Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Florian Schlosser

Entwicklung eines lernfähigen Systems zur Erzeugung von Kurzzusammenfassungen auf Basis statistischer Methoden aus der Künstlichen Intelligenz

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Florian Schlosser

Entwicklung eines lernfähigen Systems zur Erzeugung von Kurzzusammenfassungen auf Basis statistischer Methoden aus der Künstlichen Intelligenz

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung

im Studiengang Bachelor of Science Technische Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Michael Neitzke
Zweitgutachter: Prof. Dr. Stephan Pareigis

Eingereicht am: 13. Juni 2014

Florian Schlosser

Thema der Arbeit
Entwicklung eines lernfähigen Systems zur Erzeugung von Kurzzusammenfassungen auf Basis statistischer Methoden aus der Künstlichen Intelligenz

Stichworte
Künstliche Intelligenz, Automatisierte Zusammenfassung, Statistische Methoden, Sentiment Analyse

Kurzzusammenfassung
Diese Arbeit beginnt mit einer kurzen Einführung in das Thema der automatisierten Erzeugung von Kurzzusammenfassungen. Danach werden die grundlegenden Verfahren vorgestellt, die in dem entwickelten System verwendet wurden. Hierauf folgt eine detailiertere Beschreibung der Systemkomponenten, und die Validierung des Systems. Am Schluss werden in einem Fazit die Erkenntisse aus der Validierung näher betrachtet und eine Bewertung abgegeben.

Florian Schlosser

Title of the paper
Development of a trainable Summarization System based on statistical method

Keywords
Artificial Intelligence, Automatic Summarization, Statistical Methods,

Abstract
This thesis begins with an introduction to the field of automated summarization, followed by explanation of the fundamental algorithms. Then it illustrates the components of the system and it's validation. At the end it will lead a conclusion and interpret the final results of the validation.

Inhaltsverzeichnis

1 Einleitung

1.1 Vergangenheit

Die Extrahierung von wichtigen Informationen aus Texten hat in der Informatik eine lange Geschichte. Bereits 1945 veröffentlichte Vannevar Bush *[Bush,1945]* einen Artikel über ein Informationsaufbereitungs- und Verarbeitungssystem. Es sollte als Wissensspeicher dienen und Verbindungen zwischen Informationen erlauben. Dieses System war zwar rein fiktiver Natur und wurde nie direkt umgesetzt, die Idee hat allerdings viel Anklang gefunden. In den 50er Jahren gab es einen großen Schub in der Künstlichen Intelligenz, der zu großen Versprechungen und Ankündigungen führte, welche dann allerdings später mit der ernüchternden Realität einen Dämpfer erhielt. Eine vielzitierte Aussage von Herbert Simon 1957 lautete:

> *Ich will Sie nicht überraschen oder schockieren - aber am einfachsten kann ich zusammenfassend berichten, dass es heute Maschinen gibt die denken, die lernen und die kreativ sind. Darüber hinaus wächst ihre Fähigkeit, diese Dinge zu tun, zügig weiter, bis - in absehbarer Zukunft - der Bereich der Aufgabenstellungen, mit denen sie zurechtkommen, genau so groß ist wie der Bereich, den der menschliche Versand bewältigen kann.*[RN,2012]

Auch wenn der Begriff "in absehbarer Zukunft" verschieden ausgelegt werden kann, weiß man heute, dass diese Prognosen für die damalige Zeit etwas zu hoch gesteckt waren. Der Grund für diese anfängliche Euphorie waren wohl die guten Ergebnisse, welche die damaligen Systeme bei einfachen Problemstellungen erbracht haben. Es hat sich allerdings gezeigt, dass diese Systeme unbrauchbar wurden, sobald die Aufgabenstellungen und Problemfelder erweitert oder komplizierter wurden *[RN,2012]*?. Trotz dessen wurden auch einige Erkenntnisse hervorgebraucht, die sogar heute noch

hoch aktuell sind. Diese sind zum Beispiel Luhn's Ansatz zur Information Retrieval oder Eugene Garfields Arbeiten an Zitierindices. Gerade ersterer wird im Laufe dieser Arbeit noch eine wichtige Rolle spielen, welcher dann zu einem späteren Zeitpunkt weiter vertieft wird.

Im Laufe der folgenden Jahren entstanden die ersten kommerziellen Systeme im Bereich Information Retrieval. Diese waren beispielsweise das ISI (Institute for Scientific Information) oder die von Siemens entwickelten GOLEM und PASSAT. In den weiteren Jahrzehnten gab es zwar viele weitere Systeme, diese haben allerdings eher Neuerung in systematischer Hinsicht als in fundamentaler mit sich gebracht. In den 90er Jahren, als das Internet immer größer wurde, wuchs auch die Möglichkeit neuer Textverarbeitungsanwendungen. Das Internet hatte nicht nur positive Aspekte zum Vorschein gebracht, es tauchten nun auch Problemfelder auf, für die Lösungen entwickelt werden mussten. Es erschlossen sich also neue Anwendungen und somit neue Ansätze. Probleme und Anwendungen dieser Natur waren zum Beispiel die Flut von Informationen, das Analysieren von Spam-Mails, das Entwickeln von Suchmaschinen und das Auswerten von Benutzerdaten. Ende der 90er und in den 2000ern wurden einige Projekte wie die DUC (Document Understanding Conferences) oder die TREC (Text REtrieval Conference) "ins Leben gerufen" die zu einem Wissensaustausch in dieser Disziplin anregten und Arbeitsgrundlagen anzubieten.

Zusammenfassend kann man über den Verlauf der letzten 60 Jahre sagen, dass die wesentlichsten Erkenntnisse bereits zu Beginn erschlossen wurden, und ab den 90er Jahren die Systeme zwar durch die steigende Leistung von Computern und ausgereifteren Architekturen ihre Qualität und Einsatzmöglichkeiten stark ausbauen konnten, jedoch die Neuerungen weniger fundamentaler sondern mehr konzeptioneller Natur waren.

1.2 Gegenwart

In diesem Abschnitt soll mehr auf die heute gängigeren Konzepte eingegangen werden. Die hier vorgestellten Konzepte werden den Fokus auf TS-Systeme (Text Summarization Systeme) haben, um mehr auf das Thema dieser Arbeit einzugehen.

1.2.1 Systementwurf

Laut Spärck Jones [K.S. Jones, 2007] hat sich als Struktur für TS-Systeme der folgende Ansatz sehr häufig bewährt, welcher die Aufgliederung des Systems in drei Hauptteile unterteilt:

Interpretation - Die Interpretation des Quelltextes zu einer Quelltextrepräsentation (POS, Lemmatising, Sentencesplitting, usw...).
Transformation - Die Umwandlung der Quelltextrepräsentation zu einer Zusammenfassungsrepräsentation (Analyse der wichtigen Features, usw...).
Generation - Die Erzeugung des eigentlichen Textes der Zusammenfassung aus der Zusammenfassungsrepräsentation.

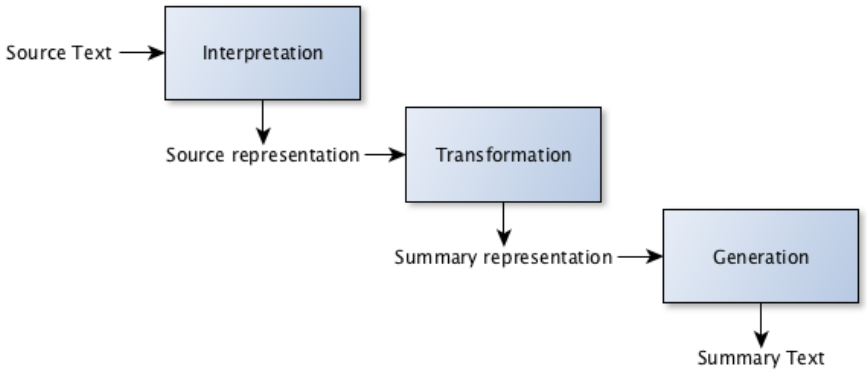

Abbildung 1.1: Ein bewährter Systemaufbau.

Dieser Aufbau des Systems hat mehrere Vorteile, welche den Entwicklern Flexibilität in einigen wichtigen Bereichen bietet. Ein paar werde hier kurz aufgelistet:

1. Der Entwickler hat durch die Interpretationsphase die Möglichkeit diverse Vorverarbeitungen, wie Säuberungen, Part-Of-Speech-Tagging, usw., an den Texten durchzuführen.

2. Durch die Trennung von Transformation und Generation, wird die Möglichkeit gegeben, den Algorithmus für die Erfassung der wichtigen Informationen, von dem Algorithmus für die Erzeugung der finalen Kurzzusammenfassung zu treffen. Dieses Vorgehen erleichtert, das Testen und Validieren der einzelnen Verarbeitungsschritte.

1.2.2 Summarising Factors

Das System wie gerade beschrieben, stellt einen geradlinigen Prozess, ohne ersichtliche äußere Einflüsse, zur Generierung einer Kurzzusammenfassung dar. Häufig jedoch ist es erforderlich äußeren Einflüsse zu gewähren, damit die erzeugte Kurzzusammenfassung bestimmte Aufgaben erfüllen. Solche Aufgaben könnten zum Beispiel, das Beantworten von Fragen, die Ausrichtung auf bestimmte Zielgruppen oder Ähnliches sein. Diese Einflüsse haben starke Auswirkungen auf den Entwurf und die Evaluierung eines solchen Systems, und werden von nun als Summarising Factors [K.S. Jones, 2001] bezeichnet. Um diese Summarising Factors besser handhaben zu können, werden diese in drei Kategorien eingeteilt:

1. **Input Factors** - Die Input Factors beschreiben die Eigenschaften des Quelltextes (Sprache, Struktur, Medium, ...).

2. **Purpose Factors** - Die Purpose Factors beschreiben, wozu die Zusammenfassung dienen soll (Fragen beantworten, Zielgruppe, ...).

3. **Output Factors** - Die Output Factors beschreiben die Charakteristika des Ergebnisses (Sprache, Format (z.B. Absätze oder Liste mit Kerninhalten), ...).

Die Input Factors bilden dabei den Kontext für die Interpretationsphase und die Output Factors bilden den Kontext für die Generationsphase. Die Purpose Factors bilden in erster Linie den Kontext für die Transformationsphase, haben aber unter Umständen auch einen Einfluss auf die Generationsphase, da der eigentlich angestrebt Zweck (z.B. Fragen beantworten) Auswirkungen auf das Format der Zusammenfassung haben kann.

1.2.3 Evaluierung

Das Problem der Evaluierung von TS-Systemen ist schon seit längerer Zeit bekannt, und hat sich über die letzten 50 Jahre zunehmend geändert. Zu Beginn war der Ansatz, zur Erstellung von Kurzzusammenfassungen, dass bestimmte Sätze, die von einem Algorithmus ausgesucht wurden und dann gänzlich in die Kurzzusammenfassung kopiert wurden. Dieser Ansatz hatte zum Vorteil, dass, ohne Mehraufwand, grammatikalisch gute Sätze entstanden, und dies keine weitere Evaluierung für grammatikalische Belange benötigt hat. Das Evaluieren bezog sich also lediglich auf den Algorithmus selbst. Neben diesem Ansatz gab es im Laufe der Zeit noch weitere, da es den Sätzen untereinander manchmal an Zusammenhang fehlte, und man sich mit solchen Limitierungen nicht zufriedengeben wollte. Dies hatte zur Folge, dass nicht nur der Algorithmus zur Bestimmung, was in die Kurzzusammenfassung gehört, evaluiert werden musste, sondern auch der Teil des Systems, welcher den Kurzzusammenfassungstext generiert. Da die Generierung des Kurzzusammenfassungstextes nach dem eigentlichen Algorithmus stattfindet, und somit in die, aus heutiger Sicht, Generationsphase gehört, ist man von dem Ansatz von Edmundson (1969), welcher das genauere Untersuche des Zwecks der Zusammenfassung außen vor hielt, dazu übergegangen, den späteren Zweck der Kurzzusammenfassung mit in die Validierung einfließen zu lassen.

Der Auftrag, welcher der Anstoß für die Erstellung einer Kurzzusammenfassung ist, stellt somit eine der Hauptquellen für die Definition der Summarising Factors dar, und wirkt sich vor allem auf die Purpose und Output Factors aus. Aus diesem Grund ist es für die Evaluierung ebenso wichtig, dass nicht nur der Algorithmus an sich, sondern in Zusammenhang mit dem eigentlichen Auftrag, evaluiert wird.

Hierbei muss gesagt werden, dass es grundsätzlich zwei Blickwinkel für die Evaluierung gibt. Diese sind der “innerliche” und der “äußerliche” (intrinsic und extrinsic) Ansatz. Bei dem innerlichen Ansatz wird das System gegen seine eigenen Ziele evaluiert, während bei dem äußerlichen Ansatz das System im Bezug auf die Erfüllung von Anforderungen durch den Auftrag evaluiert wird (Zum Beispiel, ob die erstellten Kurzzusammenfassung qualitativen Richtlinien genügen, die für den Auftrag erforderlich sind, ohne Rücksicht auf den eigentlichen Algorithmus zu nehmen).

Bei diesen Evaluierungen werden vier Grade an “Auftragsnähe” unterschieden. Diese lauten:

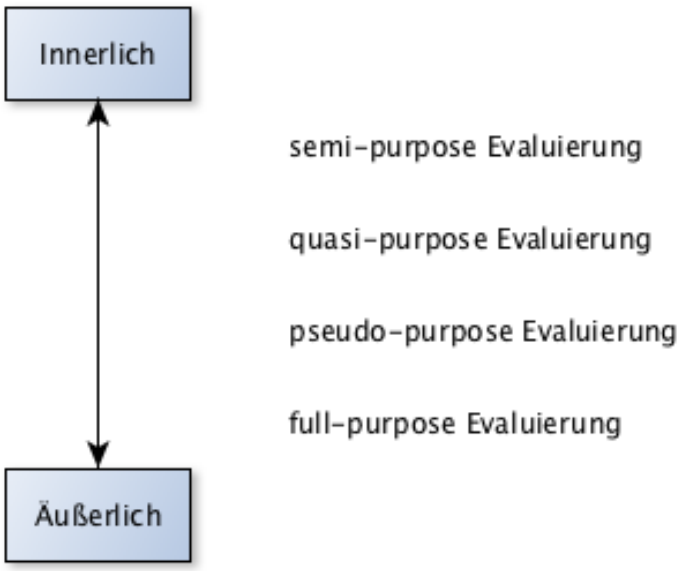

Abbildung 1.2: Die vier Grade der Evaluierung in Bezug auf ihre Auftragsnähe

1. semi-purpose evaluation (z.B. Evaluierung auf korrekte Grammatik. Es ist deshalb Auftragsfern, da die Erzeugung von grammatikalisch korrekten Sätzen eine eher allgemein gültige Anforderung ist, welche auch erfüllt werden sollte, wenn sie im Auftrag nicht explizit erwähnt wurde. Eine Ausnahme stellt dies nur dar, wenn im Auftrag ein Verfahren beschrieben wird, wie dies passieren soll.)
2. quasi-purpose evaluation (z.B. das Vergleichen mit Modellen, um die Strukturen der Ergebnisse zu evaluieren.)
3. pseudo-purpose evaluation (z.B. Simulation von auftragsnahen Testdaten.)
4. full-purpose evaluation (z.B. Erzeugen einer konkreten Kurzzusammenfassung, wie Sie durch den Auftrag gefordert wird.)

Problemstellungen bei der Evaluierung

Ein großes Problem ist, wie bei der Evaluierung das Erkennen der Kernaussage eines Textes und seine Repräsentation in der Kurzzusammenfassung zu erfassen ist. Diese Beziehung zwischen der Kurzzusammenfassung und dem Gesamttext ist selbst bei alltäglichen Texten eine Schwierigkeit, da die Kernaussagen eines Textes auf viele verschiedene Arten formuliert sein kann. Es ist bei solch einem Problem naheliegend, dass für die Evaluierung genau spezifiziert wird, was in der Kurzzusammenfassung enthalten sein soll. Dies führt allerdings zu weiteren Problemen, da selbst Menschen untereinander sich oft nicht darüber einig sind, welchen Details sie den höchsten Stellenwert geben, wie in früheren Untersuchungen [Rath et al, 1961] gezeigt wurde.

Concept Capture

Bei der Lösung dieses Problems hat sich anfangs der sogenannte “source-summary comparison” Ansatz durchgesetzt, bei dem die Quelltexte mit ihren Kurzzusammenfassungen verglichen wurden. Das Problem hierbei war, dass das methodische Vergleichen von Quelle und Ergebnis, häufig nur schwer umsetzen lässt. Um einen Algorithmus A durch einen Evaluierungsalgorithmus B zu evaluieren, sollte zuerst der Evaluierungsalgorithmus selbst evaluiert sein. Da die Ergebnisse sonst keine wirkliche Zuverlässigkeit gewährleisten. Und genau das war das Problem bei diesem Ansatz.

