz erhalten bleibt. Die Features können dadurch auf etwa 2-5% ihrer ursprünglichen Menge reduziert werden (z.B. von 1000 auf 50 Features bzw. Dimensionen).

Die Features wurden extrahiert und die Dimensionalität der Daten konnte reduziert werden. Im nächsten Schritt werden die Daten im reduzierten Feature-Raum einer Clusteranalyse unterzogen.

4.2.4 Clustering

Die verwendeten Clusteralgorithmen wurden bereits in Abschnitt 3.2 vorgestellt. Im folgenden Abschnitt wird erläutert, mit welchen Verfahren positive Ergebnisse erzielt werden konnten und mit welchen Parametern sie durchgeführt wurden. Eine konkrete Evaluierung folgt im nächsten Abschnitt.

Verfahren mit automatischer Bestimmung der Clusteranzahl

Zunächst wurden folgende Verfahren angewandt, welche die Anzahl der Cluster automatisch bestimmen.

Affinity Propagation neigt dazu, eine sehr hohe Anzahl an Clustern zu finden. Dabei sind die Clusteraufteilungen häufig derart gekennzeichnet, dass einige wenige Cluster die meisten Datenpunkte zusammenfassen und eine große Menge an Clustern jeweils nur einzelne Datenpunkte beinhalten. Hierbei konnte keine Einstellung gefunden werden, anhand derer die Daten annähernd korrekt geclustert werden.

Folgende Parameter wurden variiert: Präferenz-Wert, Damping-Faktor, Metrik/Ähnlichkeitsmaß (negative quadratische euklidische Distanz und Gauß-Kernel).

DBSCAN kann für einzelne Testdaten gute Ergebnisse erzielen und die tatsächlichen Klassenaufteilungen gut annähern. Es hat sich jedoch gezeigt, dass die Dichte-Parameter für verschiedenartige Testdaten zu sensibel sind und eine global funktionierende Einstellung daher nicht gefunden werden konnte. Es ist anzunehmen, dass das Problem vor allem in der sehr unterschiedlich ausgeprägten Dichte der Klassen liegt: einerseits begründet durch die stark variierenden Textlänge eines Autors und andererseits abhängig von der unterschiedlichen Ausprägung der Features (des Schreibstils).

Folgende Parameter wurden variiert: ϵ, $minPts$, Metrik/Ähnlichkeitsmaß (euklidische Distanz und Gauß-Kernel).

GMM mit Dirichlet-Prozess neigt dazu, die vorgegebene obere Schranke der Clusteranzahl (die verwendete Implementierung fordert diesen Parameter) anzunähern. Als Oberschranke wurden verschiedene Werte getestet. Mit iterativer Steigerung der Schranke konnte festgestellt werden, dass das Verfahren dazu tendiert, in der Nähe der korrekten Klassenanzahl häufiger zu halten. Die resultierende Clusteraufteilung entsprach jedoch nicht den tatsächlichen Klassen.

Folgende Parameter wurden variiert: Kovarianztyp (*spherical, tied, diagonal, full*), Konzentrations-Parameter α, Konvergenz-Schwellenwert, Initial- und Update-Parameter (Gewichte, Mittelpunktvektoren, Kovarianzmatrizen).

Für die angeführten Verfahren konnten die korrekten Klassenaufteilungen nicht angenähert werden (teilweise aufgrund der Parameter, die nur für eine Gruppe der Testdaten zu positiven Ergebnissen führten). Ein anderer Ansatz wurde mit den nachfolgenden Verfahren versucht, die die Clusteranzahl nicht automatisch bestimmen (die Anzahl ist entweder ein vorgegebener Parameter oder wird ignoriert).

Verfahren ohne automatische Bestimmung der Clusteranzahl

Folgende Verfahren wurden zunächst mit der Vorgabe der korrekten Klassenanzahl angewandt bzw. beim agglomerativen Clustering im Nachhinein auf diese Zahl beschränkt.

Agglomeratives Clustering teilt die Daten je nach verwendetem Linkage-Typ sehr unterschiedlich auf. Single- und Complete-Linkage verfehlen die korrekten Klassenaufteilungen stark. Average-Linkage und Wards Methode nähern die korrekten Lösungen an, wobei die Methode von Ward deutlich besser abschneidet. Der hierarchische Baum wird vollständig berechnet und im Anschluss an der Stelle geschnitten, an der die Anzahl der Cluster mit der vorgegeben Anzahl übereinstimmt. Auch das Finden eines internen Stopp-Kriteriums wurde versucht. Die Clusteranzahl sollte aufgrund der Höhenunterschiede des Dendrogramms festgestellt werden. Die korrekte Klassenanzahl konnte mit diesem internen Kriterium jedoch nur für einzelne Testdaten angenähert werden.

Folgende Parameter wurden variiert: Linkage-Typ (Single-, Complete-, Average-Linkage und Wards Methode), Metrik (euklidische Distanz).

GMM mit vorgegebener Clusteranzahl liefern ähnliche Clusteraufteilungen wie agglomeratives Clustering mit Wards Methode. Es wurden verschiedene Arten von Kovarianzmatrizen versucht. Mit der sphärischen Form konnten die besten Ergebnisse erzielt werden. Neben zufällig gewählten Startpunkten wurde auch eine Initialisierung mit K-Means angestellt, die zu vergleichbaren Ergebnissen führte.

Folgende Parameter wurden variiert: Kovarianztyp (*spherical, tied, diagonal, full*),

Konvergenz-Schwellenwert, Initial- und Update-Parameter (Gewichte, Mittelpunktvektoren, Kovarianzmatrizen).

K-Means ist stärker von der Initialisierung der Startpunkte abhängig als GMM, liefert aber trotzdem ähnliche Clusteraufteilungen, die die korrekte Lösung jeweils annähernd gut treffen.

Folgende Parameter wurden variiert: Initialisierung (*kmeans++, random*), Anzahl Initialisierungen, Metrik (euklidische Distanz).

Spectral Clustering hängt vom verwendeten Ähnlichkeitsmaß ab. Wird eine Ähnlichkeitsmatrix verwendet, die aus den k-nächsten Nachbarpunkten (*knn*) berechnet wird, hängt die Performance vom gewählten k ab und schwankt mit unterschiedlichen Werten stark. Mit der Verwendung eines Gauß-Kernels kann die korrekte Klassenaufteilung gut angenähert werden.

Folgende Parameter wurden variiert: Label-Zuweisung (*kmeans, discretize*), Ähnlichkeitsmaß (*knn*-Adjazenzmatrix, Gauß-Kernel).

Mit vorgegebener korrekter Clusteranzahl k liefern alle vier Verfahren ähnliche Ergebnisse, wobei K-Means eine etwas geringere Trefferquote aufweist.

Durch die Verarbeitung mit dem Sliding-Window wird ein Satz von mehreren Datenpunkten repräsentiert und kann zu unterschiedlichen Clustern zugeordnet werden: Bei einer Sliding-Window-Länge von sechs Sätzen und einer Schrittweite von einem Satz wird ein Satz von sechs Datenpunkten repräsentiert (sofern sich der Satz nicht am Anfang oder Ende des Textes befindet). Im nicht eindeutigen Fall wird geprüft, ob es einen Übergang zu den anliegenden Sätzen bzw. Absätzen gibt und die Zuordnung dementsprechend getroffen.

Da die Verfahren mit vorgegebener Clusteranzahl die tatsächliche Klassenaufteilung gut annähern konnten, wurde versucht, die korrekte Clusteranzahl mit einem relativen Evaluierungskriterium zu bestimmen.

4.2.5 Bestimmung der Clusteranzahl

Durch relative Evaluierungskriterien ist es möglich, die Anzahl der Cluster aufgrund der Veränderung des gewählten Maßes (mit steigender Clusteranzahl) zu bestimmen. Das Clusterverfahren wird dabei wiederholt mit steigendem k durchgeführt und der Clusteranzahl ($k = 1..n$) gegenübergestellt. Es wird versucht, in der daraus entstehenden Kurve einen Punkt zu finden, der eine starke Veränderung ausdrückt, und ab dem die Kurve beginnt abzuflachen (vgl. Duda et al. 2001, S. 557).

Verschiedene Maße für die Bewertung einer Clusteraufteilung sind möglich. Goutte et al. (1999) haben die Varianz innerhalb der Cluster in Verbindung mit einem agglomerativen Clusterverfahren verwendet. In Goutte et al. (2001) kommen GMM zum Einsatz, und

Maße wie Akaikes Informationskriterium (AIC) und das Bayessche Informationskriterium (BIC) werden der Clusteranzahl gegenübergestellt.

