Fachhochschule Nordwestschweiz
Hochschule für Technik

Bachelor-Thesis

Semantic Similarity

Jan Fässler

14. August 2014

Auftraggeber:	Institut für 4D-Technologien, FHNW
Betreuer:	Prof. Dr. André Csillaghy & Simon Felix
Studiengang:	Informatik

Abstract

In dieser Arbeit wird sich mit dem Thema auseinandergesetzt, wie ein Computer die semantischen Ähnlichkeit zweier Wörter bewerten kann. Es wird aufgezeigt, welches der aktuelle Stand der Forschung ist und welche verschiedenen Vorgehensweisen es gibt, die semantische Ähnlichkeit zu messen. Eines der modernsten Verfahren wird genauer betrachtet und implementiert. Mit dieser Implementierung können Werte erreicht werden, welche sich durchaus von Zufallsfunktionen abheben, jedoch kommt die Implementation nicht an die Werte heran, welche mit einer anderen Implementation aber demselben Verfahren erreicht werden können. Am Ende werden jedoch Anhaltspunkte genannt, mit welchen die Möglichkeit besteht, die Werte dieser Implementation zu verbessern.

Inhaltsverzeichnis

1. Einleitung

Diese Arbeit beinhaltet die Entwicklung einer Funktion, welche die semantische Ähnlichkeit von zwei verschiedenen Wörtern ermitteln kann. Sie muss im Stande sein, Wörter wie "Hund" und "Katze" als ähnlicher zu bewerten, als "Hund" und "Zeitgeist". Das Ergebnis dieser Funktion ist ein numerischer Wert sein.

Das Einsatzgebiet einer solchen Funktion ist vielseitig. Sie ermöglicht beispielsweise einer Suchmaschine, Texte vorzuschlagen, nach denen nicht explizit gesucht wurde. Die Suchmaschine sucht nicht nur nach den angegebenen Wörtern, sondern auch nach Wörtern, welche den angegebenen ähnlich sind. Ein Musikprogramm könnte anhand dieser Funktion Bands finden, die ähnliche Musik produzieren und diese als nächstes abspielen. Weiter könnte sie dazu benutzt werden, Wörter thematisch zu sortieren oder um mehrdeutige Wörter zu identifizieren.

Die Funktion kann in zwei Teile aufgeteilt werden. Im ersten Teil werden Daten analysiert und die Funktion wird trainiert. Im zweiten Teil wird die Ähnlichkeit von zwei verschiedenen Wörtern aufgrund der Trainingsdaten bewertet. Die Analyse der Daten muss nur einmal pro Sprache ausgeführt werden, weshalb dieser Schritt auch nicht zeitkritisch ist. Der zweite Teil muss aber das Ergebnis effizient berechnet werden. Dies hat den Grund, dass diese Funktion je nach Einsatzgebiet einige tausend Male nacheinander aufgerufen wird.

Die Funktion wird als ein C#-Programm in einem NuGet-Package umgesetzt. Sie wird mit Daten trainiert um danach die Ähnlichkeiten von zwei verschiedenen Wörtern numerisch wiedergeben zu können.

Eingabe String 1	**Eingabe String 2**	**Ausgabe**
Hund	Katze	0.5
Hund	Zeitgeist	0.1

Tabelle 1.1.: Beispiel für eine Eingabe-Ausgabe

2. Vorgehen

Es gibt mehrere Ausdrücke, in Bezug auf die semantische Ähnlichkeit, welche zwar unterschiedliche Bedeutungen haben, aber oft als Synonym gebraucht werden. Die Ausdrücke, ihre Bedeutung und ihre Unterschiede werden in Kapitel 3 beschrieben.

Es gibt verschiedene Verfahren, welche Lösungen für die beschriebene Aufgabenstellung liefern. Zum einen gibt es Portale wie WordNet oder OpenThesaurus und auf handgemachten Ressourcen basieren. Zum anderen gibt es auch Verfahren, mit denen gute Bewertungen ohne menschliche Unterstützung erzielt werden können. In Abschnitt 3.3 werden die verschiedenen Methoden genauer beschrieben.

2.1. Verfahrensauswahl

Es wurde ein Verfahren ausgewählt, welches ohne handgemachte Ressourcen verwendet werden kann. Die Entscheidung wurde aufgrund der Aktualität und der Bewertungen getroffen. Als Entscheidungsgrundlage für die Bewertungen wurden die Ergebnisse in den entsprechenden Publikation, in denen die Verfahren vorgestellt wurden, genommen. Da es ein tiefes Verständnis der Vorgehensweise erfordert, um die Methode zu verbessern und verfeinern, war auch die Verständlichkeit ein weiteres Kriterium. Aus diesen Gründen wurde das "***Salient Semantic Analysis***"-Verfahren ausgewählt. Im Kapitel 5 wird das Verfahren näher beschrieben.

2.2. Vorbereitungen

Vor der Implementation muss festgelegt werden, wie die Genauigkeit der Bewertungen einer Implementation gemessen werden kann. Dazu wurde ein Bewertungsverfahren definiert, welches in Kapitel 4 genauer beschrieben wird. Gemessen wird die Korrelation der Bewertungen der Funktion mit menschlichen Bewertungen. Diese Korrelationen werden mit denen anderer Implementationen verglichen.