Gold Standards

Mit dem Gold standards ist das menschliche Urteilsvermögen gemeint, welches zu Rate gezogen wird, um über die Qualität der Ergebnisse zu entscheiden. Hierbei wird der Algorithmus anhand der Kurzzusammenfassungen und der Quellen von einem Menschen bewertet. Dies ist zwar ein wesentlich zeitintensiveres Vorgehen, aber zur Not die letzte verlässliche Instanz, um eine Entscheidung treffen zu können. Wie bereits erwähnt neigen Menschen dazu, ein und dieselbe Sache unterschiedlich zu bewertet. Was also der eine Mensch als gut bewertet, könnte der Nächste als mittelmäßig oder vielleicht sogar mangelhaft bewerten. Um diesem Problem entgegenzutreten, wird meistens nicht nur eine Person sondern eine Vielzahl von Personen als “Richter”

beauftragt, welche dann die Ergebnisse untersuchen und gemeinsam eine Bewertung ausdiskutieren.

Baselines und Benchmarks

Bei diesem Ansatz wird versucht, eine Bewertung der Kurzzusammenfassung anhand eines Index durchzuführen. Es gab hier einige Versuche, allerdings alle mit mäßigem Erfolg. Einer war zum Beispiel, dass jeder Satz im Gesamttext über die Worthäufigkeit mit einem Score versehen und danach überprüft wurde, wie viele Scores in der Kurzzusammenfassung enthalten sind. Diese Vorgehensweise ist allerdings nur sinnvoll, wenn die Erzeugung der Kurzzusammenfassung nicht über die Worthäufigkeit stattgefunden hat, da ansonsten der Algorithmus gegen sich selbst getestet wird.

1.2.4 Strategien

In den letzten 20 Jahren hat es viele neue Versuche und unterschiedliche Strategien gegeben um den Anforderungen an TS-Systemen gerecht zu werden. Diese gingen von flachen, hauptsächlich statistischen Ansätzen bis hin zu formalen / modellgetriebenen Ansätzen. Gerade in den letzten 10 Jahren wurde wesentlich intensiver der statistische Ansatz untersucht, als es davor der Fall war. Ziemlich viel Aufmerksamkeit bekam zum Beispiel die Kombination aus statistischen und formalen Techniken, welche früher eher selten versucht wurden, aber in den letzten 10 Jahren zu einer sehr populären Strategie wurde, und sich in einigen Implementierungen wieder findet, wie zum Beispiel Lite-GISTexter [Lacatusu et al., 2003]. Antrieb dieser Entwicklung war teilweise das Aufkommen neuer Evaluierungssysteme, ebenso aber die Verfügbarkeit von wesentlich vielfältigeren Trainings- und Testdaten durch das Großwerden des Internets.

Da es zu viele Projekte gibt, um jedes einzeln zu erläutern, werde ich die verschiedenen Systeme in extractive und non-extractive unterteilen. Mit den extractive Systemen sind jene gemeint, welche die ausgewählten Sätze unverändert in die Kurzzusammenfassung kopieren, und mit den non-extractive Systemen sind jene gemeint, welche versuchen eher eine Reproduktion des Rextes aus der Zusammenfassungsrepräsentation zu erzeugen (also eigene Sätze bilden).

1.2.5 Extractive

Der statistische Ansatz ist wohl der einfachste und weitest verbreitete Ansatz für den Entwurf von TS-Systemen. Der grundlegende Gedanke geht auf Luhn zurück, welcher die einzelnen Sätze mit einem Score versehen hat, welcher auf der Score = TF * IDF Formel [H.P. Luhn, 1957] beruht, und im Grunde die Wichtigkeit verschiedener Worte analysiert, und anhand dessen die Sätze bewertet. Die bewerteten Sätze werden dann ganzheitlich in einem Ranking angeordnet, und in die Kurzzusammenfassung kopiert, bis diese eine entsprechende Länge aufweist. Ein Problem bei diesem Ansatz ist, dass durch das Ranking von Worten, bestimmte Worte einen höheren Score bekommen, wodurch es dazu kommen kann, dass sehr ähnliche Sätze, oder Sätze mit ähnlicher Aussage zu häufig in der Kurzzusammenfassung vorkommen. Um diesem Problem entgegen zu wirken, wird das so genannte Maximal Marginal Relevance [Carbonell, Goldstein, 1998] angewendet, welches die Sätze vergleicht und dann aussortiert, falls sie zu ähnlich sind. Dieses Problem hat noch eine weitere Auswirkung, nämlich, dass nur die aussagekräftigsten Sätze genommen werden, und andere Sätze, welche unter Umständen ebenfalls interessante Details enthalten können, ignoriert werden. Diesem Problem ist man mittels Clustering begegnet (Details folgen später). Dabei wurden dann mehrere Ranking (eine Ranking Liste pro Cluster) erstellt, und von den anderen Clustern jeweils ein oder mehrere Sätze hinzugefügt.

Der erweiterte statistische Ansatz mit Lexical Units und Features

Gerade für die generischen Strategien (also jene, die auch zum Beispiel mit Fragestellungen der Benutzer umgehen können) hat sich gezeigt, dass eine auf Anfragen basierende Strategie mehr als nur die grundlegenden statistischen Methoden benötigt. Hierfür wurde dann ein Anfragen-konzept entwickelt, welches die Anfrage analysiert und diese dann mit Symbolen oder Features ausstattet, sodass diese dann als Grundlage für die statistischen Methoden benutzt werden. Dabei werden häufig Modellierungen wie N-Gramme, Wortpaarbildungen oder Wortrelationen angewendet. Für die eher linguistischen, auf Symbolen basierenden Vorgehensweisen werden häufig Parser verwendet um zum Beispiel Nomengruppen oder dominante Satzstrukturen hervorzuheben und entsprechend mit höheren Scores zu bewerten. Der Prozess zur Analyse

ist hierbei definitiv keine triviale Aufgabe, egal ob eher statistisch oder symbolisch ausgelegt. Das Ziel ist es hierbei ebenfalls ein Scoring für die Sätze durch das Einbinden von Features zu bilden. Dieser Ansatz wurde in den letzten 10 Jahren sehr häufig verfolgt und hat ein nicht ganz eindeutiges Ergebnis hervorgebracht. Während für die einfache Erzeugung von Kurzzusammenfassungen die Resultate teilweise nicht erwähnenswert besser waren und den Mehraufwand kaum rechtfertigen konnten, hat sich bei der DUC (Document Understanding Conference) gezeigt, dass dieser Ansatz eine gute Strategie ist, um das Beantworten von Fragen zu realisieren, da durch die Modelle mehr Bezug auf die Factors genommen werden kann.

Der erweiterte statistische Ansatz mit Strukturen

Bei diesem Ansatz geht es darum, nicht Features oder Symbole zu bestimmen und danach im Text zu suchen, sondern die Struktur des Textes als ganzes zu erfassen und daraufhin wichtige Informationen zu gewinnen. Hierbei werden häufig die Sätze in einer Baumstruktur gegliedert, welche nicht nur für das Scoring dienen sollen, sondern auch bereits eine Art Leitfaden für die Auswahl von Komponenten für die spätere Zusammensetzung der Kurzzusammenfassung dienen soll. Das Hauptkriterium zum Erstellen der Kurzzusammenfassung stellt hierbei nicht mehr eine Liste von Features dar, sondern vielmehr das Einbeziehen von Strukturvergleichen, Merging, Scoring und Ranking. Das Eingliedern in Baumstrukturen bezieht sich immer nur auf ganze Sätze. Möglich wäre es zum Beispiel noch, dass die Sätze selbst zerlegt und in Baumstrukturen aufgegliedert werden, um zum Beispiel Rückschlüsse auf den Redefluss zu bekommen. Es gab auch einige Versuche bei denen man versucht hat die Sätze in eine logische Form zu interpretieren, um daraufhin logische Verknüpfungen zwischen den einzelnen Sätzen herstellen zu können. Es gab auch weitere Versuche, bei denen diese logischen Verbindungen genutzt wurden, um Satzaussagen logisch zu erfassen oder ganze Netzwerke von logischen Formen (als aussagekräftige Stellvertreter für die Wörter) zu erzeugen. Die meiste Arbeit wurde für das sogenannte Rhetorical Structure Theory (RST) aufgebracht. Hierbei werden den Sätzen Marker angeheftet, welche Informationen über ihre Redestruktur darstellen, und diese dann in einer Baumstruktur angeordnet. Der Baumstruktur wurde danach an den Knoten jeweils ein Wichtigkeitsstatus hinzugefügt, welches die Sätze für die Kurzzusammenfassung qualifiziert

hat. Bei Implementierungen hat sich gezeigt, dass eine Vertiefung der symbolischen Strukturen nicht unbedingt zu besseren Ergebnissen führt. Vielmehr geht man davon aus, dass ein mehr anwendungsorientierter Ansatz für diese Strukturen zu besseren Ergebnissen führen könnte.

1.2.6 Non-Extractive

Bei den non-extractive Ansätzen ist es das Ziel, wie bereits weiter oben erwähnt, die Sätze für die Kurzzusammenfassung nicht direkt aus dem Gesamttext zu kopieren, sondern diese generieren zu lassen. Der Unterschied zwischen den extractive und den non-extractive Ansätzen zeigt sich in der Transformationsphase dahingehend, dass zwar die statistischen oder symbolischen Techniken ebenso Anwendung finden, diese allerdings mehr Details über die syntaktische Verwendung der ausgewählten Wörter oder Aussagen in der Zusammenfassungsrepräsentation liefern müssen. Der Arbeitsaufwand in der Generationsphase ist bei den non-extractive Ansätzen dementsprechend wesentlich höher und hat den Bedarf an Modellen zur Erzeugung von wohlklingenden Sätze verdeutlicht. Im Großen und Ganzen muss man sagen, dass die non-extractive Ansätze noch nicht genug ausgereift sind, um die extractive Kurzzusammenfassungen abzulösen, allerdings ist es auf lange Sicht ein, mit Sicherheit lohnenswertes und realistisches Ziel.

1.3 Motivation und Ziel dieser Arbeit

Das Ziel dieser Arbeit ist es, einen neuen Ansatz im Bereich der statistischen TS-Systeme zu validieren. Hierfür wird eine Behauptung aufgestellt (s.u.), welche mittels dieser Arbeit belegt werden soll. Die Behauptung lautet:

"Durch das zusätzliche Auswerten einer Sentimentanalyse zur Häufigkeitsanalyse, kann die Qualität von Kurzzusammenfassungen verbessert werden."

Diese Aussage beinhaltet mehrere Begriffe, auf welche nun kurz eingegangen werden sollen, um Missverständnisse auszuschließen.

- **Sentimentanalyse**
  Das Analysieren von Stimmungen in Texten. Die Stimmung kann zum Beispiel negativ, neutral oder positiv sein. Beispiele für Sätze mit negativen oder positiven Stimmungen lassen sich sehr gut in Kritiken finden.
- **Häufigkeitsanalyse**
  Das statistische Erfassen der Häufigkeiten von Wörter. Es wird hierbei also auf den statistischen Ansatz von TS-Systemen eingegangen, nicht auf formal- oder modellgetriebene Ansätze.
- **Qualität**
  Die Qualität wird an dem Maß gemessen, wie deckungsgleich die erzeugten Kurzzusammenfassungen mit einem, von Menschenhand erstellten, Ranking sind.

Auf all diese Punkte wird im weiteren Verlauf dieser Arbeit detailiert eingegangen. Am Ende dieser Arbeit wird eine Auswertung über den Erfolg oder Misserfolg dieses Ansatzes erbracht.

2 Erläuterung zugrundeliegender Algorithmen

Bevor es los geht, wird es eine kurze Einführung in die Algorithmen geben, die als Werkzeuge für diese Arbeit ausgewählt wurden, und erläutert, in wie weit sie welchen Zweck erfüllen sollen.

2.1 TF-Score

2.1.1 Allgemeines

Die Frage die sich im automatischen Erzeugen von Zusammenfassungen früher oder später immer stellt, lautet: *Wie bewerte ich was wichtig ist, und was nicht?*

Mit dieser Frage hat sich auch Hans Peter Luhn [Luhn,1958] während seiner Zeit bei IBM beschäftigt, und hat hierfür eine wichtige Behauptung aufgestellt. Er sagte, dass die Wichtigkeit eines Terms anhand seiner Häufigkeit im gegebenen Dokument bewertet werden kann.

Terme mit mittlerer Häufigkeit haben hierbei einen höheren Wert, während Terme mit hoher oder niedriger Häufigkeit einen niedrigeren Wert haben. Dieses Phenomen lässt sich dadurch begründen, dass Terme die sehr häufig auftreten, meist Artikel, Präpositionen oder Ähnliches sind, welche eher zur Bildung von Grammatik dienen, als die eigentliche Aussage eines Textes zu transportieren. Terme die hingegen zu selten auftreten, haben nicht genug Relevanz für das behandelte Thema haben.
Gerade in natürlich sprachigen Texten (wie z.B. Zeitungsartikeln) ist es häufig der Fall,

das nur wenige Terme mit hoher Häufigkeit und viele Terme mit geringer Häufigkeit vorkommen. Dies kann zu dazu führen, dass das Ermitteln der “mittleren Häufigkeit” sehr erschwert wird. Deshalb wurde die These noch weiter ergänzt, indem der Interval eingegrenzt wurde. Hierbei handelte es sich um einen oberen und unteren “Cut-Off” welche zum Ziel hatten, alle Terme in den hohen und niedrigen Häufigkeitsbereichen zu entfernen. Dadurch konnte der Fokus der Bewertung mehr auf den wesentlichen Teil (mittlere Häufigkeit) gelenkt werden.

Dieses Relation wird in der folgenden Abbildung 2.1 graphisch dargestellt.

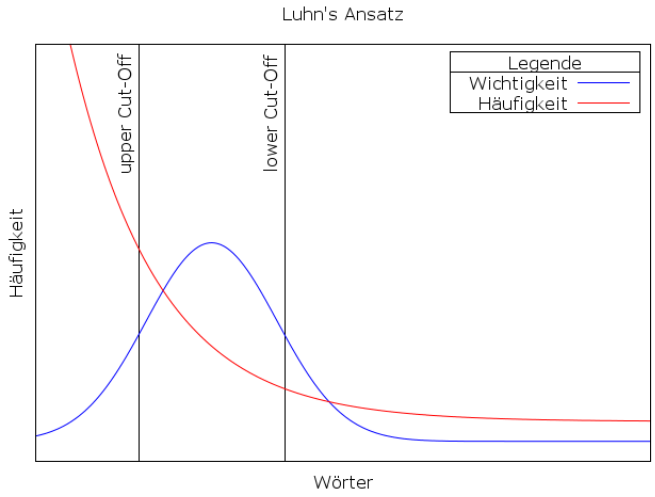

Abbildung 2.1: Der Zusammenhang zwischen der Häufigkeit und Wichtigkeit von Termen. Der Häufigkeitsgraph stellt die nach Ihrer Häufigkeit sortierten Wörter dar.

Dieser Ansatz wurde unter dem Begriff “Luhn‘s claim on information retrieval” [Luhn,1958] bekannt. Dieser Ansatz wurde aufgrund verschiedener Eigenschaften in der Vergangenheit häufig aufgegriffen und erweitert. Diese Eigenschaften waren zum Beispiel, dass er sprachunabhängig war, und nur wenig Vorverarbeitung benötigte.

2.1.2 Der Algorithmus

Die wohl grundlegendste Realisierung dieses Gedanken wird in der

$$score = tf \times idf \tag{2.1}$$

(term frequeny-inverse document frequency) Gleichung umgesetzt.

tf steht hierbei für die *term frequency* (Termhäufigkeit) und *idf* für die *inverse document frequency* (inverse Dokumentenhäufigkeit).*tf* ist die Häufigkeit des Terms innerhalb des untersuchten Dokuments. Die *idf* beschreibt das Verhältnis der Anzahl an Dokumenten in denen der Term vorkommt zu der Anzahl aller Dokumente im Dokumentencluster. Beide Größen lassen sich folgendermaßen berechnen:
$t =$ der momentane Term
$w =$ ein beliebiger Term
$d =$ das aktuelle Dokument
$D =$ die Menge aller Dokumente im Dokumentencluster
$f(t, d) =$ die effektive Häufigkeit eines Terms t im Dokument d (Abzählen)

$$tf(t, d) = \frac{f(t, d)}{max\{f(w, d) : w \in d\}} \tag{2.2}$$

$$idf(t, D) = log \frac{N}{|\{d \in D : t \in d\}|} \tag{2.3}$$

Die Gleichung 2.2 zeigt hier die Normalisierung der Termfrequency. Das ist zwar nicht nötig, wird aber sehr häufig gemacht (ebenso in dieser Ausarbeitung). Dies war eine sehr elementare Implementierung des IR Ansatzes, und es gibt noch viele weitere, die gerade die Behandlung der upper und lower Cut-Offs behandeln. Für die gegebenen Zwecke sind die oben aufgeführten Vorgehensweisen ausreichend, wie später noch detailierter erläutert wird. Die durch den IR Ansatz gewonnenen Scores werden die erste Grundlage zur Ermittlung der Relevanz von Termen und Sätzen im weiteren Verlauf darstellen.