Um den Knick in der Kurve zu finden, gibt es die Vorgehensweise, das erste lokale Maximum der Kurve (bzw. der ersten oder zweiten Ableitung) zu verwenden. Zhao et al. (2008) stellen eine *Knee Point Detection* für BIC-Werte vor, in der sie die gesamte Kurve betrachten und Lösungen mit höherem k unterschiedlich stark gewichten.

Die vorgestellten Verfahren, welche die Clusteranzahl nicht automatisch bestimmen, nähern die korrekte Klassenaufteilung ähnlich gut an. Für die Ermittlung der Clusteranzahl anhand eines relativen Evaluierungskriteriums wurde agglomeratives Clustering mit Wards Methode und die mittlere Varianz innerhalb der Cluster als Bewertungsmaß gewählt. Das Verfahren ist nicht abhängig von einer Initialisierung der Startpunkte und liefert für steigende Clusteranzahl k konsistente Clusteraufteilungen. Da die Clusterbildung so funktioniert, dass die Varianz minimiert wird, existiert damit gleichzeitig ein Maß mit dem das Clustering bewertet werden kann. Die geringe Laufzeit agglomerativen Clusterings[14] ist für das Messen von Lösungen mit unterschiedlicher Clusteranzahl von Vorteil. Die Berechnung von Zhao et al. (2008) wurde in abgeänderter Form übernommen:

$$C = \frac{(\sigma^2 - \sigma^2_{min})}{(\sigma^2_{max} - \sigma^2_{min})}$$

$$C_k = \frac{C}{(k+2)^d}$$

$$C_{diff} = \frac{|C - C_k|}{2}$$

C stellt die normalisierte Kurve der fallenden Varianz dar. C_k nimmt den Wert der Varianz und teilt ihn durch die Clusteranzahl $(k+2)^d$. Zhao et al. (2008) teilen lediglich durch die Anzahl der Cluster. Die Anpassung ist notwendig, um auch Lösungen für $k = 1$ zu ermöglichen. Es wurden verschiedene Werte für die Konstante d getestet, dabei hat sich der Wert von $d = 0,95$ als geeignet erwiesen (Details dazu siehe Kapitel 5).

Der k-Wert des globalen Maximums der C_{diff}-Kurve wird als Clusteranzahl übernommen. In Abbildung 4.4 wird die normalisierte Varianz C eines Testdatensatzes in Orange und die Differenz C_{diff} in Blau dargestellt. Die Varianz ist am Beginn (mit nur einem Cluster) maximal und sinkt mit steigender Clusteranzahl (sobald jeder Cluster genau einen Datenpunkt enthält, beträgt die Varianz Null). Das Maximum erreicht die C_{diff}-Kurve an der der Stelle $k = 4$.

Um eine Aussage darüber zu treffen, wie vertrauenswürdig die getroffene Wahl der Clusteranzahl ist, wurde ein Qualitätsmaß entwickelt. Für die Berechnung spielen zwei Faktoren eine Rolle: die Gewichtung des globalen Maximums und die Gleichmäßigkeit

[14] Das Verfahren muss theoretisch nur einmal durchgeführt werden. Das Dendrogramm wird einmal berechnet und kann anschließend schrittweise nach unten verarbeitet werden.

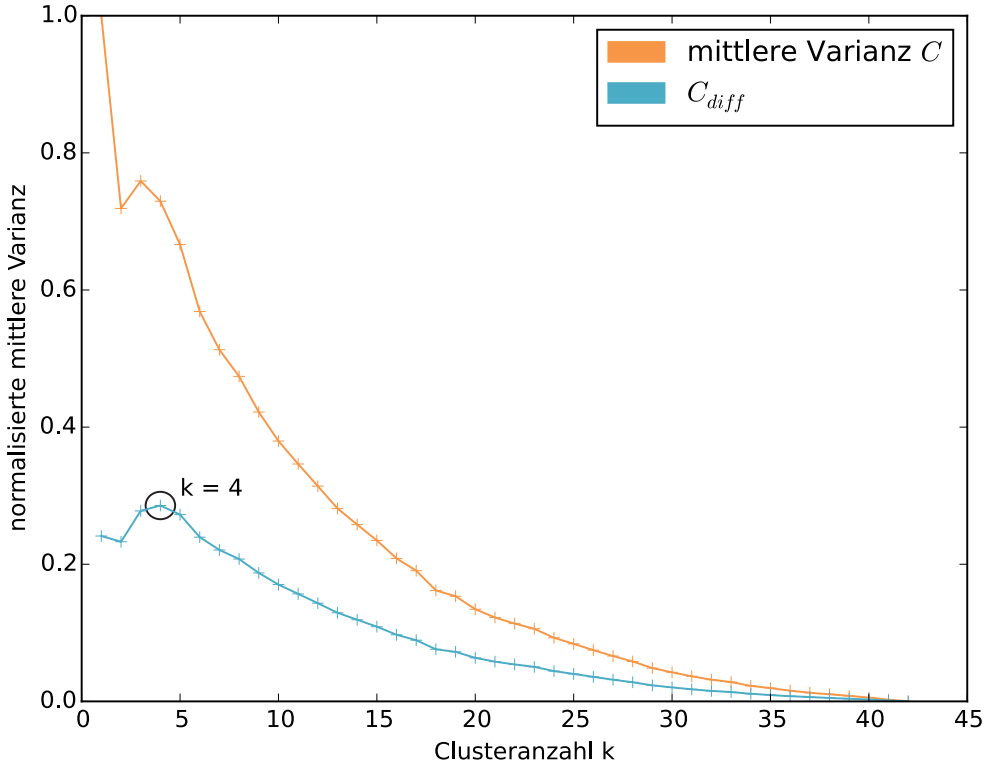


Abbildung 4.4: Ermittlung der Clusteranzahl k über den Vergleich mit der fallenden Varianz. Die C_{diff}-Kurve wird in blau dargestellt und befindet sich unter der C-Kurve. Die tatsächliche Klassenanzahl dieser Testdaten beträgt 5 und das Qualitätsmaß 0,9091.

des restlichen Kurvenverlaufes. Die Gewichtung des globalen Maximums wird berechnet, indem alle lokalen Maxima anhand der Höhendifferenz (y-Achse) und Entfernung zum nächsten Anstieg (x-Achse) bewertet werden und die Bewertung des globalen Maximums in Relation zur Gesamtheit gesetzt wird. Die Gleichmäßigkeit der Kurve rechts vom gewählten k wird berechnet, indem die Differenz der beiden Summen der lokalen Maxima und Minima von Eins abgezogen wird (gibt es keine starken Schwankungen, gleichen sich die beiden Summen aus). Die beiden Parameter werden miteinander multipliziert und ergeben einen Wert zwischen Null und Eins. Weist die Kurve ein sehr klares globales Maximum auf und verläuft die Kurve rechts davon gleichmäßig gegen Null, strebt der Wert des Qualitätsmaßes gegen Eins.

Damit wurden die wesentlichen Schritte der Implementierung des Prototypen erläutert. Im nächsten Kapitel wird die Evaluierung mit den Testdaten beschrieben und eine Bewertung der Ergebnisse angestellt.

Kapitel 5

Evaluierung der Ergebnisse

Dieses Kapitel befasst sich mit der Validierung der Ergebnissen, welche anhand der verwendeten Clusterverfahren erzielt werden konnten. Der erste Abschnitt bezieht sich auf die Methodik der Evaluierung und beschreibt, welche Testdaten zum Einsatz gekommen sind und mit welchen statistischen Gütemaßen die Ergebnisse bewertet wurden. Im zweiten Abschnitt werden die Ergebnisse präsentiert und erläutert.

5.1 Methodik der Evaluierung

Für die Erstellung der Testdaten wurde (wie in Abschnitt 4.2.1 beschrieben) eine Datenbank-Dumpdatei der englischsprachigen Ausgabe der Wikipedia verarbeitet. Die daraus entnommenen Daten umspannen 150 Texte mit unterschiedlicher Autorenanzahl (1 bis 13 Autoren) und Länge (15 bis 1002 Sätze). Die Aufteilung nach Autorenanzahl wird in Tabelle 5.1 veranschaulicht. Von den 150 Texten weisen 50 einen Autor und 100 mehrere Autoren auf.

Autorenanzahl	**Anzahl Dokumente**
1	50
2	28
3	23
4	15
5	14
>6	20
Summe	150

Tabelle 5.1: Aufteilung der Testdaten nach Autorenanzahl.

Testdaten, welche nur von einem Autor verfasst wurden, sind in der Regel kürzer. Außerdem ist die Anzahl der extrahierten Artikel, welche nur von einem Autor stammen, wesentlich kürzer. Daher wurden zu den ursprünglichen 35 Artikeln einzelner Autoren zusätzlich 15 Texte aus Segmenten von Artikeln mehrerer Autoren erzeugt. Dabei wurden Textsegmente mit höherer Satzanzahl bevorzugt.