2.3. Implementation

Die grösste Schwierigkeit liegt darin, die Funktion effizient zu machen. Da der zeitliche Rahmen beschränkt ist, darf auch die Laufzeit des ersten Teils, in dem die Implementation trainiert wird, nicht übermässig lange dauern. Die Parallelisierung des Programmcodes ist mit viel Aufwand verbunden. Zusätzlich wird in die Optimierung der CPU- und RAM-Auslastung Zeit investiert. Die Umsetzung der Implementation umfasst das Kapitel 6.

2.4. Analyse

Das Training der Implementation hat eine lange Laufzeit, deshalb konnten nur wenige Tests durchgeführt werden. Es wurde versucht, die Paramater für die Implementation von Anfang an sinnvoll

zu wählen und danach effizient zu optimieren. Die generierten Daten wurden mit Hilfe von Charts analysiert und im Kapitel 7 aufgezeigt. Im Kapitel 8 werden zudem Verbesserungsansätze erläutert, um die Bewertungen zu optimieren.

3. Semantische Ähnlichkeit

3.1. Terminologie

Es gibt drei Begriffe, welche im Kontext der semantischen Ähnlichkeit verwendet werden:

- "**semantische Ähnlichkeit**" (*semantic similarity*)
- "**semantische Bezogenheit**" (*semantic relatedness*)
- "**semantische Distanz**" (*semantic distance*)

Diese Begriffe werden oft als Synonyme verwendet, jedoch ist die Bedeutung der einzelnen Begriffe unterschiedlich.

Der Ausdruck ist "semantische Bezogenheit" umfassender als "semantische Ähnlichkeit". Semantisch ähnliche Wörter haben eine Verbindung zueinander, die sich über ihre Eigenschaften auszeichnen. Beispielsweise bei den Wörtern "Landhaus" - "Wohngebäude" erkennbar. Wörter, die eine "semantisch Bezogenheit" haben, müssen sich nicht ähnlich sein, sonden können eine andere Art von Beziehung haben (BH06):

gegenteilig: "hoch"-"tief"

hierarchisch: "Haus"-"Dach"

funktional: "Stift"-"Papier"

In der Regel verhält sich die "semantische Ähnlichkeit" und die "semantische Bezogenheit" gleichmässig. Ein Beispiel, dass dies nicht immer der Fall ist liefert Resnik: Die Wörter "Auto" und "Benzin" haben auf den ersten Blick einen stärkeren Bezug zueinander als die Wörter "Auto" und "Fahrrad". Tatsächlich sind sich die letzten beiden Wörter ähnlicher, da es sich bei beiden um ein Fortbewegungsmittel handelt (Res95).

Die "semantische Distanz" zweier Wörter wird durch beide Begriffe beeinflusst. Die "semantische Distanz" ist klein, wenn die "semantische Ähnlichkeit" oder die "semantische Bezogenheit" gross ist. Andernfalls, wenn die "semantische Ähnlichkeit" und die "semantische Bezogenheit" klein sind, ist die Distanz gross.

3.2. Messen der semantischen Ähnlichkeit

Die Messverfahren für die semantische Ähnlichkeit und die semantische Bezogenheit basieren auf handgemachten lexikalischen Ressourcen wie beispielsweise WordNet oder OpenThesaurus. Solche sind jedoch nicht in allen Sprachen verfügbar und haben auch nur einen beschränkten Umfang. Das Problem tritt vor allem dann auf, wenn Wörter aus sehr spezialisierten Themengebieten verwendet werden. Um das Problem zu umgehen, wird die Verteilungsähnlichkeit von Wörtern als Proxy für die semantische Ähnlichkeit verwendet. Das bedeutet, welche Wörter werden in welchem Kontext oft verwendet (Kol09).

3.3. Aktuelle Verfahren

Eines der ältesten und am weitesten verbreitetsten Verfahren, ist die "***latent semantic analysis***", kurz LSA. Es wurde 1997 von Landauer und Dumais beschrieben (LD97). In dem Verfahren wird angenommen, das ähnliche Wörter in ähnlichen Texten vorkommen. Es wird gezählt, welches Wort in welchem Paragraph wie oft enthalten ist. Mit einer mathematischen Methode, der Singulärwertzerlegung, werden in der Datenmatrix die Spalten, die für die Paragraphen stehen, reduziert. Die Wörter werden schliesslich mit der Kosinus-Ähnlichkeit, dem Winkel zwischen zwei Vektoren, verglichen (Wik14d).

Eine andere Variante ist das Verfahren "***pointwise mutual information and information retrieval***", kurz PMI-IR, von Turney aus dem Jahre 2001. Es basiert auf statistischen Daten, welche durch Abfragen von Informationen aus Suchmaschinen im Internet gewonnen werden. Da allerdings die meisten Suchmaschinen die Abfragen auf eine gewisse Anzahl in einem Zeitraum begrenzen, ist diese Variante nur beschränkt nutzbar (Tur01).