2.2 Sobel-Filter

2.2.1 Allgemeines

In diesem Abschnitt wird etwas Distanz zur Textverarbeitung genommen und der Fokus auf ein anderes Gebiet der Künstlichen Intelligenz gelegt, die Bildverarbeitung. In den folgenden Absätzen wird ein Verfahren erläutert, dass zur Erkennung von Kanten dient. Dieses Verfahren erlaubt das Bewerten der Stärke und der Richtung von Kanten. Die Rede ist von dem Sobel-Filter. Das tiefere Verständnis des Sobel-Filter ist zwar nicht unbedingt erforderlich um die spätere Implementierung nachvollziehen zu können, die Idee darhinter wird allerdings an einer bestimmten Stelle im System verwendet werden.

2.2.2 Das Problem und der Lösungsansatz

Das Problem mit Kanten in Bildern ist, dass viele Verfahren lediglich die Kante erkennen können, allerdings keine weiteren Details dazu liefern. Häufig jedoch ist es auch interessant, ob die Kante zum Beispiel steigt oder fällt. Dieses Problem lässt sich mit dem Sobel-Filter lösen.

2.2.3 Der Algorithmus

Der Sobel-Filter verwendet zwei Masken. Die erste Maske **??** dient zur Erkennung von vertikal Kanten. Während die zweite Maske **??** dazu dient, horizontale Kanten zu erkennen. Wenn die Kanten in einem Bild untersucht werden, wird das gesamte Bild, Bildpunkt für Bildpunkt durchlaufen, und auf jeden Bildpunkt werden diese Masken angewendet. Die Werte in den Masken stellen Faktoren dar, die mit den Punkten an den jeweiligen Stellen multipliziert werden. Danach wird die Summe aller Produkte gebildet und die resultierenden Werte (für G_x und G_y) erzeugen mittels der unten angegebenen Formeln (2.4,2.5) eine Vektor. Die Länge des Vektors stellt die Stärke der Steigung dar, während der Winkel die Richtung in welche die Kante steigt angibt. Wenn in einem Bereich des Bildes keine Kanten sind, nähert sich die Länge des Vektors an diesen Bildpunkten der Null an.

-1	0	1
-2	0	2
-1	0	1

Abbildung 2.2: G_x

-1	-2	-1
0	0	0
1	2	1

Abbildung 2.3: G_y

$$G = \sqrt{G_x^2 + G_y^2} \tag{2.4}$$

$$\alpha = \arctan\left(\frac{G_y}{G_x}\right) \tag{2.5}$$

G liefert eine Aussage über die Stärke der Kante, und α sagt uns, in welche Richtung die Kante steigt.

2.2.4 Abschliessende Worte

Wie bereits oben erwähnt, geht es hier lediglich um die Idee, dass aus zwei Eingangsdimensionen ein repräsentativer Vektor entsteht, welcher gewisse Informationen liefert. Wie die genaue Umsetzung dieser Idee im System aussieht, wird im weiteren Verlauf erläutert.

2.3 K-Means

2.3.1 Allgemeines

Der K-Means ist ein Clustering-Algorithmus der eine Menge von (in diesem Fall) Punkten, einer festen Anzahl (K) von Clustern zuweist. Die Wahl ist auf diesen Algorithmus gefallen, da er an sich leistungsstark ist, und die bekannten Probleme für die Zwecke dieser Arbeit akzeptabel sind.

2.3.2 Der Algorithmus

Der Algorithmus beginnt damit, dass für jeden Cluster ein Startpunkt gewählt wird. Diese Startpunkte können beliebig aus der Menge aller Punkte gewählt werden. Im weiteren Verlauf werden sich die Startpunkte verändern, daher wird ab jetzt der Begriff Clusterzentren anstelle von Startpunkten gewählt. Nach dem die Clusterzentren gewählt wurden, werden Linien zwischen all diesen Clusterzentren gezogen, und die Linien in der Mitte geteilt. Diese Teilung stellt nun die Grenze eines Clusters dar. Alle Punkte innerhalb der $Clustergrenzen_A$ werden dem $Cluster_A$ zugewiesen. Nach dem ersten Durchlauf kann dies wie in Grafik 2.4 aussehen.

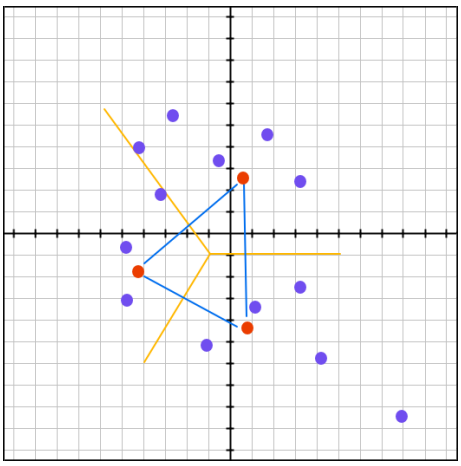

Abbildung 2.4: Die roten Punkte repräsentieren die Clusterzentren. Die blauen Linien sind die Verbindungen zwischen den Clusterzentren, und die gelben Linien stellen die Clustergrenzen dar.

Nachdem alle Punkte den jeweiligen Clustern zugeordnet wurden, werden die Clusterzentren neu berechnet. Danach werden wieder die Grenzen gezogen, usw... . Falls durch das Neuberechnen der Clustergrenzen ein Punkt nun in einem anderen Cluster liegt, muss dieser Punkt dem neuen Cluster zugeordnet werden. Diese Arbeitsschritte werden solange wiederholt, bis sich die Clusterzentren zum vorherigen Durchlauf nichtmehr geändert haben. Für die Berechnung der neuen Clusterzentren gibt es verschiedene Ansätze dies zu tun, in dieser Arbeit gehen wir von einem lokalen Zentrum aus.

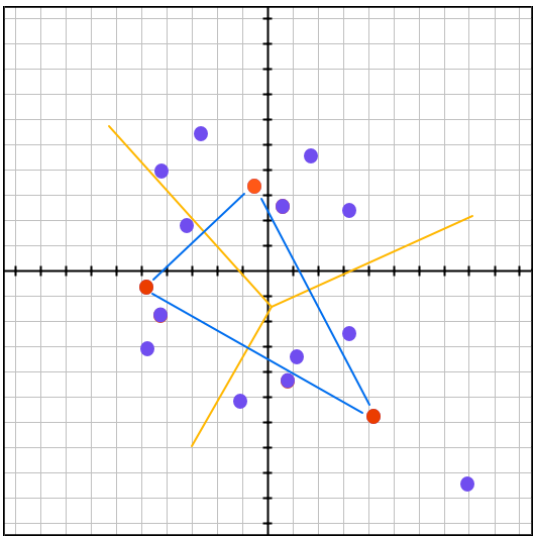

Abbildung 2.5: Die roten Punkte repräsentieren die Clusterzentren. Die blauen Linien sind die Verbindungen zwischen den Clusterzentren, und die gelben Linien stellen die Clustergrenzen dar.

In Pseudocode kann das folgendermaßen beschrieben werden:

```
...
List<Map<Point>> clusters = ...;
Map<Point> oldCenters = ...;
Map<Point> centers = ...;

while !centers.equals(oldCenters):
    computeClusterBorders();
    clusters = assignPointsToClusters();

    oldCenters = centers;
    center = computeNewCenters();

return clusters;
...
```

2.3.3 Abschliessende Worte

Das Clustering wird später Verwendung in der Satzauswahl finden. Bekannte Nachteile des K-Means sind, dass es zum Beispiel in der Natur des Algorithmus liegt, relative gleich große Cluster zu erzeugen (da die Grenzen immer bei der Hälfte der Distanz liegen), was allerdings weniger ein Problem für diesen Anwendungsfall darstellt, da die Grenzen zwischen den Clustern relativ weit distanziert sind (Begründet durch die grobe Auflösung der Sentimentanalyse, aber dazu später mehr). Desweiteren muss das K im Vorfeld bestimmt werden, was allerdings für das spätere System durch die Validierung ermittelt wird. Vorteile hingegen sind, das der K-Means Algorithmus relativ performant ist im Vergleich zu vielen anderen Clusteringalgorithmen. Da die Probleme des Algorithmus von diesem System gut handhabbar sind, ist die Wahl für den Clusteringalgorithmus auf den K-Means gefallen.

3 Das System

3.1 Allgemeines

Für die Entwicklung des Systems stehen in dem Bereich Summarizing Factors ausschließlich die Input Factors im Vordergrund. Da es keine besonderen Anforderungen an das Format der Kurzzusammenfassung oder die gewonnenen Informationen gibt, fallen die Output- sowie Purpose Factors weg. Zu den Input Factors zählen für dieses System folgende Vorgaben:

- Der Input muss in der englischen Sprache verfasst sein.
- Es werden ausschließlich Einzeltexte zusammengefasst, keine Dokumentencluster.
- Alle erforderlichen Parameter für die Verarbeitung müssen zu Beginn der Verarbeitung feststehen.

Das Gesamtsystem lässt sich in die drei Kernaufgaben unterteilen, wie es in der Einleitung bereits vorgestellt wurde:

- **Interpretation**
  Die Interpretationsphase wird in diesem System hauptsächlich durch das zugrundeliegende Stanford CoreNLP Framework dominiert. Es nimmt den ursprünglichen Text entgegen, und vollzieht die Vorverarbeitung. Es ermöglicht dem System, selbst komplizierte Sätze korrekt zu erkennen und zu splitten. Desweiteren nimmt es noch das POS-Tagging vor sowie die Sentimentanalyse. Die Funktionalität des CoreNLP wurde in einen Wrapper gekapselt, um die Systeminternen Komponenten nicht zu sehr von der Schnittstelle des CoreNLP

abhängig zu machen. Daher werden für die Repräsentation der Sätze und Terme eigene Klassen verwendet, und nicht die des CoreNLP. Für die Umwandlung der CoreNLPKlassen zu den eigenen Klassen ist ebenfalls der CoreNlpWrapper verantwortlich.

- **Transformation**
  Die Transformationsphase ist das Kernstück des Systems, und hat in erster Linie folgende Aufgabe die nacheinander abgearbeitet werden müssen:

  1. Berechnen der TF-Scores und der Sentiments.
  2. Durchführung des Clusterings.
  3. Erzeugung der Kurzzusammenfassungsrepräsentation mittels Satzauswahl.

  Um diese Aufgaben zu erfüllen, wird jede einzelne Teilaufgabe gemäß dem Single Responsibility Prinzip von einem eigenen kleinen Modulen verarbeitet. Diese Module sind zu elementar, um sie an dieser Stelle nur kurz zu überfliegen, deswegen werden sie in den folgenden Punkten dieses Kapitel detailiert erläutert.

- **Generation**
  Die Generationsphase fällt recht einfach aus, da keine besonderen Anforderungen an die Darstellung der entnommenen Informationen gestellt wurden. Die Kurzzusammenfassungrepräsentation liefert bereits die entnommenen Information inklusive der "Rohdarstellung" und muss daher nichtmehr umfangreich bearbeitet werden. Die Rohdarstellungen der einzelnen ausgewählten Sätze wird gemäß ihrer ursprünglichen Anordnung zu einem Text zusammengehängt, welcher den finalen "Summarytext" darstellt.

Die Aufteilung wird in der folgenden Darstellung 3.1 veranschaulicht.

3.2 Systemübersicht

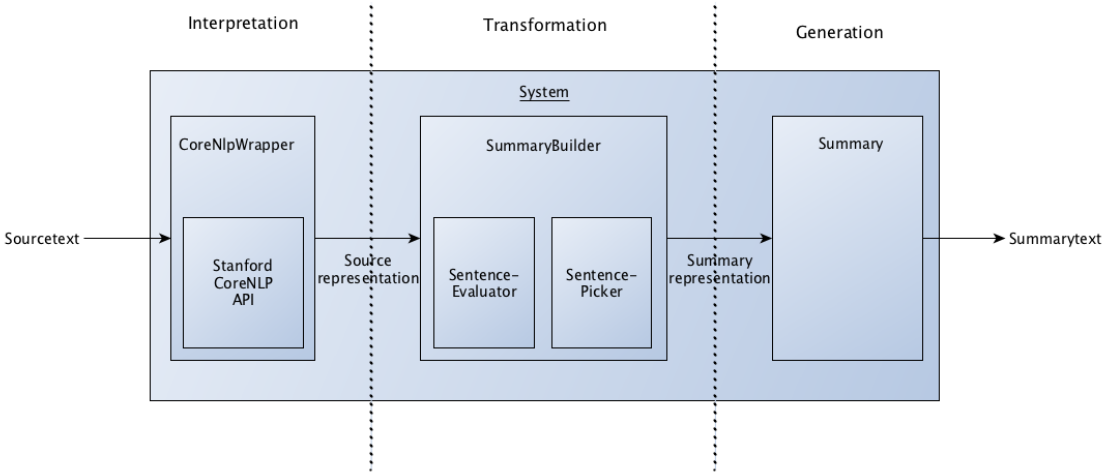

Abbildung 3.1: Diese Abbildung zeigt die Systemunterteilung mit ihren Hauptkomponenten gemäß dem, in der Einleitung vorgestellten Entwurfskonzept.

Für die Schnittstellen zwischen diesen Bereichen dienen gewisse Container- bzw. Datenmodellklassen um die Informationen, abhängig von dem momentanen Verarbeitungsschritt, zu übermitteln. Zu diesen Klassen zählen folgende:

- **Term** Die Termklasse repräsentiert alles was das CoreNLP als ein eigenes Wort versteht. Dazu gehören für gewöhnlich nicht nur normale Wörter sondern auch Satzzeichen, Zahlen, Einheitssymbole und so weiter. Diese Klasse beinhaltet ebenso die Information über das POS-Tagging und das Stemming.

- **Sentence**

  Die Sentenceklasse stellt ebenso wie die Termklasse eine eigene Datenklasse für die Resultate des CoreNLP dar. Ihre Aufgabe ist es, die in diesem Satz befindlichen Terme aufzubewahren, und jegliche Informationen über Sobel,

ursprüngliche Position im Text sowie weitere Zusatzinformationen welche für die weitere Verarbeitung während des Clustering notwendig sind, zu speichern.

- **Sobel**

  Die Sobelklasse repräsentiert den Container für die beiden Eingangsdimensionen, welche für die Bewertung von Sätzen herangezogen werden. Es gibt folglich eine Objekt dieser Klasse pro Sentence in einem Verarbeitungsdurchlauf.

- **Point** Die Pointklasse wird ausschliesslich für das Clustering verwendet, und stellt eine Verbindung zwischen einem Sentence und der Position im Koordinatensystem dar.

- **Summary** Die Summaryklasse stellt eine Art reduzierten geordneten Container dar, der sicherstellt, dass jeder Satz, welche der Kurzzusammenfassung zugeordnet wird, in der richtigen ursprünglichen Position eingegliedert wird. Diese Klasse übernimmt ebenfalls die Rolle der Generationphase, was bedeutet, dass sie für die Umwandlung aus der "Summaryrepresentation" zum "Summary Text" verantwortlich ist.

Die eben vorgestellten Klassen, werden während des gesamten Arbeitsablaufs von den algorithmischen Klassen zur Übergabe oder zum Caching von Informationen verwendet. Ihr Anteil an Logik reduziert sich auf das Wesentliche. Die Relation der drei Kernklassen wir in dem folgenden Diagramm 3.2 graphisch dargestellt.

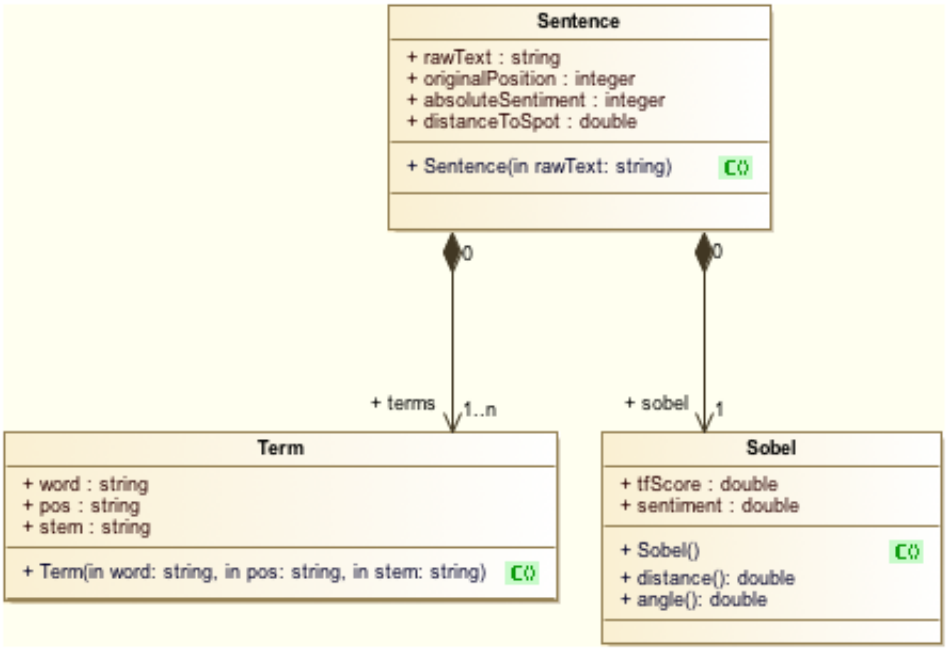

Abbildung 3.2: Die Beziehungen zwischen den drei Klassen Sentence, Term und Sobel.