Um festzustellen, wie gut die korrekte Klassenaufteilung erreicht werden kann, wurde die Performance der vorgestellten Clusterverfahren mit vorgegebener Clusteranzahl untersucht. Des Weiteren wurde eine SVM trainiert, um die Effektivität überwachter Verfahren zu testen. Abschließend wurde Wards Methode mit der mittleren Varianz innerhalb der Cluster als relatives Evaluierungskriterium untersucht.

Da der Ground-Truth der Daten bekannt ist, wurde das F1-Maß und der angepasste Rand-Index (Details dazu siehe Abschnitt 3.3.3) für die Evaluierung herangezogen. Das F1-Maß ist das harmonische Mittel von Präzision und Trefferquote und nimmt Werte zwischen 0 (schlechtester Wert) und 1 (bester Wert) an. Der angepasster Rand-Index setzt den erreichten Rand-Index in Relation dazu, wie gut die Klassenaufteilung durch eine Zufallsverteilung erreicht werden kann und liegt zwischen -1 (schlechtester Wert) und 1 (bester Wert). Interne Evaluierungsmaße wie der Silhouettenkoeffizient, welcher die Cluster-Kohäsion und -Separation bewertet, haben sich nicht als effizient für die Auswertung erwiesen. Sie fallen im Mittel sehr niedrig aus und ermöglichen keine deutliche Unterscheidung zwischen gut und schlecht angenäherten Clusteraufteilungen (zurückzuführen auf die nicht eindeutige Separation der Cluster).

Nachdem die Methodik der Evaluierung erläutert wurde, werden im nächsten Abschnitt die Ergebnisse präsentiert.

5.2 Ergebnisse

In diesem Abschnitt werden die Ergebnisse gruppiert nach verschiedenen Kriterien (Autorenanzahl, Satzanzahl, Abweichung von der korrekten Klassenanzahl etc.) veranschaulicht und bewertet.

5.2.1 Performance mit vorgegebener Clusteranzahl

Zunächst wurde die Performance verschiedener Clusterverfahren mit vorgegebener Clusteranzahl (entsricht der jeweiligen korrekten Autorenanzahl) gemessen. Unabhängig von der Schätzung der Autorenanzahl soll überprüft werden, wie gut die Sätze eines Textes zu Clustern zusammengefasst werden. Jeder Cluster stellt einen Autor dar und es wird getestet, ob die resultierenden Cluster der tatsächlichen Autorenaufteilung entsprechen. Abbildung 5.1 veranschaulicht das mittlere F1-Maß und Abbildung 5.2 den mittleren angepassten Rand-Index. Es kann festgestellt werden, dass alle vier Verfahren ähnlich gut abschneiden, wobei GMM und K-Means etwas hinter der Methode von Ward und Spectral

Clustering liegen. Das beste F1-Maß wird mit der Methode von Ward erreicht (0,71), das schlechteste mit GMM (0,67). Gemessen am Rand-Index wird der Unterschied zwischen den Verfahren deutlicher. Das beste Ergebnis wird wiederum mit dem agglomerativen Clustering nach Wards Methode erreicht (0,46), das schlechteste mit GMM und K-Means (0,36).

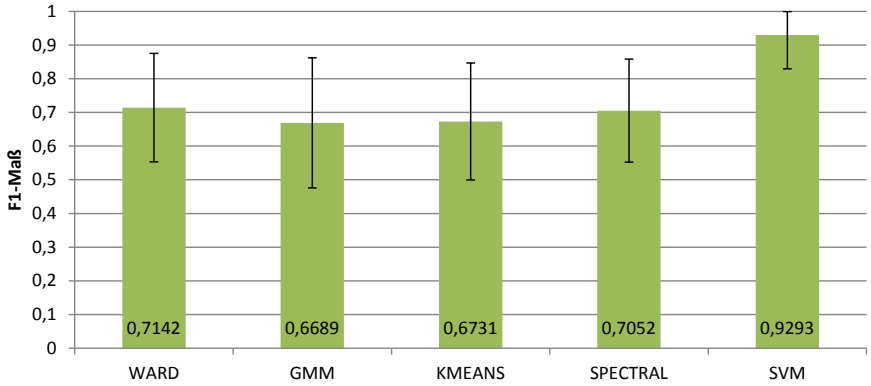


Abbildung 5.1: Clusterperformance mit vorgegebener Autorenanzahl und Performance einer SVM im Vergleich: mittleres F1-Maß $\pm$ σ. Testdaten mit nur einem Autor wurden ignoriert.

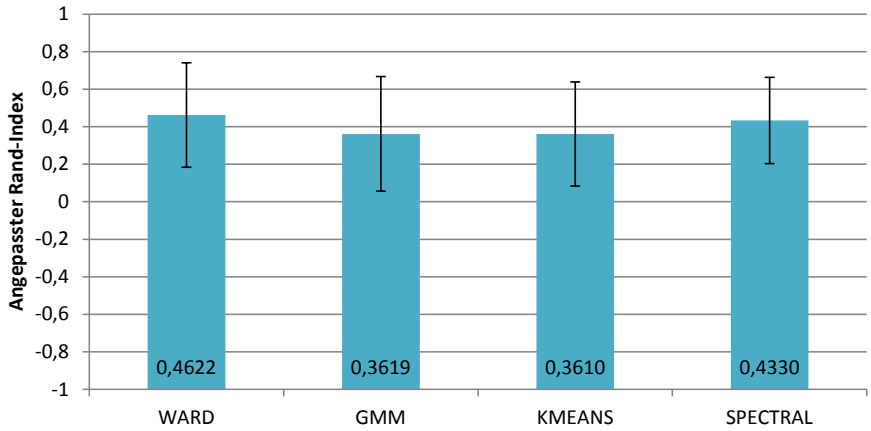


Abbildung 5.2: Clusterperformance mit vorgegebener Autorenanzahl: mittlerer angepasster Rand-Index $\pm$ σ. Testdaten mit nur einem Autor wurden ignoriert.

Aus diesen Ergebnissen kann geschlossen werden, dass die verwendeten Features eine Unterscheidung der Texte verschiedener Autoren ermöglichen, wodurch die tatsächliche Klassenaufteilung annähernd gut getroffen wird. Das Ergebnis eines überwachten Verfahrens kann das bestätigen. Eine SVM wurde trainiert und mit einer stratifizierten Kreuzvalidierung (10 Teilmengen) ausgewertet. Das resultierende mittlere F1-Maß ist mit den Ergebnissen von Abbasi & Chen (2008) vergleichbar (sie verwenden einen ähnlichen

Feature-Vektor) und beträgt 0,93. Die Differenz zwischen der Clustering- und Klassifikationsperformance verdeutlicht die Stärke überwachter Verfahren und zeigt, dass die Klassen nicht eindeutig voneinander trennbar sind (ansonsten würde sich die Performance unüberwachter Verfahren nicht so deutlich unterscheiden). Um das Ergebnis nicht zu verfälschen, wurden jene Testdaten ignoriert, die nur von einem Autor stammen (kein Clustering würde stattfinden und alle vier Verfahren würden die Datenmenge dem einen korrekten Autor zuweisen).

Die bisherige Auswertung beschränkt sich darauf, wie unüberwachte und überwachte Verfahren mit dem Wissen der Ground-Truth-Daten abschneiden. Die Problemstellung der intrinsischen Plagiatserkennung kann jedoch nicht auf Ground-Truth-Daten zurückgreifen und fordert die automatische Ermittlung der Clusteranzahl.

5.2.2 Performance mit relativem Evaluierungskriterium

Die Realisierung mit Wards Methode und der mittleren Varianz innerhalb der Cluster als relatives Evaluierungskriterium hat zu folgenden Ergebnissen geführt.

Gruppiert nach Autorenanzahl

Die Abweichung der ermittelten Clusteranzahl von der tatsächlichen Autorenanzahl wird in Abbildung 5.3 veranschaulicht. Die mittlere Abweichung der gesamten Testdaten beträgt 1,27 mit einer Standardabweichung von 1,44. Es kann beobachtet werden, dass die Performance mit zunehmender Klassenanzahl abnimmt, was sich mit den Ergebnissen der Arbeiten aus Abschnitt 2.4 deckt.