3.2.1 Satzbewertung

Nachdem der Sourcetext von der Interpretationphase vorverarbeitet wurde, und nun in einem sauberen, leichter zu verarbeitenden Format vorliegt, müssen die Sätze bewertet werden. Hierbei ist es das Ziel für jeden Satz eine Kenngröße zu erzeugen, welche dessen Wichtigkeit in Relation zu allen anderen Sätzen im Text wiederspiegelt. Dies ist einer der kritischsten Bereiche eines TS-Systems, und wird im Rest der Arbeit sehr detailiert beschrieben. Das folgende Diagramm 3.3 veranschaulicht kurz den Ablauf bei der Bewertung der Sätze. Hier eine verbale Ausführung diese Ablaufs:

1. Der SentenceEvaluator bekommt eine Liste von vorverarbeiteten Sätzen.
2. Dieser weist den SobelOperator an, die Bewertung der Sätze durchzuführen.
3. Der SobelOperator ermittelt nun für jeden Satz ein Objekt der Sobelklasse welches die Wichtigkeit dieses Satzes repräsentiert. Dafür bedient es sich bei

der TFScoreklasse, sowie den im CoreNLP entstandenen Sentiment. Auf diesen Vorgang wird später noch detailiert eingegangen.

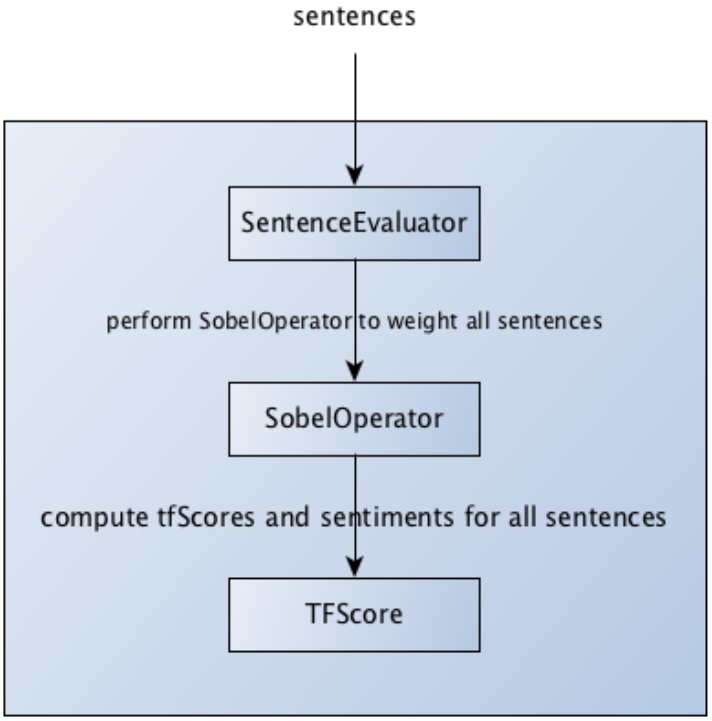

Abbildung 3.3: Der Ablauf, wenn eine Liste von Sentences durch das SentenceEvaluator bewertet werden soll.

Der zweite Teil des SummaryBuilders, der SentencePicker, wird, ebenso wie die einzelnen Schritte im oben stehenden Diagramm 3.3, in einem eigenständigen Kapitel beschrieben, und daher an dieser Stelle nicht weiter erläutert. Es sei lediglich gesagt, dass es die Aufgabe des SentencePicker ist, die Sätze anhand ihrer zuvor ermittelten Kenngrößen für die Kurzzusammenfassungsrepäsentation auszuwählen. Diese kurze Übersicht über das System sollte dazu dienen, die grobe Struktur der Komponenten zu vermitteln, und den Ablauf des Systems, vom Sourcetext bis hin zur Kurzzusammenfassung zu veranschaulichen. Im weiteren Verlauf dieses Kapitels, werden die wichtigen Komponenten des Systems näher erläuert.

Im folgenden wird der Ablauf im System beschrieben um einen Überblick darüber zu vermitteln, wie die Komponenten für die Erzeugung der Kurzzusammenfassung eingesetzt werden:

1. Der Text wird aus der Inputdatei eingelesen.
2. Der eingelesene Text wird mittels des CoreNLP Framework vorverarbeitet. (POS-Tagging, Sentencesplitting, Stemming und Sentimentanalyse)
3. Der SummaryBuilder beauftragt den SentenceEvaluator damit die Sätze zu bewerten.
4. Der SentenceEvaluator ruft die Sobelkomponente auf, um für jeden Satz den Vektor zu berechnen.
5. Die Sobelkomponente entnimmt der Sentimentanalyse und dem TFScore die Werte für die jeweiligen Sätze, und normalisiert diese um sie auf ein einheitliches Format zu bringen.
6. Nach diesem Schritt ist die erste Verarbeitungsaufgabe erledigt. Nun beauftragt der SummaryBuilder den SentencePicker die bewerteten Sätze auszuwählen.
7. Hierfür nutzt der SentencePicker den Clusteringalgorithmus um die Sätze ihrer Eigenschaften nach zu ordnen.
8. Danach wird der SummaryBuilder dem SentencePicker solange Sätze entnehmen, bis die Kurzzusammenfassung die gewünschte Länge erreicht hat.
9. Aus dieser Auswahl von Sätzen wird nun die Kurzzusammenfassung von der Summaryklasse generiert.
10. Im Anschluss wird die generierte Kurzzusammenfassung als Output ausgegeben.

Die wichtigen Systemkomponenten werden nun in den folgenden Kapiteln ausführlich beschrieben.

3.3 TF-Score

3.3.1 Allgemeines

Die Implementierung dieses Moduls nimmt in Gesamtsystem die Rolle als Teil der Transformationskomponente ein. Die Aufgabe ist das Liefern der Daten für die erste Dimension des späteren Sobels. Sie weicht etwas von der klassischen Implementierung mit der Ermittlung des Medians ab, und behandelt die Häufigkeiten in Rohform. Grund hierfür ist, dass durch das zugrundeliegende Stanford CoreNLP Framework einige Zusatzinformationen zur Verfügung stehen, die sinnvoll genutzt werden können. Die Notwendigkeit von Upper- und Lower-Cutoffs entfällt ebenso wie die Bestimmung der optimalen mittleren Häufigkeit. Die wichtigste Zusatzinformation, neben dem Stemming, ist in diesem Fall das POS-Tagging (PartOfSpeech-Tagging).

In dieser Arbeit wird davon ausgegangen, dass bestimmte Wortarten wichtiger sind als andere. Also dass zum Beispiel das Nomen “Kernschmelze” mehr Information über den Inhalt eines Textes liefert als das Pronomen “dieses”. Auf diesem Ansatz beruhend werden Faktoren für die Gewichtung dieser Wortarten aus einer XML Datei geladen. Diese Faktoren sind nicht zufälliger Natur, sondern durch ein Trainingstool entstanden (siehe 3.5). Diese Gewichtung der Wortarten hat das Ziel die wichtigen von den unwichtigen Termen abzugrenzen. Es ersetzt also die Upper- und Lower-Cutoffs.

Um nun den TF-Score für einen Term zubestimmen, werden zwei Größen herangezogen:

1. Die Erste ist die Häufigkeit, wie es aus Luhn’s Ansatz her bekannt ist. Die Häufigkeit wird nicht anhand des Wortes selbst, sondern anhand des Stemmings ermittelt. Das bedeutet, dass beispielsweise die Worte “renne” und “rennt” auf dem selben Stemming “renn” abgebildet werden und somit für die Häufigkeitsermittlung als gleich betrachtet werden.

2. Die Zweite stellen trainierbare Faktoren für die POS-Tags dar. Diese Faktoren werden aus einer XML-Datei geladen, und sind bereits zum Beginn der Verarbei-

tung bekannt. Wenn diese XML-Datei nicht existiert, muss Sie im Vorfeld durch das Trainingstool erzeugt werden.

3.3.2 Der Algorithmus

Die Berechnung der Scores wird zu Beginn der Verarbeitung, während der Initialisierung der Klasse durchgeführt, und nicht erst wenn der konkrete Term abgefragt wird. Grund hierfür ist, dass nur während der Initialisierung diese Klasse den Zugriff auf den Gesamttext hat, und die Häufigkeit der Terme, bezogen auf den Gesamttext, ermitteln kann. Wenn später die Scores für die Sätze ermittelt werden, werden die Scores für die Terme aus einer Map abgefragt. Die unten abgebildete Grafik (3.4) soll dabei helfen, die Zusammenhänge zwischen den einzelnen Schritten zur Erzeugung des TF-Scores nachzuvollziehen.

Wie unten im Pseudocode zu sehen ist, wird mit der absoluten Häufigkeit gerechnet, und diese direkt mit dem trainierten Faktor multipliziert. Dies hat den Hintergrund, das die Berechnung der relativen Häufigkeit und die Normalisierung der TF-Scores zu den Aufgaben des Sobels gehört, welcher erst danach stattfindet.

```
1  ...
2  int[] frequencies = ... ;
3
4  for term in text:
5     frequencies[term]++;
6
7  double[] scores = ... ;
8  for term in text:
9     double posFactor = getPosFactor(term);
10    score[term] = frequencies[term] * posFactor;
11 ...
```

Term	Absolute Häufigkeit	POS-Faktoren	TF-Score
.	14	0.003	0.042
,	12	0.01	0.12
Putin	3	0.94	2.82

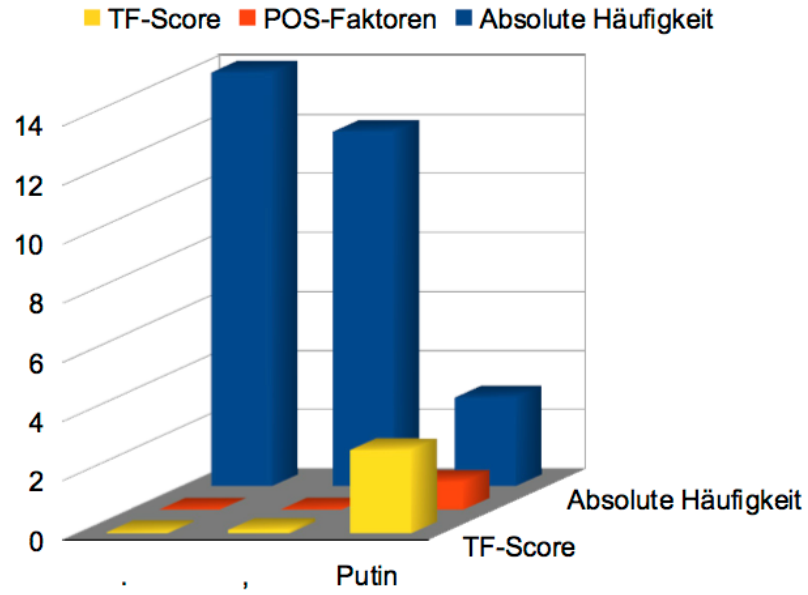

Abbildung 3.4: Diese Abbildung verdeutlicht, wie die POS-Faktoren genutzt werden, um die Scores der "wirklich" wichtigen Terme hervorzuheben. Blau stellt die absolute Häufigkeit eines Terms dar, rot ist der trainierte POS-Faktor zu der Wortart des Terms und gelb ist der resultierende TF-Score für diesen Term.

3.3.3 POS-Faktoren

Zur Bestimmung der Faktoren wird ein Tool zur Verfügung gestellt, dass ermöglicht in Texten, aus einem ausgewählten Trainingsset, die wichtigen Wörter zu markieren. Wichtig ist dabei, dass der Ausführende sich keinerlei Gedanken über die Wortarten macht, sondern nur die Wörter markiert, die in den Augen des Betrachters wichtig sind. Dabei handelt es sich um eine sehr subjektive Erhebung von Werten weshalb es sich lohnt, dies von mehreren Personen ausführen zu lassen, und dann die Ergebnisse abzugleichen.

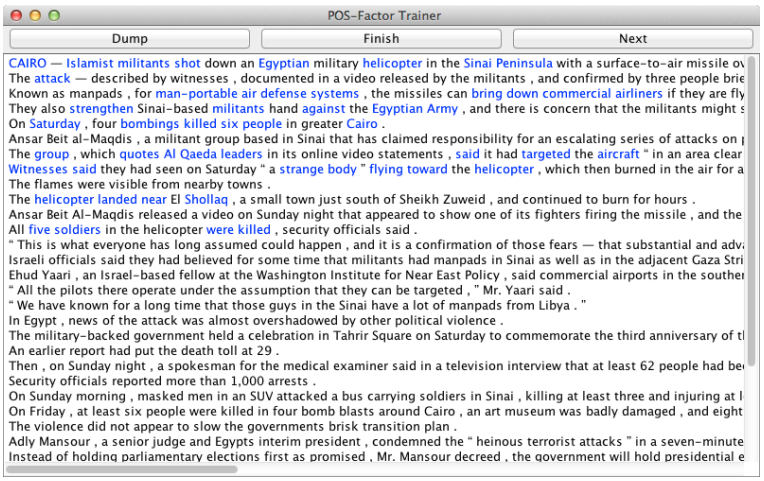

Abbildung 3.5: Die POS-Tags der blau marktierten Wörter werden als wichtig bewertet.

Mit jeder Markierung wird der Zähler dieser Wortart erhöht. Bevor die Werte in eine XML-Datei weggeschrieben werden, müssen alle Werte normalisiert werden, bis sie 0 <= factor <= 1.0 betragen. Für die Normalisierung wird zuerst der größte Wert gesucht, danach wird jeder Zähler durch diesen Wert geteilt, damit der größte Wert dem Faktor 1.0 entspricht und alle anderen Faktor darunter liegen. Nach der Ausführung werden beispielsweise Faktoren wie diese vorliegen:

```
1 ...
2         <tag key="NNP" info="...">0.5087719298245614</tag>
3         <tag key="JJS" info="...">0.017543859649122806</tag>
4         <tag key="NNS" info="...">0.9473684210526315</tag>
5         <tag key="JJR" info="...">0.03508771929824561</tag>
6 ...
```

Eine Auflistung dieser möglichen POS-Tags kann im Anhang eingesehen werden (8.1 [Leeds,2014]).

3.4 Sentiment

3.4.1 Allgemeines

Die hier verwendete Sentimentanalyse wird von dem Stanford CoreNLP Framework vollzogen und bildet eine der wichtigsten Komponenten in diesem System. Die Aufgabe besteht darin, die Satzstimmung, also ob der Satz positiv oder negativ ist, als einen Zahlenwert darzustellen. Diese Zahlwerte sind integer und liegen zwischen -2 und +2. -2 wird auf *sehr negativ*, -1 auf *negativ*, 0 auf *neutral*, +1 auf *positiv* und +2 auf *sehr positiv* abgebildet. Dies sind die 5 Klassen auf die alle Sätze abgebildet werden. Es hat die State of the Art im Bereich Einzelsatzklassifizierung für positiv/negativ von 80% auf 85,4% erhöht.

3.5 Der Algorithmus

Der Algorithmus ist an sich nicht kompliziert, aber sehr stark von dem Trainingsset abhängig. Es wird zur Laufzeit der Satz in einen Baum umgeformt, bei dem die Reihenfolge der Wörter bei horizontalem Durchlaufen erhalten bleibt. Dieser Baum wird rekursiv durchlaufen und die einzelnen Terme mit den benachbarten Termen in einer Funktion aufgelöst. Nach der Auflösung stellt der Rückgabewert eine “Phrase” dar, welche mit einem eigenen Sentimentwert behaftet ist. Diese Phrase wird dann wiederholt mit anderen benachbarten Termen oder Phrases der Funktion übergeben, um daraus eine neue Phrase zu erzeugen, welche durch die Funktion einen neuen Sentimentwert erhält. Dies wird solange durchlaufen, bis aus allen Phrases eine letzte Phrase erzeugt wurde, welche dann den gesamten Satz beinhaltet. der Sentimentwert dieser letzten Phrase stellt dann den Sentimentwert des Satzes dar.

Die Zuverlässigkeit dieses Verfahrens ist also sehr stark von der Auflösungsfunktion abhängig. Ist diese Funktion nicht ausreichend trainiert, leidet die Qualität des gesamten Verfahrens darunter. Zum Training wurde daher ein Set von 11.855 Einzelsätzen aus dem Bereich Filmrezensionen verwendet, und alle Wörter von drei Personen

per Hand markiert (positiv, negative, ... usw.) [ST,2013]. Dieser Aufwand hat sich hinsichtlich der guten Ergebnisse offenbar gelohnt.

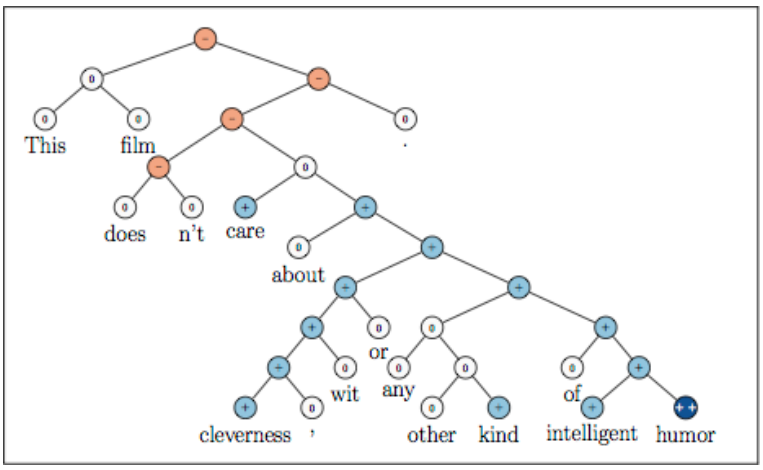

Abbildung 3.6: Eine Darstellung eines Sentiment Trees. Quelle: [ST,2013]

3.5.1 Einbettung

Im System wird die Sentimentanalyse direkt im Anschluss an das SentenceSplitting und POS-Tagging durchgeführt. Dies bedeutet, dass die Bestimmung des Sentiments bereits in der Interpretationsphase durchgeführt wird, und nicht auf die Sätze während der Transformationsphase.

3.6 Sobel

3.6.1 Allgemeines

Die Implementierung des Sobel weicht etwas von dem Sobel in der Bildverarbeitung ab. Grund hierfür ist, dass die Masken des Sobels in der Bildverarbeitung auf die Problemfelder von Bilder ausgelegt sind. Die Problemfelder anderen Bereichen können daher nicht mit unveränderten Masken behandelt werden. Die Aufgabe der beiden Masken wird in diesem System durch die Komponenten des TF-Scores und der Sentimentanalyse erfüllt. Die beiden Eingangsdimensionen stellten daher den TF-Score und das Sentiment dar. Die Berechnungen für die Vektorlänge ist gemäß des Satzes von Pythagoras 2.4 realisiert worden und die Berechung des Winkels über den Arkustangens 2.5 (wie bereits weiter oben beschrieben).

Für die Berechnung der TF-Scores der Sätze stehen zwei Möglichkeiten zur Auswahl:

1. **Summierung**
   Hierbei werden die TF-Scores der einzelnen Terme im Satz aufsummiert und bilden dardurch den TF-Score des Satzes.

2. **Durchschnitt**
   Der Durchschnitt aller TF-Scores der Terme in einem Satz bilden in diesem Ansatz den TF-Score des Satzes.

Beide Varianten stehen dem System zur Auswahl, und sind über den **“useAverage”**-Parameter auswählbar. Jede dieser Varianten hat seine eigenen Vor- und Nachteile. Beispielsweise:

- Durch das Aufsummieren, können Sätze mit eher weniger wichtigen Termen durch Ihre Länge höhere TF-Scores erhalten als kurze Sätze mit wichtigeren Termen.

- Der Durchschnitt klingt zwar fair für kurze Sätze, kann aber zu Problemen führen, wenn zum Beispiel in den Texten Dialoge mit sehr kurzen Sätzen vorkommen. Beispielsweise wenn ein Satz aus einer Unterhaltung nur aus dem

Ansprechen eines der Protagonisten besteht, und dieser nur beim Namen genannt wird. Dann würde der Satz nur aus einem Wort bestehen, mit einem sehr hohen POS-Faktor. Dieser Satz würde einen sehr hohen TF-Score erhalten obwohl er keinerlei Wichtigkeit aufweist.

Welche dieser beiden Varianten sich bewähren wird, ist Aufgabe der Validierung.

3.6.2 Abschliessende Worte

Der Sobel ist klar der Transformationsphase zuzuordnen. Es findet keinerlei Vorberechnung wie bei dem TF-Score oder der Sentimentanalyse statt. Der Sobel hat ebenfalls die Aufgabe, die Eingangswerte beider Dimensionen auf den gleichen Wertebereich zu “normalisieren”. Diese Klasse nimmt zwar eine sehr wichtige, allerdings auch sehr passive Rolle im Gesamtsystem ein. Sie nimmt die Werte der Eingangsdimensionen entgegen, normalisiert diese und fügt die Ergebnisse den Objekten der Datenmodellklassen hinzu.

3.7 Satzauswahl

Wenn nun für alle Sätze der TF-Score und das Sentiment berechnet wurde, und der durch den Sobel entstandenen Vektoren Anhaltspunkte für die Wichtigkeit der Sätze liefern, muss man sich Gedanken darüber machen, wie man diese Vektoren auswertet. Die zugrundeliegende Situation wird in Abbildung 3.7 nochmals graphisch dargestellt.

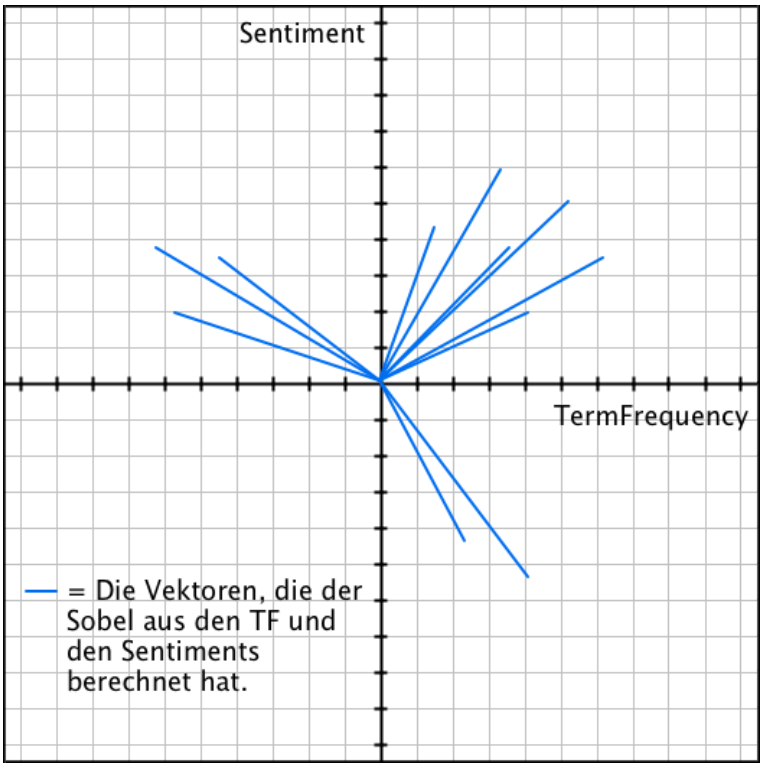

Abbildung 3.7: Die Vektoren welche die Repräsentationen der Sätze darstellen.

Die erste Idee, die einem vielleicht in den Sinn kommen mag ist, dass man die Vektorlänge als Anhaltspunkt wählt, da sie entweder auf eine hohe TF und/oder auf ein starkes Sentiment schliessen lässt. Wenn man solch einem Gedanken nachgeht, werden wichtige Zusatzinformation unberührt bleiben. Zum Beispiel liefern uns die Vektoren nicht nur die Länge sondern ebenfalls die Richtung. Nun könnte man den Gedanken verfolgen, dass man nach Richtungen sucht, in die möglichst viele Sätze zeigen, und dann bevorzugt Sätze aus diesen Richtungen wählt. Dieser Gedanke wurde hier aufgegriffen und umgesetzt. Die Realisierung ist mittels von Clustering erfolgt,

welches versucht Ballungszentren von Vektoren zu erkennen, und dann die Sätze aus diesen Zentren bevorzugt zu verwendet.

In der Implementierung wurde dies mittels des SentencePicker umgesetzt. Der SentencePicker hat die Aufgabe die wichtigen Sätze aus den Clustern zu suchen, und einen aussagekräftigen Score der Sätze zu bestimmen. Wenn es mehrere Cluster gibt, soll ein Score für jeden Satz in Bezug auf den Cluster erstellt werden, in dem sich der Satz befindet. Die Cluster werden dann untereinander nach ihrer Größe und Lage verglichen und in eine Reihenfolge gebracht.

Dies wird in zwei Schritten erledigt:

1. Der SentencePicker sucht alle Cluster, deren Zentren einen X-Wert von größer 0 haben (Also der Winkel ist $-90° < \alpha < +90°$). Dies bedeutet, dass die Sätze in solchen Clustern relativ hohe TF-Scores haben und somit laut Luhn [Luhn,1958] als wichtiger einzustufen sind.

2. Wenn es mehr als einen Cluster in dieser Auswahl gibt, werden sie der Größe nach sortiert, wobei die großen Cluster nach vorne wandern. Die großen Cluster sind deshalb interessant, weil sie eine große Menge an Sätzen mit einer ähnlichen Ausrichtung haben.

Der SentencePicker bietet dann die Möglichkeit an, die wichtigsten Sätze der Reihe nach abzufragen. Somit stellt der SentencePicker das eigentliche ausführende Modul dar, welches die Wichtigkeit der Sätze letzten Endes festlegt.

Um die Sätze nun mit einem Score zu versehen, berechnet der SentencePicker zuerst die wichtigste Tendenz. Dies macht er, indem er das Zentrum des Clusters wählt und eine Linie durchzieht (siehe 3.8). Aus Gründen der Übersichtlichkeit, wird in den folgenden Darstellungen die Verarbeitung lediglich auf einen und nicht alle Cluster ausgeführt. In der Implementierung muss diese Verarbeitung auf alle Cluster erfolgen.

Für das Ziehen dieser Linie gilt ein bestimmtes Kriterium, auf das ich später noch genauer eingehen werde:

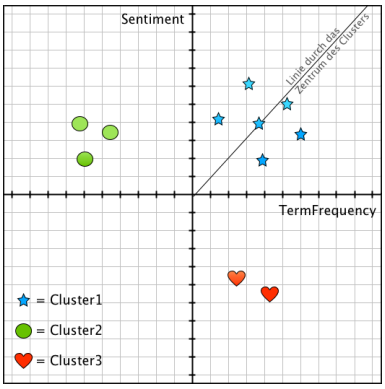

Abbildung 3.8: Wichtigkeitslinie

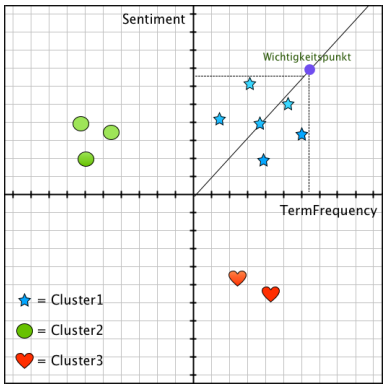

Abbildung 3.9: Wichtigkeitspunkt

Kriterium 1 *Der X-Wert des Wichtigkeitspunkts muss größer oder gleich dem größten X-Wert aller Punkte im Cluster sein, und der Y-Wert des Wichtigkeitspunkts muss größer oder gleich dem größten Y-Wert aller Punkte im Cluster sein.*

Mit "größer" wird hier die absolute Distanz zum Urspungspunkt (0,0) verstanden. Wenn also der WP im negativen Bereich liegt, muss an dieser Stelle im Programm mit Absolutwerten gearbeitet werden. Sobald diese Linie gezogen ist, und sie das eben beschriebene Kriterium erfüllt, wird an dieser Stelle ein neuer Punkt markiert. Dieser Punkt wird mit dem Name **"Wichtigkeitspunkt"** bezeichnet und im weiteren Verlauf mit **"WP"** abgekürzt. Die folgende Abbildung 3.9 stellt diesen Arbeitsschritt grafisch dar.

Wenn mehrere Cluster existieren, wird für jeden Cluster ein solcher WP definiert. Es wird für jeden Satz der Score im Bezug auf den WP seines Clusters berechnet. Der Score eines Satzes wird dadurch definiert, wie kurz die Distanz von dem Punkt, welcher den Satz repräsentiert, zu dem WP ist. Dies wird in der Darstellung 3.10 verdeutlicht.

Zu der Distanz muss gesagt werden, um so geringer sie ist, desto näher liegt der Punkt am Wichtigkeitspunkt und desto wichtiger ist der Satz.

Nun da der grundlegende Ablauf geklärt ist, gehen wir auf das Kriterium 1 näher ein. Die Frage dich sich hierbei stellt ist, warum brauchen wir so einen WP? Würde es nicht aussreichen, wenn wir einfach vom Zentrum des Clusters ausgehen, und

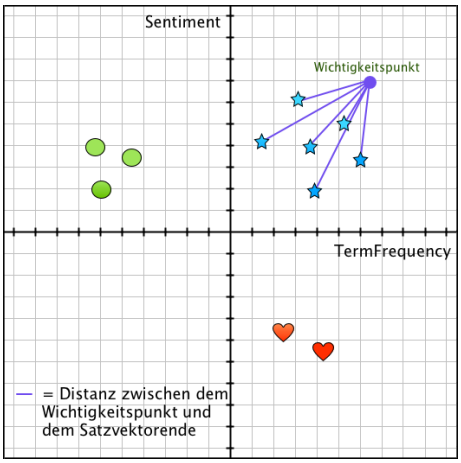

Abbildung 3.10: Der Score des Satzes wird durch die Distanz definiert, die zwischen dem Punkt des Satzes und dem WP liegt.

die Distanzen berechnen? Würde man so vorgehen, dann würden Sätze, die zwar näher am Zentrum sind, aber weiter links liegen (geringerer TF-Score) eine höhere Wichtigkeit bekommen als Sätze die eventuell geringfügig weiter weg aber dafür sehr weit nach rechts verlagert (höherer TF-Score) sind. Um dies zu vermeiden, schieben wir den WP mindestens bis zu den Extremwerten (max-X und max-Y) des Clusters hinaus. In Abbildung 3.10 ist zu sehen, wie die Wichtigkeiten der Sätze ausgehend vom WP bewertet werden.
Für die Auswahl der Sätze aus diesen Clustern gibt es nun zwei mögliche Vorgehensweisen:

1. Es werden erst alle Sätze (ihrer Wichtigkeit nach) aus dem größten Cluster genommen, und sobald ein Cluster leer ist, bedient sich der Algorithmus an dem größten, der noch übrig ist.

2. Es wird der größte Cluster bestimmt, und nun sieht man, ob es noch Cluster mit ähnlicher Größe gibt. Wenn welche gefunden wurden, werden die Sätze abwechselnd aus diesen Clustern gewählt.

Der erste Ansatz bewirkt, dass Sätze aus einer Richtung bevorzugt in der Kurzzusammenfassung aufgenommen werden. Der zweite Ansatz bewirkt, dass die Sätze

aus möglichst vielen Richtungen gewählt werden, um möglichst viele verschiedene (aber dennoch wichtige) Sätze in die Kurzzusammenfassung zu bringen. Beide Ansätze können vertreten werden, und haben gewisse Vor- und Nachteile. Beispielsweise:

- Ansatz 2 kann dazu führen, dass die ausgewählten Sätze keinen Sinn mehr ergeben, da sie sich im Gesamttext auf vorherige Sätze beziehen, die es aber nicht in die Kurzzusammenfassung geschafft haben. Dies kann zwar bei Ansatz 1 ebenfalls passieren, allerdings ist die Wahrscheinlichkeit dort geringer, da bevorzugt Sätze mit ähnlichen Tendenzen gewählt werden.
- Bei Ansatz 1 kann durch das Wählen von ähnlichen Sätzen passieren, dass interessante Nebeninformationen oder wohlmöglich Wendungen im Text wegfallen. Dies ist besonders wahrscheinlich, umso geringer die Länge der Kurzzusammenfassung gewählt wird.

In dem System stehen beide Varianten zur Auswahl und können über den **"mixed"**-Parameter ausgewählt werden. Welcher Ansatz sich mehr bewährt, wird in der Validierung geprüft.

4 Systemvalidierung

4.1 Allgemein

Die Validierung des Systems wurde mittels einer "Golden Standard" Vorgehensweise umgesetzt. Das System erhält als Input vollständige Texte und nach der Verarbeitung wird das Ergebnis mit den von Menschen bewerteten Satzrankings verglichen. Die Satzrankings werden durch eine Liste repräsentiert, in der alle Sätze des Textes nach ihrer Wichtigkeit sortiert (subjektive Einschätzungen) enthalten sind. Der erste Satz stellt dabei den wichtigsten Satz dar, und der letzte Satz den unwichtigsten Satz. Die Evaluierung ist daher als "pseudo-purpose evaluation" einzustufen, da sie zwar über Testdaten verfügt, welche realen Texten ensprechen, allerdings beruht der Bewertungsvorgang nicht auf einer tatsächlichen Kurzzusammenfassung, sondern auf einem Satzranking.
Bei dem Vergleichen wurde bewusst keine von Menschen erzeugte Kurzzusammenfassung genommen, da bei einer Veränderung der Zusammenfassungslänge im System, die von Menschen erzeugte Kurzzusammenfassungen neu erzeugt werden müssten. Durch die Verwendung einer Liste mit nach Wichtigkeit sortierten Sätzen, sind die Validierungsdaten sehr flexibel einsetzbar. Der Vergleich einer Kurzzusammenfassung mit der Liste kann auf zweierlei Arten ablaufen:

1. Es wird der prozentuale Anteil der richtig ausgewählten Sätze in der Kurzzusammenfassung ermittelt.

2. Die Sätze werden mit einem Score gewichtet, und dann zusammengezählt. Hierbei sollte der wichtigste Satz (Satzranking Platz 1) das höchste Gewicht haben. Wenn nun also der erste (wichtigste) Satz fehlt, hat das eine stärkere

Auswirkung auf das Ergebnis als wenn zum Beispiel der viertwichtigste Satz fehlt.