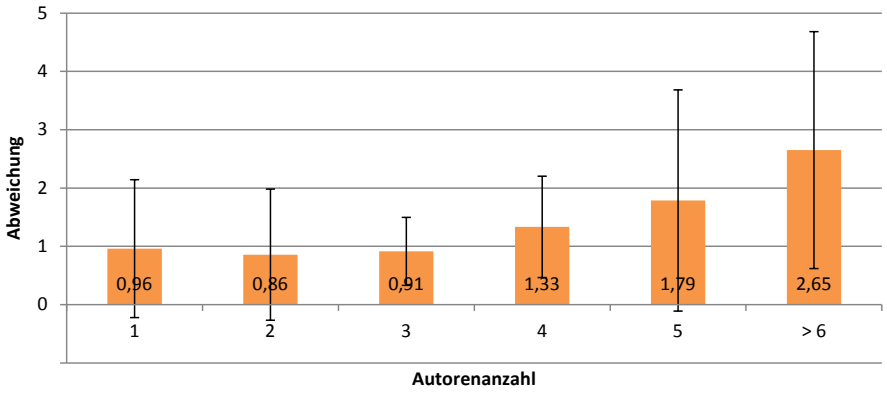


Abbildung 5.3: Abweichung der ermittelten Clusteranzahl $\pm$ σ gruppiert nach Autorenanzahl.

Dasselbe kann bei der Auswertung des F1-Maßes festgestellt werden. Das mittlere F1-Maß der gesamten Testdaten beträgt 0,65 und weist eine Standardabweichung von 0,16

auf. Abbildung 5.4 zeigt eine abnehmende Tendenz bei steigender Autorenanzahl. Der Rand-Index verläuft in Bezug auf die Anzahl der Autoren hingegen relativ stabil und beträgt im Durchschnitt 0,40 – mit einer Standardabweichung von 0,25 (Abbildung 5.5). Jene Testdaten, die nur von einem Autor stammen, wurden für die Berechnung des F1-Maßes und des Rand-Index wiederum nicht berücksichtigt.

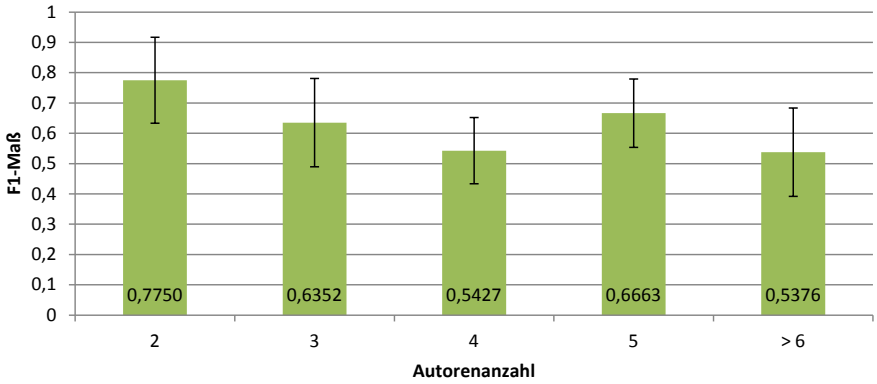


Abbildung 5.4: F1-Maß $\pm$ σ gruppiert nach Autorenanzahl.

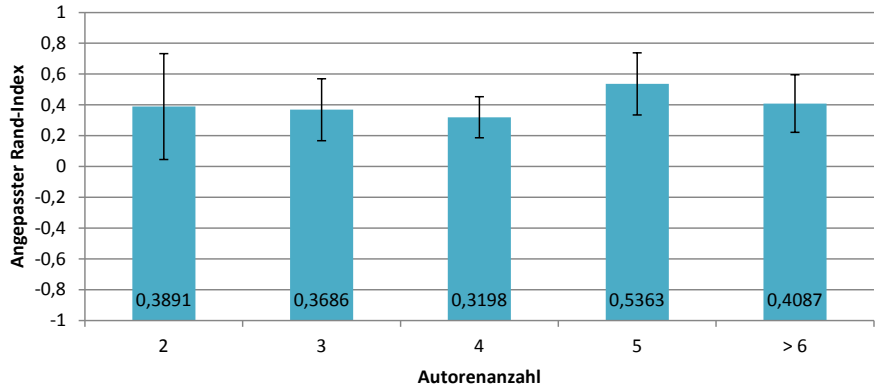


Abbildung 5.5: Angepasster Rand-Index $\pm$ σ gruppiert nach Autorenanzahl.

Gruppiert nach Textlänge

Abbildung 5.6 veranschaulicht die Abweichung der ermittelten Clusteranzahl bezüglich der Länge des Textes (gruppiert nach Satzanzahl-Intervallen). Es kann festgestellt werden, dass die Anzahl der Cluster mit steigender Textlänge zunimmt und von der tatsächlichen Autorenanzahl zunehmend abweicht. Dies deckt sich mit der Auswertung des F1-Maßes, das mit höherer Abweichung der korrekten Klassenanzahl abnimmt (Abbildung 5.7).

Eine mögliche Erklärung dafür könnte sein, dass sich mit zunehmender Textlänge innerhalb eines Autorenclusters weitere Cluster herausbilden, die sich stark genug voneinander

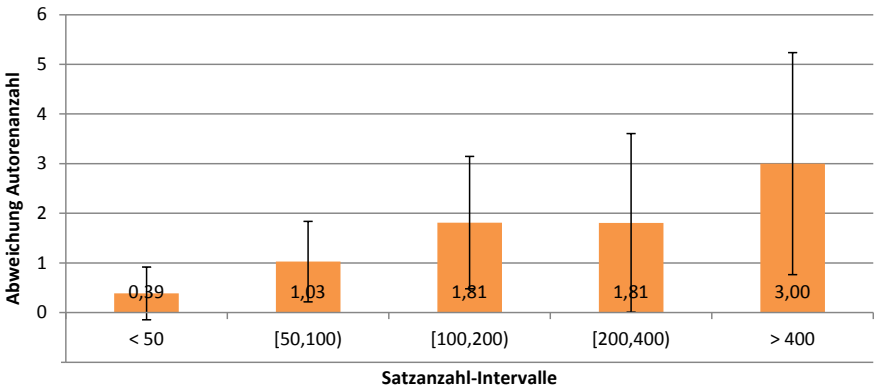


Abbildung 5.6: Abweichung der ermittelten Clusteranzahl $\pm\ \sigma$ bezüglich der Textlänge (gruppiert nach Satzanzahl-Intervallen).

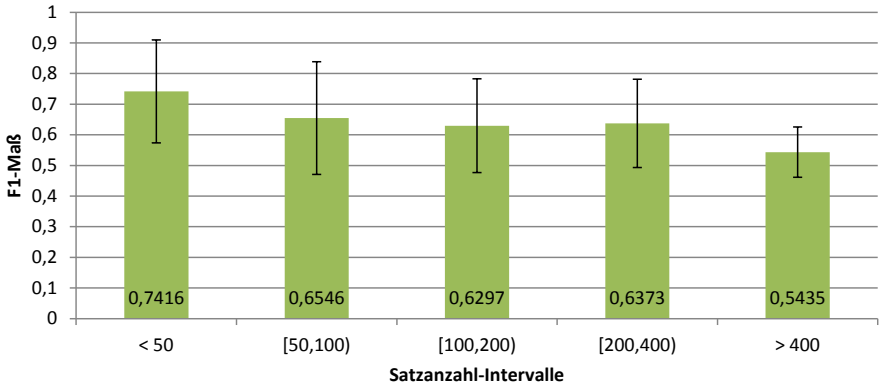


Abbildung 5.7: F1-Maß $\pm\ \sigma$ gruppiert nach Satzanzahl-Intervallen.

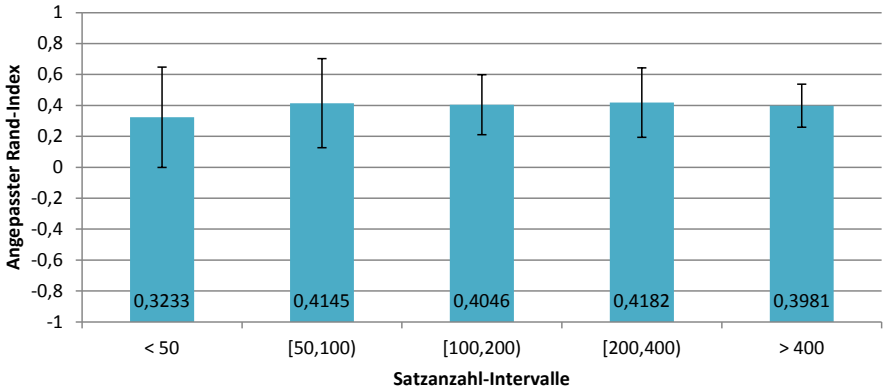


Abbildung 5.8: Angepasster Rand-Index $\pm\ \sigma$ gruppiert nach Satzanzahl-Intervallen).

unterscheiden, sodass sie (fälschlicherweise) als eigenständige Cluster erkannt werden. Die abnehmende Performance bei zunehmender Textlänge könnte auch auf die gleichzeitig steigende Autorenanzahl zurückzuführen sein (Testdokumente mehrerer Autoren sind tendenziell länger). Es kann beobachtet werden, dass der Rand-Index unabhängig von der Textlänge relativ stabil bleibt (Abbildung 5.8).