Hier eine detailiertere Beschreibung für Variante 1:

Für die Anzahl an Sätzen in der Kurzzusammenfassung, werden die ersten Sätze der Liste entnommen, bis genau so viele Sätze entnommen wurden, wie in der Kurzzusammenfassung enthalten sind. Wenn die Kurzzusammenfassung zum Beispiel 5 Sätze beinhaltet, werden die ersten 5 Sätze aus der Liste genommen (Welche die 5 wichtigsten Sätze darstellen). Nun wird geprüft, ob jeder dieser gewählten Sätze in der Kurzzusammenfassung vorkommt. Ist dies nicht der Fall, werden dementsprechend keine Punkte für diesen Satz vergeben. Wenn also von 5 Sätzen in der Auswahl ebenfalls 4 in der Kurzzusammenfassung vorkommen, wurde ein Score von 4/5 also 80% erreicht.

In Pseudo-Code sieht das folgendermaßen aus:

```
1  double validate(List vali_list):
2      i = 0;
3      matches = 0;
4          possibleMatches = summary.size;
5
6          while i < summary.size:
7              if summary.contains(vali_list(i)):
8                  matches += 1;
9              i++
10         return matches / possibleMatches;
```

Hier eine detailierte Beschreibung für Variante 2:

Der Anfang ist der Gleiche wie bei Variante 1, es werden so viele Sätze aus der Liste entnommen wie es Sätze in der Kurzzusammenfassung gibt. Der Unterschied ist, dass ab nun der Score pro Satz auf folgende Art und Weise berechnet wird:

S_s = *Score des Satzes*
N = *Länge der Kurzzusammenfassung (Anzahl Sätze)*

$x =$ *Number des Satzes*

$$Sum(n) = \frac{n^2 + n}{2} \tag{4.1}$$

$$S_s(x) = \frac{N - x}{Sum(N)} \tag{4.2}$$

Beispiel: Es gibt 5 Sätze in der Kurzzusammenfassung:

$$S_0 = \frac{N - x}{Sum(N)} = \frac{5 - 0}{15} = 0.\overline{33} \tag{4.3}$$

$$S_4 = \frac{N - x}{Sum(N)} = \frac{5 - 4}{15} = 0.0\overline{6} \tag{4.4}$$

Der erste Satz hat ein Gewicht von $0.\overline{33}$ während der letzte Satz ein Gewicht von $0.0\overline{6}$ hat.

In Pseudo-Code sieht das folgendermaßen aus:

```
1  double validate(List vali_list):
2      i = 0;
3      score = 0;
4      possibleMatches = summary.size;
5
6      while i < summary.size:
7          if summary.contains(vali_list(i)):
8              reward = (vali_list.size - i) / sum(vali_list.size);
9              score += reward;
10         i++
11     return score;
```

4.2 Validierungsszenario

Nun da klar ist, wie die Zusammenfassung validiert wird, stellt sich die Frage, nach was genau soll eigentlich validiert werden. Der Zweck der eigentlichen Arbeit war es zu zeigen, dass durch das Hinzufügen einer Sentimentanalyse zu dem bestehenden TF-Score, die Qualität von Kurzzusammenfassungen verbessert werden kann.

Daraus leitet sich ein Szenario ab, in dem die Ergebnisse des Systems unter Verwendung von Sentimentanalyse mit den Ergebnissen ohne die Sentimentanalyse verglichen werden.

Desweiteren bietet das System diverse Parameter bei denen überprüft werden soll, wie genau sie sich auf die Qualität der Kurzzusammenfassungen auswirken. Diese Parameter sind folgende:

- **useSentiment**
  Über diesen Parameter kann man dem System mitteilen, ob es die Sentimentanalyse durchführen soll, oder ausschließlich den TF-Score verwendet.
- **k**
  Das k ist die Anzahl an Clustern welche von der Satzauswahl für die Sentimentanalyse genutzt wird.
- **useAverage**
  Dieser Parameter entscheidet, ob der TF-Score der Sätze über den Durchschnitt der TF-Scores der Terme berechnet werden soll, oder mittels einer Aufsummierung.
- **mixed**
  Mixed bedeutet, dass die Sätze nach dem Clustering so gewählt werden, dass wenn mehrere ähnlich große Cluster existieren, der Algorithmus die Sätze aus diesen Clustern abwechseln entnimmt. Ist dieser Wert auf false, werden erst alle Sätze aus dem größten Cluster genommen, und danach aus den jeweils kleineren Clustern.

Für die Validierungen werden nun 5 Ansätze verfolgt:

1. **TF-Only**
   Hierbei handelt es sich um die klassischen Vorgehensweise bei der die Termhäufigkeit die ausschlaggebende Quelle für die Bewertung darstellt. Gegen diesen Ansatz müssen sich nun alle anderen, welche die Sentimentanalyse verwenden, behaupten.

2. **SSN**
   SSN steht für Sentiment + Summierung + NotMixed. Das bedeutet, dass dieser Ansatz die Sentimentanalyse betreibt, die TF-Scores der Sätze aus der Aufsummierung der Term-TF-Scores berechnet, und einem dominierenden Cluster zur Satzauswahl verwendet.

3. **SSM**
   SSM steht für Sentiment + Summierung + Mixed. Es ist quasi das Gleiche wie der SSN, nur dass statt einem dominierenden Cluster die Sätze aus mehreren Clustern genommen werden.

4. **SAN**
   SAN steht für Sentiment + Durchschnitt (Average) + NotMixed. Der Unterschied zum SSN besteht darin, dass die TF-Scores der Sätze nun nicht mehr aufsummiert werden, sondern der Durchschnitt ermittelt wird.

5. **SAM**
   SAM steht für Sentiment + Durchschnitt (Average) + Mixed. Die einzelnen Begriffserklärungen sind wie in den vorherigen Punkten zu verstehen.

Diese für die Validierung vorgesehenen Durchführungen sind der Tabelle 8.3 zu entnehmen. Im weiteren Verlauf werden die oben stehenden Bezeichnung häufig um eine Ziffer ergänzt erscheinen. Diese Ziffer steht für das k, welches die Anzahl der Cluster in der Satzauswahl repräsentiert.

4.3 Datenauswahl

Die Daten für die Validierung bestehen aus einem Set von Zeitungsartikeln der New York Times aus dem Zeitrraum 1. Quartal 2014. Diese Artikel weisen eine gute Mi-

schung aus langen und kurzen Texten auf. Die Themen sind aus den unterschiedlichsten Bereichen, von Weltpolitik über Naturkatastrophen bis hin zu Filmkritiken oder Kochrezepten.

Zu den Daten lassen sich folgende statistische Angaben machen:
Anzahl der Texte: 50
Längster Text: 39 Sätze
Kürzester Text: 5 Sätze
Durchschnittliche Textlänge: 17.56 Sätze

Anzahl der Sätze insgesamt: 878
Längster Satz: 66 Terme
Kürzester Satz: 2 Terme
Durchschnittliche Satzlänge: 26.34 Terme

Anzahl der Terme insgesamt: 23131

Themenverteilung:
Weltpolitik: 28%
Kultur/Kunst: 28%
Geschichtlich: 6%
Wirtschaft: 12%
National: 26%

4.4 Validierungsablauf

Die Validierung wir in der ersten Runde, die beste Parameterwahl für die 5 Ansätze bestimmen (siehe 8.3). Dafür wird für jeden Text auf dem Set eine Kurzzusammenfassung mit den gewählten Parametern erzeugt, und diese dann gegen das Satzranking validiert. Die entstandenen Ergebnisse werden innerhalb des Ansatzes verglichen. Nachdem aus jedem der 5 Ansätze die beste Parameterwahl ermittelt wurde, werden

diese 5 gegeneinander antreten. Dies bedeutet, dass die 4 Ansätze mit Sentimentanalyse versuchen bessere Ergebnisse zu liefern als der “TF-Only”-Ansatz. Wenn dies gelingt, stärkt das die Hypothese dieser Arbeit, falls nicht, gilt die Hypothese mit dieser Herangehensweise als widerlegt.

4.5 Validierungskriterien

Die nun folgenden Auswertungen beruhen auf den Ergebnisdaten, die während der Validierung entstanden sind. Alle Validierungsergebnisse werden durch eine Prozentzahl dargestellt, die veranschaulicht, in wie weit das Ergebnis der Kurzzusammenfassung, dem Satzranking der Validierung entspricht. Für die Erzeugung dieser Prozentzahl wird nach der Variante 2 (Gewichtete Sätze) vorgegangen. Alle Ansätze mit ihren Parameterwahlen haben einen Endscore. Dieser stellt eine Kenngröße für die Beurteilung des Gesamterfolges dar. Der Endscore wird durch einfaches Aufsummieren der einzelnen Validierungsergebnisse berechnet. Diese Zahl soll dann nicht mehr als eine Prozentzahl verstanden werden, sondern als einheitslose Kennzahl.

4.6 Validierung

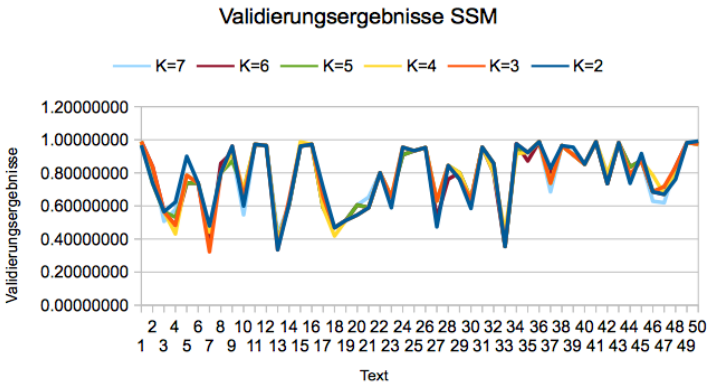

Abbildung 4.1: Validierungsergebnisse des Ansatzes SSM.

Als die beste Clustergröße hat sich bei dem **SSM** k=2 erwiesen, mit einem Endscore von 38.49 Punkten. Der SSM könnte auch mit k=3, k=4, oder k=5 verwendet werden, da die Unterschiede verschwindend gering waren.

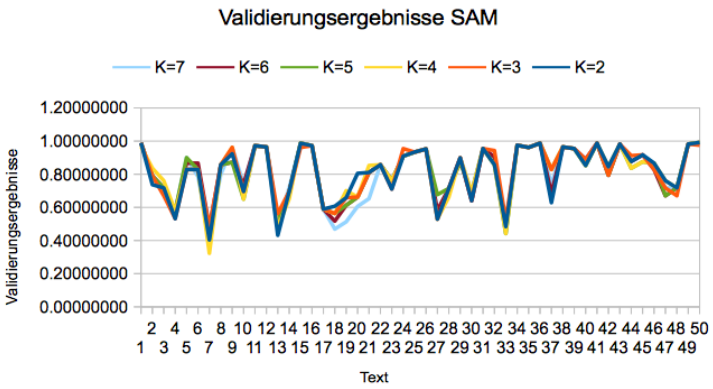

Abbildung 4.2: Validierungsergebnisse des Ansatzes SAM.

Bei dem **SAM** hat sich k=3 als die beste Wahl gezeigt, und ein Endscore von 40.98 Punkten ergeben.

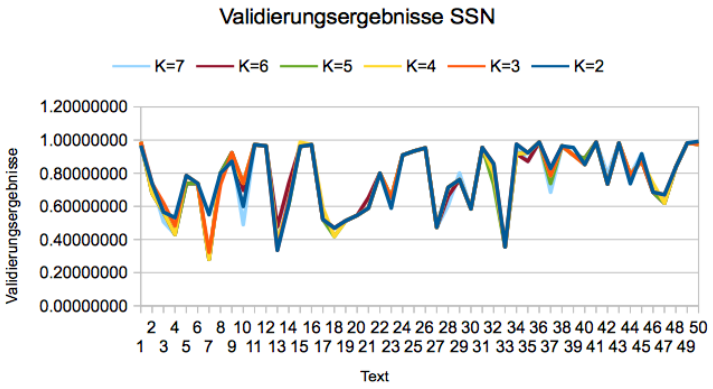

Abbildung 4.3: Validierungsergebnisse des Ansatzes SSN.

Der **SSN** hatte die besten Ergebnisse bei einer Clustergröße von k=2 erzielt, mit einem Endscore von 38.42 Punkten.

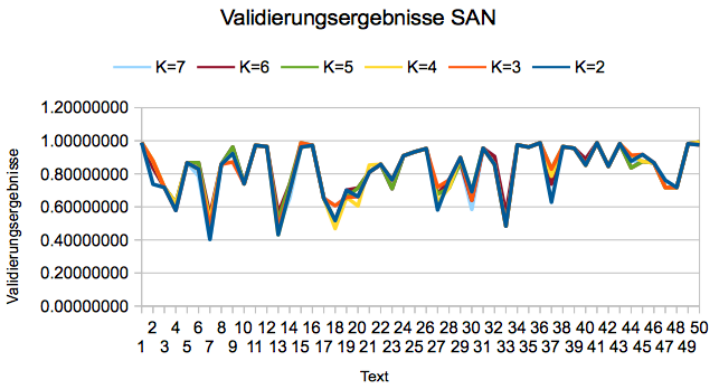

Abbildung 4.4: Validierungsergebnisse des Ansatzes SAN.

Der letzte Ansatz mit Sentimentanalyse, der **SAN** hat seine optimale Leistung bei einer Clustergröße von k=6 mit einem Endscore von 41.57 Punkten.

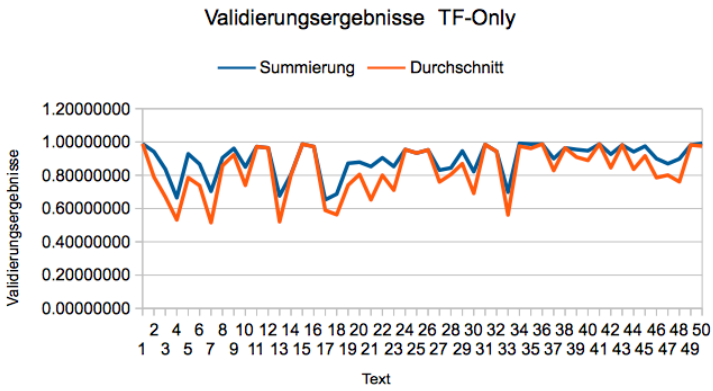

Abbildung 4.5: Validierungsergebnisse des Ansatzes TF-Only.

Bedauerlicherweise konnte keiner dieser Ansätze mit dem **TF-Only** mithalten, der eine Endscore von 44,82 bei der Verwendung von Summierung hatte.

Hier noch einmal alle Ergebnisse im Überblick:

Verfahren	Endscore	Erfolgsquote
SSM	38.49	78.02 %
SAM	40.98	81.96 %
SSN	38.42	77.00 %
SAN	41.57	83.15 %
TF-Only	44.82	89.64%

Abbildung 4.6: Die Ergebnisse der Validierung.

Wie den Grafiken 4.3, 4.1, 4.4 und 4.2 anzusehen ist, gibt es recht starke Schwankung bei den Ergebnissen der sentimentbasierten Kurzzusammenfassungen. Die Ergebnisse des TF-Only (4.5) sehen dagegen wesentlich stabiler aus. Auch das Ändern der Kurzzusammenfassungslänge von 300 auf 150, 200, 400 und 600 Terme hat an den Verhältnissen nichts geändert. Alle Daten zu den Einzelergebnissen sind im Anhang in den Tabellen 8.4, 8.5, 8.6, 8.7 und 8.8 einzusehen. Bei den letzten 4 Tabellen handelt es sich um die spaltenweise Darstellung der Validierungsergebnisse mit den 6 verschiedenen Clustergrößen (2,3,4,5,6,7). Bei der ersten Tabelle handelt es sich um den Vergleich von TF-Only mit Summierung bzw. mit Durchschnittsberechnung. Das Schlussbild (4.7) zeigt den Direktvergleich der 5 Ansätze, und bestätigt die Erkenntnis, dass der TF-Only auf gesamter Strecke besser war.

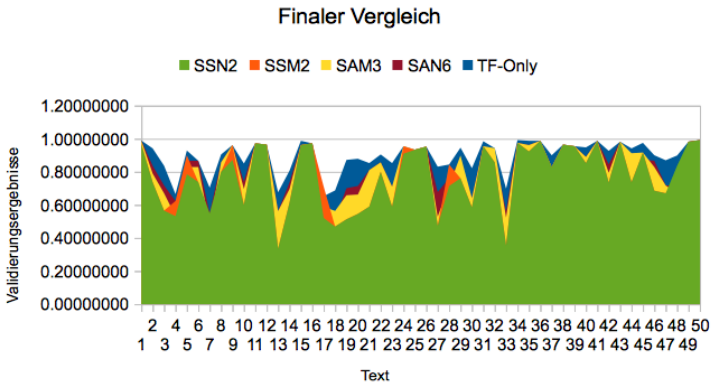

Abbildung 4.7: Vergleich der 5 Ansätze.

Was war die Ursache für solch einen Unterschied?

Um die Ursache zu finden, wurden Stichproben von Texten entnommen, bei denen die Resultate des SAN6 (weil er der Beste unter den sentimentbasierten Ansätzen war) stark von denen des TF-Only abwichen. Bei genauerer Betrachtung ist aufgefallen, das durch die Verwendung der Wichtigkeitspunkte, häufig eher unwichtige Sätze mit jedoch ähnlicher Sentimentbewertung überbewertet wurden. Die Verwendung der Wichtigkeitspunkte ist somit in Frage zu stellen. Ebenso hat sich das gemeinsame Verwenden von TF-Score und Sentimentanalyse als Basis für die Vektoren als eine eher ungünstige Kombination erwiesen, dazu allerdings mehr im Fazit. Ebenso sollte die Verwendung des K-Means Clustering-Algorithmus nochmals überdacht werden. Die Ergebnisse des Clusterings mit dem K-Means waren zu stark von der Wahl der Startpositionen der Cluster abhängig. Mit einem anderen Algorithmus wäre da wahrscheinlich mehr Präzision und Zuverlässigkeit möglich gewesen.

Schlusswort zur Systemvalidierung

Auch wenn die Ergebnisse der Validierung sehr ernüchternd waren, und das Gesamtziel der Arbeit nicht erreicht werden konnten, sind wohl die Ziele einer Validierung erfüllt worden, da sie wichtige Erkenntnisse über die Probleme und das Verbesserungspotential des Systems offengelegt hat. Mehr zur finalen Beurteilung im Fazit.

5 Fazit

5.1 Allgemeines

Das Ziel dieser Arbeit war es, zu prüfen, ob durch das zusätzliche Auswerten einer Sentimentanalyse zum TF-Score, die Qualität der Kurzzusammenfassungen gesteigert werden kann.
Dafür war es wichtig eine Brücke zwischen dem TF-Score und der Sentimentanalyse zu errichten, über welche die Zusatzinformationen aus der Sentimentanalyse in den TF-Score einfließen können. Diese Brücke stellten die erzeugten Vektoren dar. Die Vektoren wurden über Clustering zu Gruppen geordnet und für die Satzauswahl genutzt.
Dieses Vorgehen hat leider nicht die gewünschten Resultate erbracht. Die Verwendung der Sentimentanalyse hat die Ergebnisse eher ungenauer werden lassen. Sowohl mixed, als auch notMixed haben hierbei keinen Unterschied gemacht, die Qualität war durchgehend schlechter. Als Ursache für dieses Ergebnis sind folgende Punkte in den Fokus zu rücken. Es ist in Frage zu stellen, ob die gemeinsame Verwendung von TF-Score und Sentimentanalyse als Basis für die Vektoren vernünftig ist. Die Satzauswahl hat offentlichtlich, basierend auf dem Clustering, häufiger unwichtige Sätze ausgewählt, als es ohne das Clustern und die Sentimentanalyse passiert wären. Auch wenn die Idee, die Sentimentanalyse und den TF-Score zusammen auszuwerten, plausibel klingt, war diese Umsetzung nicht die richtige.

5.2 Ausblick

- Die Wichtigkeitspunkte haben die Qualität der Zusammenfassungen nicht verbessert sondern eher noch verschlechtert. Die Idee mit dem Clustering ist wohl nach wie vor plausibel, allerdings sollten die Wichtigkeitspunkte nichtmehr verwendet werden um die Wichtigkeit der Sätze in den Clustern zu bestimmen. Hierfür liegt es nahe ein Verfahren zu nehmen, welches mehr Rücksicht auf den wirklichen TF-Score des Satzes nimmt.

- Die Basis des Sobels sollte nochmals überdacht werden. Es scheint keine so gute Kombination zu sein, wenn der TF-Score und die Sentimentanalyse als Basis verwendet werden.

- Es wäre interessant zu prüfen, wie sich die Qualität verändert, wenn die Basis für den Winkel des Vektors aus der Sentimentanalyse und eine bis jetzt noch unbekannten Größe bestünden, und der TF-Score die Länge des Vektors bestimmt. Dies wäre eine interessante Modifikation die nach jetzigen Ermessen und mit den hier gewonnenen Wissen als ein plausibelnerer Ansatz wirkt und bessere Chancen hat den TF-Only übertreffen zu könnte.

- Ein Standard für die Validierung von TS-Systemen wäre eine große Bereicherung. Momentan werden die Validierungen hauptsächlich auf eigenständige Arbeit, oder durch kleine Gruppen geplant und durchgeführt. Das bedeutet, dass jeder seine eigenen Gewichtungen in die Validierung einfließen lässt, und ein Vergleichen zwischen den Systemen nur schwer möglich ist. Desweiteren ist es sehr schwer Daten für die Validierung solcher Systeme zu bekommen. Ein öffentlicher Standard wäre eine klare Verbesserung der Situation.

- Es sollten in weiteren Versuchen mit anderen Clustering-Algorithmen experimentiert werden. Ein Clustering-Algorithmus, der zuverlässiger die Cluster berechnet könnte durchaus die Stabilität der Ergebnisse verbessern.

5.3 Schlusswort

Alles in Allem halte ich die Idee hinter diesem Ansatz nach wie vor für gut, allerdings bedarf es einer neuen Idee für die Auswertung der Sentimentanalyse. Das Kombinieren mit dem TF-Score zu einem Vektor hat nicht die gewünschten Resultate gebracht. Ebenso wären Fortschritte in der Sentimentanalyse von großem Nutzen. Mit einer höheren Auflösung der Sentimentanlayse, also mehr als 5 Klassen, oder sogar mit stetigen Werten, könnte diesen Ansatz zu besseren Ergebnissen verhelfen.

6 Hilfsmittel

- Stanford CoreNLP v224n - 2013-09-24
  Stanford's Suite of NLP Tools
- Eclispe SDK 4.2.1
- Java 1.7 (Plattform Mac OS X, 10.9.2)
- yEd 3.12.2 (Für die Diagramme)
- Modelio 3.1.0 (Für die Diagramme)
- LibreOffice 4.1.5.3 (Für die Diagramme)
- Gnuplot 4.6 (Für die Grafiken)

7 Literaturverzeichnis

[Luhn,1958] "The Automatic Creation of Literature Abstracts", Hans Peter Luhn, IBM Journal, April 1958.

[Bush,1945] "As we may think", Vannevar Bush, The Atlantic Monthly, Juli 1945.

[Rath et al,1961] "The formation of abstracts by the selection of sentences", Rath, G.J., Resnick, A. and Savage, T.R., American Documentation. 1961, 139-143.

[RN,2012] "Künstliche Intelligenz - Ein moderner Ansatz", Stuart Russel - Peter Norvig, 3. Auflage, 2012

[ST,2013] "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank", Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng and Christopher Potts, Stanford University, Stanford, CA 94305, USA, 2013

[Carbonell, Goldstein, 1998] - Carbonell, J. and Goldstein, J. (1998) 'The use of MMR and diversity-based reranking for reordering documents and producing summaries', 1998, 21st Annual International ACM-SIGIR Conference (SIGIR 1998), 335-36.

[H.P. Luhn, 1957] - "A statistical approach to mechanized encoding and searching of literary information", 1957, IBM Journal Research and Development, Vol. 1, pp. 309-317.

[Reithinger, 2000] - Norbert Reithinger, "Summarizing Multilingual Spoken Negotiation Dialogues", 2000, DFKI GmbH

[K.S. Jones, 2007] - Karen Spärck Jones, "Automatic summarising: a review and discussion of the state of the art", 2007, University of Cambridge - Computer Laboratory, ISSN 1476-2986

[Leeds,2014] - "The University of Pennsylvania (Penn) Treebank Tag-set", 2014, Leeds University, http://www.comp.leeds.ac.uk/amalgam/tagsets/upenn.html

8 Anhang

Tag	Description	Examples
$	dollar	$ -$ --$ A$ C$ HK$ M$ NZ$ S$ U.S.$ US$
``	opening quotation mark	` ``
''	closing quotation mark	' ''
(	opening parenthesis	([{
)	closing parenthesis	)] }
,	comma	,
--	dash	--
.	sentence terminator	. ! ?
:	colon or ellipsis	: ; ...
CC	conjunction, coordinating	& 'n and both but either et for less minus neither nor or plus so therefore times v. versus vs. whether yet
CD	numeral, cardinal	mid-1890 nine-thirty forty-two one-tenth ten million 0.5 one forty-seven 1987 twenty '79 zero two 78-degrees eighty-four IX '60s .025 fifteen 271,124 dozen quintillion DM2,000 ...
DT	determiner	all an another any both del each either every half la many much nary neither no some such that the them these this those
EX	existential there	there
FW	foreign word	gemeinschaft hund ich jeux habeas Haementeria Herr K'ang-si vous lutihaw alai je jour objets salutaris fille quibusdam pas trop Monte terram fiche oui corporis ...
IN	preposition or conjunction, subordinating	astride among uppon whether out inside pro despite on by throughout below within for towards near behind atop around if like until below next into if beside ...
JJ	adjective or numeral, ordinal	third ill-mannered pre-war regrettable oiled calamitous first separable ectoplasmic battery-powered participatory fourth still-to-be-named multilingual multi-disciplinary ...
JJR	adjective, comparative	bleaker braver breezier briefer brighter brisker broader bumper busier calmer cheaper choosier cleaner clearer closer colder commoner costlier cozier creamier crunchier cuter ...
JJS	adjective, superlative	calmest cheapest choicest classiest cleanest clearest closest commonest corniest costliest crassest creepiest crudest cutest darkest deadliest dearest deepest densest dinkiest ...
LS	list item marker	A A. B B. C C. D E F First G H I J K One SP-44001 SP-44002 SP-44005 SP-44007 Second Third Three Two * a b c d first five four one six three two
MD	modal auxiliary	can cannot could couldn't dare may might must need ought shall should shouldn't will would
NN	noun, common, singular or mass	common-carrier cabbage knuckle-duster Casino afghan shed thermostat investment slide humour falloff slick wind hyena override subhumanity machinist ...
NNP	noun, proper, singular	Motown Venneboerger Czestochwa Ranzer Conchita Trumplane Christos Oceanside Escobar Kreisler Sawyer Cougar Yvette Ervin ODI Darryl CTCA Shannon A.K.C. Meltex Liverpool ...

Abbildung 8.1: Quelle: [Leeds,2014]

NNPS	noun, proper, plural	Americans Americas Amharas Amityvilles Amusements Anarcho-Syndicalists Andalusians Andes Andruses Angels Animals Anthony Antilles Antiques Apache Apaches Apocrypha ...
NNS	noun, common, plural	undergraduates scotches bric-a-brac products bodyguards facets coasts divestitures storehouses designs clubs fragrances averages subjectivists apprehensions muses factory-jobs ...
PDT	pre-determiner	all both half many quite such sure this
POS	genitive marker	' 's
PRP	pronoun, personal	hers herself him himself hisself it itself me myself one oneself ours ourselves ownself self she thee theirs them themselves they thou thy us
PRP$	pronoun, possessive	her his mine my our ours their thy your
RB	adverb	occasionally unabatingly maddeningly adventurously professedly stirringly prominently technologically magisterially predominately swiftly fiscally pitilessly ...
RBR	adverb, comparative	further gloomier grander graver greater grimmer harder harsher healthier heavier higher however larger later leaner lengthier less-perfectly lesser lonelier longer louder lower more ...
RBS	adverb, superlative	best biggest bluntest earliest farthest first furthest hardest heartiest highest largest least less most nearest second tightest worst
RP	particle	aboard about across along apart around aside at away back before behind by crop down ever fast for forth from go high i.e. in into just later low more off on open out over per pie raising start teeth that through under unto up up-pp upon whole with you
SYM	symbol	% & ' " ".)). * + ,. < = > @ A[fj] U.S U.S.S.R * ** ***
TO	"to" as preposition or infinitive marker	to
UH	interjection	Goodbye Goody Gosh Wow Jeepers Jee-sus Hubba Hey Kee-reist Oops amen huh howdy uh dammit whammo shucks heck anyways whodunnit honey golly man baby diddle hush sonuvabitch ...
VB	verb, base form	ask assemble assess assign assume atone attention avoid bake balkanize bank begin behold believe bend benefit bevel beware bless boil bomb boost brace break bring broil brush build ...
VBD	verb, past tense	dipped pleaded swiped regummed soaked tidied convened halted registered cushioned exacted snubbed strode aimed adopted belied figgered speculated wore appreciated contemplated ...
VBG	verb, present participle or gerund	telegraphing stirring focusing angering judging stalling lactating hankerin' alleging veering capping approaching traveling besieging encrypting interrupting erasing wincing ...
VBN	verb, past participle	multihulled dilapidated aerosolized chaired languished panelized used experimented flourished imitated reunifed factored condensed sheared unsettled primed dubbed desired ...
VBP	verb, present tense, not 3rd person singular	predominate wrap resort sue twist spill cure lengthen brush terminate appear tend stray glisten obtain comprise detest tease attract emphasize mold postpone sever return wag ...
VBZ	verb, present tense, 3rd person singular	bases reconstructs marks mixes displeases seals carps weaves snatches slumps stretches authorizes smolders pictures emerges stockpiles seduces fizzes uses bolsters slaps speaks pleads ...
WDT	WH-determiner	that what whatever which whichever
WP	WH-pronoun	that what whatever whatsoever which who whom whosoever
WP$	WH-pronoun, possessive	whose
WRB	Wh-adverb	how however whence whenever where whereby whereever wherein whereof why

Abbildung 8.2: Quelle: [Leeds,2014]

Nr.	useSentiment	k	useAverage	mixed
	TF-Only			
01	false	0	false	false
02	false	0	true	false
	SSN			
03	true	2	false	false
04	true	3	false	false
05	true	4	false	false
06	true	5	false	false
07	true	6	false	false
08	true	7	false	false
	SSM			
09	true	2	false	true
10	true	3	false	true
11	true	4	false	true
12	true	5	false	true
13	true	6	false	true
14	true	7	false	true
	SAN			
15	true	2	true	false
16	true	3	true	false
17	true	4	true	false
18	true	5	true	false
19	true	6	true	false
20	true	7	true	false
	SAM			
21	true	2	true	true
22	true	3	true	true
23	true	4	true	true
24	true	5	true	true
25	true	6	true	true
26	true	7	true	true

Abbildung 8.3: Die Kombinationsmöglichkeiten der Parameter welche durch die Validierung bewertet werden sollen.