Performance 1 oder N Autoren

Tabelle 5.2 stellt eine Kontingenztafel dar, welche die Datenmenge in zwei Teilmengen unterteilt und ausdrückt, wie gut zwischen einem und N Autoren unterschieden werden konnte. Knapp die Hälfte der Testdaten einzelner Autoren wurden als solche erkannt (23 von 50). Alle bis auf sechs der insgesamt hundert Texte mehrerer Autoren wurden richtig zugeordnet. Dies führt zu einem F1-Maß von 0,76.

		erkannt als	
		1 Autor	N Autoren
tatsächliche Klasse	1 Autor	23	27
	N Autoren	6	94

Tabelle 5.2: Kontingenztafel: ein oder mehrere Autoren.

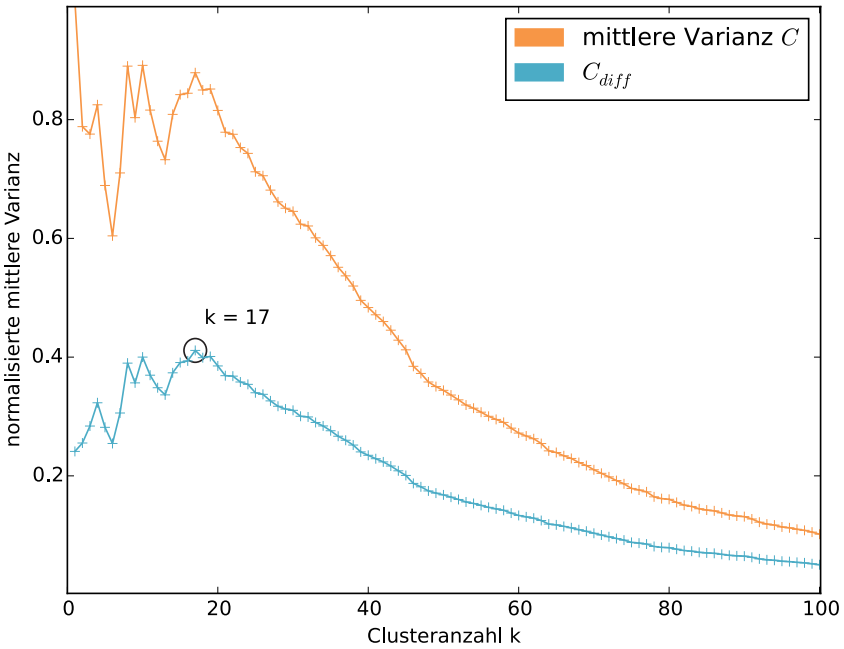


Abbildung 5.9: Knee Point Detection mit globalem Maximum bei $k = 17$ und zwei ähnlich ausgeprägten Punkten an den Stellen $k = 8$ und $k = 10$. Der Testdatensatz wurde von 8 Autoren verfasst, das Qualitätsmaß für die ermittelte Clusteranzahl (17) beträgt 0,21.

Die Ermittlung der Clusteranzahl hängt von der Knee Point Detection (siehe Abschnitt 4.2.5) ab und kann mit den entsprechenden Parametern angepasst werden. Das Verfahren neigt dazu, mit zunehmender Textlänge höher von der korrekten Autorenanzahl abzuweichen. Abbildung 5.9 zeigt den Verlauf der mittleren Varianz eines Testdatensatzes mit 726 Sätzen, der von acht Autoren verfasst wurde und dessen ermittelte Clusteranzahl die höchste Abweichung (aller Testdaten) von der tatsächlichen Autorenanzahl aufweist. Das Maximum der *diff*-Kurve befindet sich an der Stelle $k = 17$ – die nahezu selbe Höhe wird jedoch bereits an den Stellen $k = 8$ und $k = 10$ erreicht, was der korrekten Autorenanzahl eher entspricht. Würde man Lösungen mit höherer Clusteranzahl mehr bestrafen, könnte sich die Lösung bei $k = 8$ als globales Optimum herausbilden. Eine Anpassung der Parameter, die dem Einfluss der Textlänge entgegenwirkt, könnte Gegenstand weiterer Untersuchungen sein.

Qualitätsmaß-Performance

Mit der Auswertung des Qualitätsmaßes in Abbildung 5.10 kann gezeigt werden, dass das Qualitätsmaß mit steigender Abweichung der tatsächlichen Autorenanzahl sinkt. Das Maß spiegelt die Qualität der geschätzten Autorenanzahl wider, wird jedoch erst ab einer Abweichung > 2 deutlich erkennbar und weist eine relativ hohe Standardabweichung auf.

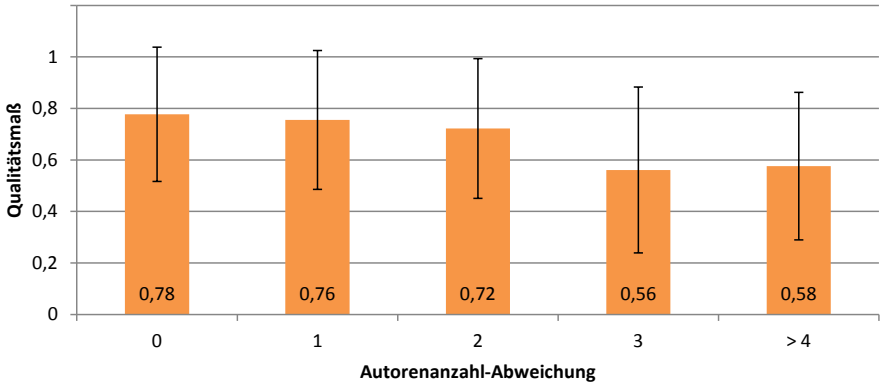


Abbildung 5.10: Qualitätsmaß der ermittelten Clusteranzahl $\pm\ \sigma$. Ab einer Abweichung von 3 ist eine deutliche Senkung erkennbar.

Zusammenfassend kann festgestellt werden, dass die verwendeten Features eine Unterscheidung der Texte verschiedener Autoren ermöglichen. Wird ein überwachtes Verfahren damit trainiert, ist eine sehr hohe Performance möglich (F1-Maß 0,93). Unüberwachte Verfahren schneiden deutlich schlechter ab und können die Aufteilung einer Datenmenge mit der Vorgabe der Clusteranzahl weniger gut annähern (F1-Maß zwischen 0,67 und 0,71). Mit der fallenden mittleren Varianz als relatives Evaluierungskriterium kann die Clusteranzahl gut angenähert werden und weicht im Schnitt 1,27 von der korrekten Anzahl ab. Mit zunehmender Autorenanzahl sinkt die Performance. Diese Tendenz konnte

unter anderem auch von Abbasi & Chen (2008) und Luyckx et al. (2008) beobachtet werden. Außerdem weicht die ermittelte Clusteranzahl (anhand des relativen Evaluierungskriteriums) mit zunehmender Textlänge stärker von der korrekten Autorenanzahl ab.

Kapitel 6

Schlussbetrachtung

Dieses Kapitel fasst die Diplomarbeit zusammen und geht kurz auf die Problemstellung und deren Herausforderungen ein. Es wird beschrieben, welche Ansätze gewählt wurden und wofür sie sich eignen. Abschließend werden weitere Möglichkeiten vorgeschlagen, die Gegenstand zukünftiger Untersuchungen sein könnten.

6.1 Zusammenfassung

Der Prozess der nicht-automatisierten Plagiatsprüfung kann mit einem erheblichen Zeitaufwand verbunden sein. Automatisierte Verfahren stellen wertvolle Instrumente zur Verfügung, die das Erkennen von plagiiertem Text effektiv unterstützen. Konventionelle Plagiatserkennungssysteme vergleichen Textpassagen mit möglichen Originaldokumenten basierend auf übereinstimmende Zeichenketten. Im Gegensatz dazu versucht intrinsische Plagiatserkennung plagiierte Abschnitte anhand stilistischer Merkmale zu erkennen. Dieser Ansatz bildet das intuitive Prüfen eines geübten Lesers nach, dem plötzliche Stilbrüche innerhalb eines Textes auffallen.

Mit dem Ansatz der Stilometrie wird angenommen, dass der Schreibstil eines Autors aufgrund textueller Eigenschaften automatisiert erfasst werden kann und dass Texte unterschiedlicher Autoren anhand ihres Schreibstils unterschieden werden können. Das Ziel dieser Arbeit bestand darin, zu überprüfen, ob Clusterverfahren geeignet sind, eine intrinsische Plagiatserkennung durchzuführen. Die Grundidee bestand darin, dass der untersuchte Text in stilistisch ähnliche Textpassagen gruppiert wird, woraus die Anzahl der Autoren abgeleitet werden kann.