0.98901099	0.98901099
0.94152047	0.78947368
0.83695652	0.67028986
0.66502463	0.53201970
0.92857143	0.78571429
0.86666667	0.73809524
0.70384615	0.51538462
0.90476190	0.85714286
0.96153846	0.92307692
0.85000000	0.74000000
0.97222222	0.97222222
0.96428571	0.96428571
0.67613636	0.52083333
0.80632411	0.80237154
0.98717949	0.98717949
0.97222222	0.97222222
0.65263158	0.58947368
0.68735632	0.56321839
0.87179487	0.74074074
0.87878788	0.80519481
0.85263158	0.65263158
0.90476190	0.80000000
0.85263158	0.71052632
0.95454545	0.95454545
0.93333333	0.93333333
0.95238095	0.95238095
0.83076923	0.76000000
0.84415584	0.80519481
0.94565217	0.86956522
0.82213439	0.69169960
0.98484848	0.98484848
0.94285714	0.94285714
0.69841270	0.56190476
0.99166667	0.97500000
0.98717949	0.96153846
0.98717949	0.98717949
0.90000000	0.82857143
0.96428571	0.96428571
0.95454545	0.90909091
0.94736842	0.88947368
0.98717949	0.98717949
0.92647059	0.84558824
0.98181818	0.98181818
0.94152047	0.83625731
0.97500000	0.91666667
0.90000000	0.78571429
0.86956522	0.80072464
0.89855072	0.76086957
0.98181818	0.98181818
0.99166667	0.97500000

Abbildung 8.4: Links: TF-Only mit Summierung; Rechts: TF-Only mit Durchschnitt

0.96703297	0.98901099	0.98901099	0.98901099	0.98901099	0.98901099
0.73684211	0.83625731	0.73684211	0.73684211	0.83625731	0.83625731
0.56521739	0.56521739	0.56521739	0.56521739	0.56521739	0.50724638
0.62315271	0.48275862	0.43103448	0.53201970	0.48275862	0.57881773
0.90000000	0.78571429	0.78571429	0.73809524	0.73809524	0.78571429
0.73809524	0.73809524	0.73809524	0.73809524	0.73809524	0.73809524
0.47948718	0.32307692	0.44230769	0.44230769	0.36410256	0.36410256
0.80000000	0.80000000	0.80000000	0.80000000	0.85714286	0.85714286
0.96153846	0.96153846	0.92307692	0.87179487	0.92307692	0.92307692
0.60000000	0.65000000	0.69666667	0.69666667	0.65000000	0.54666667
0.97222222	0.97222222	0.97222222	0.97222222	0.97222222	0.97222222
0.96428571	0.96428571	0.96428571	0.96428571	0.96428571	0.96428571
0.33522727	0.33522727	0.38446970	0.38446970	0.38446970	0.43181818
0.61264822	0.64031621	0.61264822	0.61264822	0.60474308	0.60474308
0.96153846	0.96153846	0.98717949	0.96153846	0.96153846	0.96153846
0.97222222	0.97222222	0.97222222	0.97222222	0.97222222	0.97222222
0.71052632	0.65263158	0.58947368	0.58947368	0.58947368	0.58947368
0.46896552	0.46896552	0.41839080	0.46896552	0.46896552	0.46896552
0.51282051	0.51282051	0.51282051	0.51282051	0.51282051	0.51282051
0.54545455	0.54545455	0.54545455	0.60606061	0.60606061	0.60606061
0.58947368	0.58947368	0.58947368	0.58947368	0.58947368	0.65263158
0.80000000	0.80000000	0.80000000	0.80000000	0.80000000	0.80000000
0.58947368	0.65263158	0.65263158	0.65263158	0.65263158	0.65263158
0.95454545	0.95454545	0.95454545	0.90909091	0.90909091	0.90909091
0.93333333	0.93333333	0.93333333	0.93333333	0.93333333	0.93333333
0.95238095	0.95238095	0.95238095	0.95238095	0.95238095	0.95238095
0.47384615	0.63076923	0.63076923	0.63076923	0.52923077	0.63076923
0.84415584	0.84415584	0.84415584	0.84415584	0.76190476	0.76190476
0.76086957	0.76086957	0.80072464	0.80072464	0.80072464	0.80072464
0.58498024	0.64031621	0.64031621	0.64031621	0.64031621	0.64031621
0.95454545	0.95454545	0.95454545	0.95454545	0.95454545	0.95454545
0.85714286	0.85714286	0.80000000	0.85714286	0.80000000	0.80000000
0.35555556	0.35555556	0.44285714	0.35555556	0.35555556	0.35555556
0.97500000	0.97500000	0.91666667	0.95000000	0.97500000	0.97500000
0.92307692	0.92307692	0.92307692	0.92307692	0.87179487	0.92307692
0.98717949	0.98717949	0.98717949	0.98717949	0.98717949	0.98717949
0.82857143	0.73809524	0.73809524	0.73809524	0.78571429	0.68571429
0.96428571	0.96428571	0.96428571	0.96428571	0.96428571	0.96428571
0.95454545	0.90909091	0.90909091	0.90909091	0.90909091	0.90909091
0.85263158	0.85263158	0.85263158	0.85263158	0.85263158	0.85263158
0.98717949	0.98717949	0.98717949	0.98717949	0.98717949	0.98717949
0.73529412	0.73529412	0.79411765	0.73529412	0.73529412	0.79411765
0.98181818	0.98181818	0.98181818	0.98181818	0.98181818	0.98181818
0.73684211	0.78947368	0.78947368	0.83625731	0.83625731	0.83625731
0.91666667	0.87500000	0.87500000	0.87500000	0.87500000	0.87500000
0.68571429	0.68571429	0.78571429	0.68571429	0.68571429	0.62857143
0.67028986	0.71739130	0.67028986	0.71739130	0.67028986	0.61956522
0.76086957	0.83695652	0.80072464	0.80072464	0.83695652	0.83695652
0.98181818	0.98181818	0.98181818	0.98181818	0.98181818	0.98181818
0.99166667	0.97500000	0.97500000	0.97500000	0.97500000	0.97500000

Abbildung 8.5: Sentiment mit Durschnitt und ohne Mixed, k=2 bis k=7

0.98901099	0.98901099	0.96703297	0.98901099	0.98901099	0.98901099
0.73684211	0.78947368	0.83625731	0.78947368	0.78947368	0.78947368
0.71739130	0.67028986	0.76086957	0.71739130	0.71739130	0.71739130
0.53201970	0.53201970	0.57881773	0.57881773	0.57881773	0.57881773
0.82857143	0.82857143	0.82857143	0.90000000	0.86666667	0.90000000
0.82857143	0.82857143	0.82857143	0.82857143	0.86666667	0.78571429
0.40384615	0.47948718	0.32307692	0.36410256	0.47948718	0.44230769
0.85714286	0.85714286	0.85714286	0.85714286	0.85714286	0.80000000
0.92307692	0.96153846	0.96153846	0.87179487	0.87179487	0.96153846
0.69666667	0.69666667	0.65000000	0.65000000	0.74000000	0.74000000
0.97222222	0.97222222	0.97222222	0.97222222	0.97222222	0.97222222
0.96428571	0.96428571	0.96428571	0.96428571	0.96428571	0.96428571
0.43181818	0.56250000	0.47727273	0.52083333	0.52083333	0.47727273
0.69565217	0.69565217	0.65217391	0.69565217	0.69565217	0.71936759
0.98717949	0.96153846	0.98717949	0.98717949	0.98717949	0.98717949
0.97222222	0.97222222	0.97222222	0.97222222	0.97222222	0.97222222
0.58947368	0.58947368	0.58947368	0.58947368	0.58947368	0.58947368
0.60689655	0.56321839	0.56321839	0.56321839	0.51724138	0.46896552
0.65811966	0.65811966	0.70085470	0.61253561	0.61253561	0.51282051
0.80519481	0.66233766	0.66233766	0.66233766	0.66233766	0.60606061
0.81052632	0.81052632	0.85263158	0.81052632	0.81052632	0.65263158
0.85714286	0.85714286	0.85714286	0.85714286	0.85714286	0.85714286
0.71052632	0.71052632	0.76315789	0.76315789	0.76315789	0.76315789
0.90909091	0.95454545	0.90909091	0.90909091	0.90909091	0.90909091
0.93333333	0.93333333	0.93333333	0.93333333	0.93333333	0.93333333
0.95238095	0.95238095	0.95238095	0.95238095	0.95238095	0.95238095
0.52923077	0.52923077	0.52923077	0.67692308	0.58153846	0.58153846
0.71428571	0.71428571	0.66233766	0.71428571	0.71428571	0.71428571
0.89855072	0.89855072	0.89855072	0.86956522	0.89855072	0.89855072
0.64031621	0.64031621	0.69169960	0.69169960	0.64031621	0.69169960
0.95454545	0.95454545	0.95454545	0.95454545	0.95454545	0.95454545
0.85714286	0.94285714	0.85714286	0.85714286	0.90476190	0.90476190
0.48412698	0.52380952	0.44285714	0.44285714	0.48412698	0.44285714
0.97500000	0.97500000	0.97500000	0.97500000	0.97500000	0.97500000
0.96153846	0.96153846	0.96153846	0.96153846	0.96153846	0.96153846
0.98717949	0.98717949	0.98717949	0.98717949	0.98717949	0.98717949
0.62857143	0.82857143	0.82857143	0.82857143	0.68571429	0.73809524
0.96428571	0.96428571	0.96428571	0.96428571	0.96428571	0.96428571
0.95454545	0.95454545	0.95454545	0.95454545	0.95454545	0.95454545
0.85263158	0.88947368	0.85263158	0.85263158	0.88947368	0.88947368
0.98717949	0.98717949	0.98717949	0.98717949	0.98717949	0.98717949
0.84558824	0.79411765	0.84558824	0.79411765	0.79411765	0.79411765
0.98181818	0.98181818	0.98181818	0.98181818	0.98181818	0.98181818
0.87719298	0.91228070	0.83625731	0.83625731	0.87719298	0.87719298
0.91666667	0.91666667	0.87500000	0.87500000	0.91666667	0.91666667
0.86666667	0.82857143	0.86666667	0.86666667	0.82857143	0.86666667
0.76086957	0.71739130	0.76086957	0.67028986	0.67028986	0.71739130
0.71739130	0.67028986	0.71739130	0.71739130	0.71739130	0.71739130
0.98181818	0.98181818	0.98181818	0.98181818	0.98181818	0.98181818
0.99166667	0.97500000	0.99166667	0.99166667	0.99166667	0.99166667

Abbildung 8.6: Sentiment mit Summierung und Mixed, k=2 bis k=7

0.85714286	0.85714286	0.85714286	0.85714286	0.85714286	0.85714286
0.68421053	0.80701754	0.80701754	0.80701754	0.77192982	0.77192982
0.67753623	0.67753623	0.67753623	0.67753623	0.67753623	0.67753623
0.55418719	0.55418719	0.59605911	0.59605911	0.59605911	0.55418719
0.80476190	0.80476190	0.80476190	0.80476190	0.80476190	0.80476190
0.77142857	0.77142857	0.77142857	0.80476190	0.80476190	0.73333333
0.39230769	0.46538462	0.50000000	0.50000000	0.53333333	0.39230769
0.77142857	0.77142857	0.77142857	0.77142857	0.77142857	0.77142857
0.80769231	0.76923077	0.80769231	0.83333333	0.83333333	0.83333333
0.70000000	0.70000000	0.70000000	0.70000000	0.70000000	0.70000000
0.77777778	0.77777778	0.77777778	0.77777778	0.77777778	0.77777778
0.75000000	0.75000000	0.75000000	0.75000000	0.75000000	0.75000000
0.41666667	0.46022727	0.41666667	0.50189394	0.54166667	0.46022727
0.65217391	0.65217391	0.65217391	0.68774704	0.68774704	0.61264822
0.83333333	0.84615385	0.83333333	0.84615385	0.83333333	0.83333333
0.77777778	0.77777778	0.77777778	0.77777778	0.77777778	0.77777778
0.61052632	0.61052632	0.61052632	0.61052632	0.61052632	0.61052632
0.49655172	0.58160920	0.45057471	0.49655172	0.49655172	0.58160920
0.66666667	0.62678063	0.62678063	0.62678063	0.66666667	0.62678063
0.62337662	0.62337662	0.57142857	0.67099567	0.67099567	0.62337662
0.75263158	0.75263158	0.78947368	0.75263158	0.75263158	0.75263158
0.77142857	0.77142857	0.77142857	0.77142857	0.77142857	0.77142857
0.71052632	0.71052632	0.71052632	0.66315789	0.66315789	0.66315789
0.78787879	0.78787879	0.78787879	0.78787879	0.78787879	0.78787879
0.66666667	0.66666667	0.66666667	0.66666667	0.66666667	0.66666667
0.71428571	0.71428571	0.71428571	0.71428571	0.71428571	0.71428571
0.55384615	0.68307692	0.60000000	0.64307692	0.64307692	0.64307692
0.71428571	0.71428571	0.67099567	0.67099567	0.71428571	0.71428571
0.84057971	0.84057971	0.84057971	0.81521739	0.81521739	0.81521739
0.65217391	0.60474308	0.65217391	0.65217391	0.60474308	0.55335968
0.81818182	0.81818182	0.81818182	0.81818182	0.81818182	0.81818182
0.77142857	0.77142857	0.77142857	0.77142857	0.80952381	0.80952381
0.46825397	0.46825397	0.46825397	0.46825397	0.54285714	0.50634921
0.86666667	0.86666667	0.86666667	0.86666667	0.86666667	0.86666667
0.83333333	0.83333333	0.83333333	0.83333333	0.83333333	0.83333333
0.84615385	0.84615385	0.84615385	0.84615385	0.84615385	0.84615385
0.59047619	0.77142857	0.73333333	0.77142857	0.69047619	0.73333333
0.75000000	0.75000000	0.75000000	0.75000000	0.75000000	0.75000000
0.81818182	0.81818182	0.81818182	0.81818182	0.81818182	0.81818182
0.78947368	0.78947368	0.78947368	0.78947368	0.82105263	0.82105263
0.84615385	0.84615385	0.84615385	0.84615385	0.84615385	0.84615385
0.77205882	0.77205882	0.77205882	0.77205882	0.77205882	0.77205882
0.81818182	0.81818182	0.81818182	0.81818182	0.81818182	0.81818182
0.80701754	0.83625731	0.83625731	0.77192982	0.77192982	0.80701754
0.82500000	0.82500000	0.79166667	0.79166667	0.79166667	0.79166667
0.80476190	0.80476190	0.80476190	0.80476190	0.80476190	0.80476190
0.71739130	0.67753623	0.67753623	0.67753623	0.67753623	0.67753623
0.67753623	0.67753623	0.67753623	0.67753623	0.67753623	0.67753623
0.81818182	0.81818182	0.81818182	0.81818182	0.81818182	0.81818182
0.86666667	0.86666667	0.87500000	0.87500000	0.87500000	0.87500000

Abbildung 8.7: Sentiment mit Durchschnitt und ohne Mixed, k=2 bis k=7

0.98901099	0.98901099	0.98901099	0.98901099	0.98901099	0.98901099
0.73684211	0.87719298	0.87719298	0.87719298	0.83625731	0.83625731
0.71739130	0.71739130	0.71739130	0.71739130	0.71739130	0.71739130
0.57881773	0.57881773	0.62315271	0.62315271	0.62315271	0.57881773
0.86666667	0.86666667	0.86666667	0.86666667	0.86666667	0.86666667
0.82857143	0.82857143	0.82857143	0.86666667	0.86666667	0.78571429
0.40384615	0.47948718	0.51538462	0.51538462	0.55000000	0.40384615
0.85714286	0.85714286	0.85714286	0.85714286	0.85714286	0.85714286
0.92307692	0.87179487	0.92307692	0.96153846	0.96153846	0.96153846
0.74000000	0.74000000	0.74000000	0.74000000	0.74000000	0.74000000
0.97222222	0.97222222	0.97222222	0.97222222	0.97222222	0.97222222
0.96428571	0.96428571	0.96428571	0.96428571	0.96428571	0.96428571
0.43181818	0.47727273	0.43181818	0.52083333	0.56250000	0.47727273
0.69565217	0.69565217	0.69565217	0.73517787	0.73517787	0.65217391
0.96153846	0.98717949	0.96153846	0.98717949	0.96153846	0.96153846
0.97222222	0.97222222	0.97222222	0.97222222	0.97222222	0.97222222
0.65263158	0.65263158	0.65263158	0.65263158	0.65263158	0.65263158
0.51724138	0.60689655	0.46896552	0.51724138	0.51724138	0.60689655
0.70085470	0.65811966	0.65811966	0.65811966	0.70085470	0.65811966
0.66233766	0.66233766	0.60606061	0.71428571	0.71428571	0.66233766
0.81052632	0.81052632	0.85263158	0.81052632	0.81052632	0.81052632
0.85714286	0.85714286	0.85714286	0.85714286	0.85714286	0.85714286
0.76315789	0.76315789	0.76315789	0.71052632	0.71052632	0.71052632
0.90909091	0.90909091	0.90909091	0.90909091	0.90909091	0.90909091
0.93333333	0.93333333	0.93333333	0.93333333	0.93333333	0.93333333
0.95238095	0.95238095	0.95238095	0.95238095	0.95238095	0.95238095
0.58153846	0.72000000	0.63076923	0.67692308	0.67692308	0.67692308
0.76190476	0.76190476	0.71428571	0.71428571	0.76190476	0.76190476
0.89855072	0.89855072	0.89855072	0.86956522	0.86956522	0.86956522
0.69169960	0.64031621	0.69169960	0.69169960	0.64031621	0.58498024
0.95454545	0.95454545	0.95454545	0.95454545	0.95454545	0.95454545
0.85714286	0.85714286	0.85714286	0.85714286	0.90476190	0.90476190
0.48412698	0.48412698	0.48412698	0.48412698	0.56190476	0.52380952
0.97500000	0.97500000	0.97500000	0.97500000	0.97500000	0.97500000
0.96153846	0.96153846	0.96153846	0.96153846	0.96153846	0.96153846
0.98717949	0.98717949	0.98717949	0.98717949	0.98717949	0.98717949
0.62857143	0.82857143	0.78571429	0.82857143	0.73809524	0.78571429
0.96428571	0.96428571	0.96428571	0.96428571	0.96428571	0.96428571
0.95454545	0.95454545	0.95454545	0.95454545	0.95454545	0.95454545
0.85263158	0.85263158	0.85263158	0.85263158	0.88947368	0.88947368
0.98717949	0.98717949	0.98717949	0.98717949	0.98717949	0.98717949
0.84558824	0.84558824	0.84558824	0.84558824	0.84558824	0.84558824
0.98181818	0.98181818	0.98181818	0.98181818	0.98181818	0.98181818
0.87719298	0.91228070	0.91228070	0.83625731	0.83625731	0.87719298
0.91666667	0.91666667	0.87500000	0.87500000	0.87500000	0.87500000
0.86666667	0.86666667	0.86666667	0.86666667	0.86666667	0.86666667
0.76086957	0.71739130	0.71739130	0.71739130	0.71739130	0.71739130
0.71739130	0.71739130	0.71739130	0.71739130	0.71739130	0.71739130
0.98181818	0.98181818	0.98181818	0.98181818	0.98181818	0.98181818
0.97500000	0.97500000	0.99166667	0.99166667	0.99166667	0.99166667

Abbildung 8.8: Sentiment mit Durchschnitt und Mixed, k=2 bis k=7

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 13. Juni 2014 Florian Schlosser