Verschiedene Herausforderungen galt es zu bewältigen: die Generierung geeigneter Testdaten, die Wahl geeigneter Text-Features, die Art der Extraktion und der Dimensionsreduktion, das Clustern der Daten und schlussendlich die Ermittlung der Clusteranzahl und die Bewertung der gewählten Lösung.

Die Testdaten wurden aus Artikeln der englischsprachigen Ausgabe der Wikipedia erzeugt, die den Vorteil haben, dass sie zu ein und demselben Thema von einem oder mehreren Autoren verfasst wurden und dies über die Versionsgeschichte nachvollziehbar ist. Ein Wikipedia-Artikel wird in den meisten Fällen von vielen verschiedenen Autoren erzeugt, wobei der Beitrag eines einzelnen Autors selten die Länge eines kompletten bzw. längeren Artikels umspannt. Dadurch ergibt sich der Nachteil, dass die Verteilung der Testdaten bezüglich der Textlänge unausgeglichen ist und Texte, die nur von einem Autor stammen, tendenziell kürzer sind. Aus diesem Grund wurden zusätzliche Testdaten einzelner Autoren aus Textsegmenten mehrerer Autoren generiert. Dabei wurden Segmente mit höherer Satzanzahl bevorzugt.

Als Text-Features wurden vor allem jene Eigenschaften gewählt, die sich in den untersuchten Arbeiten als effizient erwiesen haben. Anhand eines Sliding-Windows wurde der Text in Segmente unterteilt und die Feature-Extraktion pro Textsegment durchgeführt. Dabei wurde nicht auf einen einheitlichen Feature-Vektor für den gesamten Text gesetzt, sondern für jedes Textsegment wurden individuelle Features extrahiert (die Zusammensetzung der N-Gramm-Features kann sich zwischen verschiedenen Textsegmenten stark unterscheiden). Durch die individuelle Ausprägung der N-Gramme ergab sich eine hohe Anzahl an Features, die durch eine PCA sehr gut reduziert werden konnte.

Um zu erkennen, von wievielen Autoren ein Text stammt, und wie die Textsegmente den jeweiligen Autoren zuzuordnen sind, wurden verschiedene Clusterverfahren getestet. Nachdem mit Verfahren, die die Anzahl der Cluster selbständig bestimmen, kein positives Ergebnis erzielt werden konnte, wurden Verfahren mit vorgegebener Clusteranzahl untersucht. Da die tatsächliche Klassenaufteilung der Daten damit gut angenähert werden konnte, wurde für die automatisierte Ermittlung der Clusteranzahl das am besten abschneidende Clusterverfahren (agglomeratives Clustering mit Wards Methode) mit einem relativen Evaluierungskriterium (mittlere Varianz innerhalb der Cluster im Vergleich zur Anzahl der Cluster) verwendet. Eine Knee Point Detection bestimmte anschließend jenen Punkt der Kurve, der die stärkste Änderung ausdrückt.

Es hat sich gezeigt, dass die Performance mit steigender Autorenanzahl, aber auch mit zunehmender Länge des Textes sinkt und die ermittelte Clusteranzahl stärker von der tatsächlichen Autorenanzahl abweicht. Ersteres deckt sich mit den Ergebnissen aus den untersuchten Arbeiten in Abschnitt 2.4. Letzteres könnte auf die Art der Knee Point Detection zurückzuführen sein. Eine Berücksichtigung der Textlänge oder der Varianz des gesamten Datensatzes könnte Gegenstand zukünftiger Untersuchungen sein: Die Varianz wurde einer Min-Max-Normalisierung unterzogen, ohne die ursprüngliche Höhe der Varianz in die Parameter der Knee Point Detection einfließen zu lassen (ein Datensatz mit höherer Textlänge weist prinzipiell eine höhere Varianz auf als ein Datensatz, der kürzer ist).

Ob sich die verwendete Methode als intrinsische Plagiatserkennung eignet, sollte kritisch betrachtet werden. Grundsätzlich wird die korrekte Klassenaufteilung gut angenähert. Trotz der abfallenden Performance bei höherer Autorenanzahl und steigender Textlänge

wird die korrekte Anzahl im Schnitt nur um 1,27 verfehlt. Verglichen mit der korrekten Klassenaufteilung wird ein mittleres F1-Maß von 0,65 und ein angepasster Rand-Index von 0,40 erreicht. Für den Einsatz als Plagiatserkennungsverfahren scheinen diese Werte jedoch zu niedrig zu sein, um zuverlässige Aussagen über die Autorenschaft eines Textes treffen zu können. Mit Vorgabe der korrekten Clusteranzahl kommt Wards Methode im Vergleich auf ein F1-Maß von 0,71 und einen angepassten Rand-Index von 0,46. Werden die Daten mit einer SVM trainiert, steigt die Performance auf einen mittleren F1-Wert von 0,93.

Für die folgenden zwei Anwendungsbeispiele ist die beschriebene Vorgehensweise gut geeignet:

- Die Anzahl der Autoren ist bekannt und es stellt sich die Frage, welche Teile des Textes stilistisch zusammenpassen.
- Die Autorschaft von Teilen des Textes ist bekannt und es stellt sich die Frage, welchen der Autoren die Segmente unbekannter Autorenschaft zugeordnet werden können.

Wenn keine zusätzlichen Informationen bekannt sind, wie dies bei der intrinsischen Plagiatserkennung der Fall ist, sind die Ergebnisse des Verfahrens akzeptabel. Die Unterscheidung, ob ein Text von einem oder von mehreren Autoren verfasst wurde, resultiert in einem F1-Maß von 0,76, wobei hervorgehoben werden muss, dass nur knapp die Hälfte der Testdaten einzelner Autoren und nahezu die gesamten Testdaten mehrerer Autoren als solche erkannt wurden. Für die effizientere Ermittlung der Clusteranzahl ist das Lösen von der Abhängigkeit der Textlänge erforderlich und für den praktikablen Einsatz als intrinsische Plagiatserkennung entscheidend.

Neben den gewählten Ansätzen gibt es weitere Möglichkeiten, die Problemstellung der intrinsischen Plagiatserkennung zu untersuchen.

6.2 Ausblick

Die Möglichkeiten alternativer Ansätze sind vielfältig. Neben der bereits beschriebenen Problematik der Länge von Texten, die nur von einem Autor stammen, muss ein weiterer Punkt hervorgehoben werden: Die verwendeten Testdaten bestehen aus Textsegmenten, die von verschiedenen Autoren selbst verfasst wurden. Interessant wäre zudem die Untersuchung von Texten, die ein Autor von einem anderen Autor übernommen und umgeschrieben hat. Dadurch könnte der reale Fall eines Plagiats besser abgebildet werden.

Weitere Möglichkeiten ergeben sich bei der Wahl der Text-Features. Neben den beschriebenen idiosynkratischen Features (Abschnitt 3.1.1), die nicht zum Einsatz gekommen sind, scheint der auf Synonymen basierende Ansatz von Koppel et al. (2006) und Clark & Hannon (2007) aus Abschnitt 2.4 vielversprechend. Eine komplexere Verarbeitung der Syntax eines Satzes (z.B. Nominal-, Verbalphrasen etc.) oder semantisches Verstehen

eines Textes könnten dazu beitragen, die Unterscheidung von Schreibstilen verschiedener Autoren zu verbessern.

Für die Berechnung der Distanz zwischen den Datenpunkten wurde die euklidische Distanz verwendet. Es wäre möglich, die Eignung weiterer Distanz- oder Ähnlichkeitsmaße zu überprüfen. Vor allem Methoden, welche die Ähnlichkeit aufgrund der Anzahl gemeinsamer Nachbarpunkte (Shared Nearest Neighbor, SNN) messen, könnten sich für die vorliegende Problemstellung eignen: Houle et al. (2010) stellen fest, dass SNN-Metriken besonders für hochdimensionale Daten mit geringer Anzahl an Datenpunkten nützlich sind.

Die Schwierigkeit der Ermittlung der Clusteranzahl könnte umgangen werden, indem die Parametrisierung der Verfahren (welche die Anzahl selbst bestimmen) automatisiert wird. Mit DBSCAN konnten für einzelne Testdaten beispielsweise gute Ergebnisse erzielt werden. Es könnte untersucht werden, ob das Schätzen der geeigneten Parameter anhand der Eigenschaften der Datensätze möglich ist.

Im Zusammenhang mit hierarchischem Clustering wäre zudem eine Analyse der Clusterbildung möglich. Es konnte beobachtet werden, dass ein korrekter Cluster schrittweise aus einzelnen Datenpunkten bzw. kleineren Clustergruppen zusammengesetzt wird. Ein Fusionieren größerer Gruppen deutet im Umkehrschluss auf zwei eigenständige Cluster hin, die nicht zusammengeführt werden sollen. Eine Möglichkeit bestünde darin, das Dendrogramm von oben nach unten zu durchlaufen, und jene Cluster, die eine minimale Anzahl an Datenpunkten unterschreiten, nicht vom übergeordneten Cluster abzutrennen. Sie können als Außenseiter innerhalb des übergeordneten Clusters gesehen werden und stellen möglicherweise eine natürliche Grenze der Clusteraufteilung dar.

Die intrinsische Plagiatserkennung ist eine noch sehr junge Disziplin. Es wird mit diversen Techniken und Vorgehensweisen experimentiert – mit der vorliegenden Arbeit wurde der Ansatz einer Clusteranalyse verfolgt. Es konnte gezeigt werden, dass Texte verschiedener Autoren aufgrund ihres Schreibstils unterschieden werden können und dass die Autorenzusammensetzung innerhalb eines Textes – wenn auch ausbaufähig – gut angenähert werden kann. Entscheidend scheinen dabei die textuellen Features zu sein, die verschiedenartige Eigenschaften von Text widerspiegeln. Entwicklungen auf dem Gebiet der NLP könnten in diesem Zusammenhang die Voraussetzungen schaffen, komplexere Merkmale zu erfassen und weitere stilistische Charakteristika offenzulegen.

Abkürzungsverzeichnis

AIC Akaikes Informationskriterium

BIC Bayessche Informationskriterium

DP Dirichlet-Prozess

EM Expectation Maximization

GMM Gaußsches-Mixture-Model

HMM Hidden Markov Model

IR Information Retrieval

KLT Karhunen-Loève-Transformation

ML Machine Learning

NLP Natural Language Processing

NLTK Natural Language Toolkit

PCA Principal Component Analysis

POS Part-of-Speech

SNN Shared Nearest Neighbor

SOM Self Organizing Map

SQA Summe der quadratischen Abweichung

SVM Support Vector Machine

TTR Type-Token-Relation

Literaturverzeichnis

Abbasi, Ahmed & Hsinchun Chen (2008), "Writeprints: A Stylometric Approach to Identity-Level Identification and Similarity Detection in Cyberspace." *ACM Transactions on Information Systems*, 26, 7:1–29.

Adamas, Douglas (1979), *The Hitch Hiker's Guide to the Galaxy*. Pan Books, London.

Aichele, Dieter (2005), "Das Werk von W. Fucks." In *Quantitative Linguistik*, chapter I, 152–158, Walterde Gruyter, Worms.

Alzahrani, Salha M, Naomie Salim, Ajith Abraham, & Senior Member (2012), "Understanding Plagiarism Linguistic Patterns, Textual Features, and Detection Methods." *IEEE Transactions On Systems, Man, And Cybernetics - Part C: Applications And Reviews*, 42, 133–149.

Bauerlein, Mark, Mohamed Gad-el Hak, Wayne Grody, Bill McKelvey, & Stanley W. Trimble (2010), "We Must Stop the Avalanche of Low-Quality Research." *The Chronicle of Higher Education*, June 13.

Bird, Steven, Ewan Klein, & Edward Loper (2009), *Natural Language Processing with Python*. O'Reilly Media, Inc., Sebastopol.

Bishop, Christopher M (2006), *Pattern Recognition and Machine Learning*. Springer, Cambridge.

Blei, David M. & Michael I. Jordan (2006), "Variational inference for Dirichlet process mixtures." *Bayesian Analysis*, 1, 121–143.

Brennan, Michael, Sadia Afroz, & Rachel Greenstadt (2012), "Adversarial stylometry." *ACM Transactions on Information and System Security*, 15, 1–22.

Büttcher, Stefan, Charles L. A. Clarke, & Gordon V. Cormack (2010), *Information Retrieval: Implementing and Evaluating Search Engines*. MIT Press.

Cambria, Erik & Bebo White (2014), "Jumping NLP Curves: A Review of Natural Language Processing Research." *IEEE Computational Intelligence Magazine*, 9, 48–57.

Chong, Miranda, Specia Lucia, & Mitkov Ruslan (2010), "Using Natural Language Processing for Automatic Detection of Plagiarism." *International Integrity & Plagiarism Conference 2010*.

Clark, Jonathan H & Charles J Hannon (2007), "A Classifier System for Author Recognition Using Synonym-Based Features." *Lecture Notes in Computer Science*, 4827, 839–849.

Collins, Jeff, David Kaufer, Pantelis Vlachos, Brian Butler, & Suguru Ishizaki (2004), "Detecting Collaborations in Text." *Computers and the Humanities*, 38, 15–36.

Collins, Michael (1997), "The EM Algorithm." *Technical Report, Departement of Computer and Information Science, University of Pennsylvania*.

Damakos, Bence Dániel (2011), "Featured article word cloud." Online erhältlich unter http://endami.blogspot.co.at/2011/06/featured-article-work-cloud.html, abgerufen am 01.09.2014.

Davis, David L. & Donald W. Bouldin (1979), "A Cluster Separation Measure." *IEEE Transactions On Pattern Analysis And Machine Intelligence*, PAMI-1, 224–227.

Dempster, A. P., N. M. Laird, & D. B. Rubin (1977), "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society. Series B (Methodological)*, 39, 1–38.

Donath, W. E. & A. J. Hoffman (1973), "Lower Bounds for the Partitioning of Graphs." *IBM J. Res. Develop.*, 17, 420 – 425.

Duda, Richard O., Peter E. Hart, & David G. Stork (2001), *Pattern Classification*, 2nd edition. Wiley-Interscience, New York.

Dunn, J C (1973), "A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters." *Journal of Cybernetics*, 3, 32–57.

Errami, Mounir & Harold Garner (2008), "A tale of two citations." *Nature*, 451, 397–399.

Ester, Martin, Hans-peter Kriegel, Jörg Sander, & Xiaowei Xu (1996), "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise." *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland*, 226–231.

Ferguson, Thomas S. (1973), "A Bayesian Analysis of Some Nonparametric Problems." *The Annals of Statistics*, 1, 209–230.

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

Forsyth, R. & D. Holmes (1996), "Feature-Finding for Text Classification." *Literary and Linguistic Computing*, 11, 163–174.

Frey, Brendan J & Delbert Dueck (2007), "Clustering by passing messages between data points." *Science*, 315, 972–976.

Frigyik, Bela A, Amol Kapila, & Maya R Gupta (2010), "Introduction to the Dirichlet Distribution and Related Processes." *Technical Report UWEETR, Department of Electrical Engineering, University of Washington*, 0006.

Fucks, Wilhelm (1962), "Mathematical analysis of formal structure of music." *IRE transactions on information theory*, 8, 225–228.

Gockel (2008), *Form der wissenschaftlichen Ausarbeitung*. Springer-Verlag, Heidelberg. Begleitende Materialien unter `http://www.formbuch.de`.

Goutte, C, L K Hansen, M G Liptrot, & E Rostrup (2001), "Feature-space clustering for fMRI meta-analysis." *Human brain mapping*, 13, 165–83.

Goutte, C, P Toft, E Rostrup, F Nielsen, & L K Hansen (1999), "On clustering fMRI time series." *NeuroImage*, 9, 298–310.

Graham, Neil, Graeme Hirst, & Bhaskara Marthi (2005), "Segmenting documents by stylistic character." *Natural Language Engineering*, 11, 397–415.

Gruner, Stefan & Stuart Naven (2005), "Tool Support for Plagiarism Detection in Text Documents." *Proceedings of the ACM Symposium on Applied Computing*, 776–781.

Grzybek, Peter & Emmerich Kelih (2005), "Zur Vorgeschichte quantitativer Ansätze in der russischen Sprach- und Literaturwissenschaft." In *Quantitative Linguistik*, chapter I, 23–64, Walterde Gruyter, Graz.

Hastie, Trevor, Robert Tibshirani, & Jerome Friedman (2009), *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd edition. Springer.

Hoover, David L (2003), "Another Perspective on Vocabulary Richness." *Computers and the Humanities*, 37, 151–178.

Houle, Michael E, Hans-Peter Krigel, Peer Kröger, Erich Schubert, & Arthur Zimek (2010), "Can Shared-Neighbor Distances Defeat the Curse of Dimensionality?" *Proceedings of the 22nd International Conference on Scientific and Statistical Database Management, Heidelberg Germany*.

Hunter, John D (2007), "Matplotlib: A 2D Graphics Environment." *Computing in Science & Engineering*, 9, 90–95.

Iqbal, Farkhund, Hamad Binsalleeh, & Benjamin C M Fung (2009), "Mining Writeprints from Anonymous E-mails for Forensic Investigation."

IThenticate (2013), "iThenticate - Plagiarism Detection Software: Misconceptions."

Jaccard, Paul (1901), "Étude comparative de la distribution florale dans une portion des Alpes et du Jura." *Bulletin de la Société Vaudoise des Sciences Naturelles*, 37, 547–579.

Jones, Michael (2009), "Back-translation : the latest form of plagiarism." *4th Asia Pacific Conference on Educational Integrity (4APCEI), September 2009.*

Juola, Patrick (2004), "On composership attribution." *Proceedings of 2004 Joint International Conference of the Association for Literary and Linguistic Computing and the Association for Computers and the Humanities (ALLC/ACH 2004) Göteborg, Sweden.*

Juola, Patrick (2007), "Authorship Attribution." *Foundations and Trends in Information Retrieval*, 1, 233–334.

Juola, Patrick (2009), "JGAAP : A System for Comparative Evaluation of Authorship Attribution." *Journal of the Chicago Colloquium on Digital Humanities and Computer Science*, 1, 1–5.

Kiss, Tibor, Jan Strunk, & Ruhr-universität Bochum (2006), "Unsupervised Multilingual Sentence Boundary Detection." *Computational Linguistics*, 32.

Koppel, Moshe, Navot Akiva, & Ido Dagan (2006), "Feature Instability as a Criterion for Selecting Potential Style Marke." *J. Amer. Soc. Inf. Sci. Technol.*, 57, 1519–1525.

Kriegel, Hans-Peter, Peer Kröger, Jörg Sander, & Arthur Zimek (2011), "Density-based clustering." *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1, 231–240.

Lakshmi & Pushpendra Kumar Pateriya (2012), "A Study on Author Identification through Stylometry." *International Journal of Computer Science & Communication Networks*, 2, 653–657.

Li, Jiexun, Rong Zheng, & Hsinchun Chen (2006), "From Fingerprint To Writeprint." *Communications of the ACM*, 49, 76–82.

Lloyd, S. (1957), "Least squares quantization in PCM." *Technical report, Bell Laboratories. Published in 1982 in IEEE Transactions on Information Theory*, 28, 129–137.

Luxburg, Ulrike Von (2007), "A Tutorial on Spectral Clustering." *Technical Report No. TR-149, Max Planck Institute for Biological Cybernetics.*

Luyckx, Kim, Walter Daelemans, Alexander Hamilton, & James Madison (2008), "Authorship Attribution and Verification with Many Authors and Limited Data." *Proceedings of the 22nd International Conference on Computational Linguistics, Manchester*, 513–520.

Lyon, Caroline, Ruth Barrett, & James Malcolm (2004), "A theoretical basis to the automated detection of copying between texts , and its practical implementation in the Ferret plagiarism and collusion detector ." *Plagiarism: Prevention, Practice and Policies Conference 2004*.

Manning, Christopher D & Hinrich Schütze (1999), *Foundations of Statistical Natural Language Processing*. Massachusetts Institute of Technology, London.

Manning, Christopher D., Hinrich Schütze, & Prabhakar Raghavan (2008), *An Introduction to Information Retrieval*. Cambridge University Press.

McLachlan, G. J., S. K. Ng, & D. Peel (2003), "On Clustering by Mixture Models." *Studies in Classification, Data Analysis, and Knowledge Organization: Exploratory Data Analysis in Empirical Research*, 141–148.

Mendenhall, T.C. (1887), "The characteristic curves of composition." *Science*, IX.

Meyer, Sven & Benno Stein (2006), "Intrinsic Plagiarism Detection." *Proceedings of the 28th European Conference on IR Research, ECIR 2006*, 1–4.

Mosteller, Frederick & David L. Wallace (1963), "Inference in an Authorship Problem." *Journal of the American Statistical Association*, 58, 275–309.

Murphy, Kevin P (1998), "Fitting a Conditional Gaussian Distribution." *Technical Report*.

Murphy, Kevin P (2012), *Machine Learning - A Probabilistic Perspective*. The MIT Press, Palo Alto.

Orlov, Y. K. (1983), "Ein Modell der Häufigkeitsstruktur des Vokabulars." In *Studies on Zipf's Law*, 154–233, Bochum.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, & E. Duchesnay (2011), "Scikit-learn : Machine Learning in Python." *Journal of Machine Learning Research*, 12, 2825–2830.

Porter, M.F. (1980), "An algorithm for suffix stripping." *Program*, 14, 130–137.

Rand, William M . (1971), "Objective Criteria for the Evaluation of Clustering Methods." *Journal of the American Statistical Association*, 66, 846–850.

Rasmussen, Carl Edward (2000), "The Infinite Gaussian Mixture Model." *Advances in Neural Information Processing Systems*, 12, 554–560.

Reichmann, Gerhard (2013), "Textplagiate in der Wissenschaft und deren Verhinderung." *Information. Wissenschaft & Praxis*, 64, 179–181.

Reynolds, Douglas (2008), "Gaussian Mixture Models." *Encyclopedia of Biometric Recognition, Springer*, 694–699.

Rockmore, Daniel (2006), "The Style of Numbers Behind a Number of Styles." *The Chronicle of Higher Education - The Chronicle Review*, 52, 10.

Rousseeuw, Peter J (1987), "Silhouettes : a graphical aid to the interpretation and validation of cluster analysis." *Journal of Computational and Applied Mathematics*, 20, 53–65.

Rudman, Joseph (1998), "The State of Authorship Attribution Studies : Some Problems and Solutions." *Computers and the Humanities*, 31, 351–365.

Santorini, Beatrice (1990), "Part-of-Speech Tagging Guidelines for the Penn Treebank Project."

Savoy, Jacques (2012), "Authorship Attribution Based on Specific Vocabulary." *ACM Transactions on Information Systems*, 30, 1–30.

Schulstad, Ida, Mark Boga, Cranston Jordan, Kara Pally, Richard Destefano, John Stewart, Charles Tappert, James Madison, Alexander Hamilton, & John Jay (2012), "Evaluation of a Stylometry System on Various Length Portions of Books." *Proceedings of Student-Faculty Research Day, SCSIS, Pace University, May 4th, 2012*, D5.1–8.

Seo, Mira J. (2009), "Plagiarism and Poetic Identity in Martial." *American Journal of Philology*, 130, 567–593.

Smith, Lindsay I (2002), *A tutorial on Principal Components Analysis*. Online verfügbar unter `http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf`, abgerufen am 24.10.2014.

Stamatatos, Efstathios (2009), "A Survey of Modern Authorship Attribution Methods." *Journal of the American Society for Information Sience and Technology*, 60, 538–556.

Stamatatos, Efstathios & Gerhard Widmer (2002), "Music Performer Recognition Using an Ensemble of Simple Classifiers." *Proceedings of the 15th European Conference on Artificial Intelligence, Lyon, France*.

Stappenbelt, Brad & Chris Rowles (2009), "The effectiveness of plagiarism detection software as a learning tool in academic writing education." *4th Asia Pacific Conference on Educational Integrity (4APCEI), September 2009*.

Tan, Pang-Ning, Michael Steinbach, & Vipin Kumar (2005), *Introduction to Data Mining*, 1st edition. Addison Wesley.

Tufféry, Stéphane (2011), *Data Mining and Statistics for Decision Making*, 2nd edition. John Wiley & Sons.

Tweedie, Fiona J. (2005), "Statistical models in stylistics and forensic linguistics." In *Quantitative Linguistik*, chapter VI., 387–397, Walterde Gruyter.

Tweedie, Fiona J & R Harald Baayen (1998), "How Variable May a Constant be? Measures of Lexical Richness in Perspective." *Computers and the Humanities*, 32, 323–352.

van der Walt, Stéfan, S Chris Colbert, & Gaël Varoquaux (2011), "The NumPy Array: A Structure for Efficient Numerical Computation." *Computing in Science & Engineering*, 13.

Ward, Joe H. Jr. (1963), "Hierarchical Grouping to Optimize an Objective Function." *Journal of the American Statistical Association*, 58, 236–244.

Webb, Andrew R (2002), *Statistical Pattern Recognition, Second Edition*. Wiley.

Witten, Ian H., Eibe Frank, & Mark A. Hall (2011), *Data Mining: Practical Machine Learning Tools and Techniques*, 3rd edition. Morgan Kaufmann, Burlington.

Yule, G U (1944), *The Statistical Study of Literary Vocabulary*. Cambridge University Press.

Zechner, Mario, Markus Muhr, Roman Kern, & Michael Granitzer (2009), "External and Intrinsic Plagiarism Detection Using Vector Space Models." *Proc. SEPLN*, 32, 47–55.

Zhao, Qinpei, Mantao Xu, & Pasi Fränti (2008), "Knee Point Detection on Bayesian Information Criterion." *2008 20th IEEE International Conference on Tools with Artificial Intelligence*, 431–438.