Im Jahre 2003 beschreiben Dorow et al. einen iterativen Algorithmus, der auf einem Graphen-Model basiert. Dieser Graph repräsentiert Wörter und ihre Beziehungen zueinander. Er wird durch das "*clustering*" von Graphen generiert, welche aus Synonymen von einem mehrdeutigen Wort besteht. Diese Graphen wiederum, werden aus einem grossen Text-Korpus anhand der Häufigkeit, wie die Wörter mit anderen auftreten, generiert (DW03).

Gabrilovich et al. beschrieben im Jahre 2007 ein Verfahren mit dem Namen "***explicit semantic analysis***", kurz ESA. Sie benutzen Techniken aus dem maschinellen Lernen um einen gewichteten Vektor zu erstellen. Die Dimensionen des Vektors bilden die Wikipedia-Artikel. Es wird ermittelt, welches Wort in welchem Artikel wie häufig enthalten ist, um daraus Rückschlüsse auf die Bedeutung der Wörter zu erhalten (GM07).

Kolb stellt 2008 sein eigenes Verfahren vor, dass er **DISCO** (*extracting DIStributionally related words using CO-occurrences*) nennt. Dieses Verfahren beruht auf reinen statistischen Analysen und Berechnungen. Es stützt sich auf die Annahme, dass ähnliche Wörter in einem ähnlichen Kontext verwendet werden (Kol08).

Hassan et al. beschreiben ihr Verfahren im Jahr 2011, welches sie "***salient semantic analysis***" nennen. Sie basiert auf den Artikeln der Wikipedia. Das Verfahren nimmt dabei vor allem die Links zur Hilfe. Die Autoren gehen von der Annahme aus, dass nur die Wörter verlinkt werden, die für das Textverständnis notwendig sind. Daraus lässt sich für jedes Wort einen Vektor bilden, der aussagt, welches Wort wie häufig in der Nähe von verlinkten Wörtern vorkommt. Um die Wörter beziehungsweise ihre Vektoren zu vergleichen, wird auf die Kosinus-Ähnlichkeit zurückgegriffen (HM11).

4. Bewertungsverfahren

Um einen numerischen Wert für die Ähnlichkeit zweier Wörter einschätzen zu können, wird er mit Werten verglichen, die von Menschen bewertet wurden. Dazu gibt es eine Reihe von Standard-Datensätzen, die von Forschenden immer wieder verwenden werden um Ähnlichkeiten von Wörtern zu überprüfen (Dar14).

Standard-Datensätze sind Listen, die pro Zeile jeweils zwei Wörter und die Ähnlichkeit in einer Zahl enthalten. Die Zahlen werden auf Korrelationen zu den errechneten Werten untersucht. Um den Bezug zum späteren Einsatzgebiet der Funktion zu erhalten, wird zusätzlich ein spezialisierter Datensatz erstellt. Dieser enthält explizit mögliche Wörter aus den Einsatzgebieten. Zehn willkürlich ausgewählten Studenten der FHNW bewerten die Wörter auf ihre semantische Ähnlichkeit. Anschliessend wird die Bewertung gemittelt (Anhang A). Allerdings sind diesen Studenten die Unterschiede zwischen der semantischen Ähnlichkeit und der semantischen Bezogenheit überwiegend unbekannt.

Wort 1	**Wort 2**	**Ähnlichkeit (1-10)**
Offerte	Angebot	8.2
Aufwand	Leistung	5.5
Angebot	Leistung	4.4
⋮	⋮	⋮

Tabelle 4.1.: Beispiel eines Datensatzes

Für die Korrelationsberechnung werden zwei Berechnungsmethoden verwendet: Der Bravais-Pearson's Korrelationskoeffizient und der Spearman-Pearson's Rangkorrelationskoeffizient.

4.1. Bravais-Pearson's Korrelationskoeffizient

Der Bravais-Pearson Korrelationskoeffizient ist das Mass für die lineare Abhängigkeit zweier Datenreihen. Er sagt aus, wie stark der lineare Zusammenhang zwischen zwei Datenreihen ist, ohne dabei durch verschiedene Skalierungen irritiert zu werden. Berechnet wird er wie folgt:

$$r = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}} \tag{4.1}$$

Dabei ist x_i der Wert der Datenreihe an der Stelle i und y_i der Wert an der Stelle i der anderen Datenreihe.

$\overline{x}$ und $\overline{y}$ sind die arithmetischen Mittel der jeweiligen Datenreihen (x oder y) (Wik14b).

Eine Datenreihe besteht aus den berechneten Ähnlichkeitswerten der Implementation. Die andere Datenreihe besteht aus den Werten eines Referenzdatensatzes, in dem die Ähnlichkeit von Menschenhand bewertet wurde. In den beiden Datenreihen muss es sich dabei um die Bewertungen derselben Wortpaare in derselben Reihenfolge handeln.

i	**Wortpaar**	**Ergebnissse Funktion** (x)	**menschlicher Referenzwert** (y)
1	Hund - Katze	0.4	3
2	Hund - Dackel	0.6	4
3	Tür - Fenster	0.45	5
⋮	⋮	⋮	⋮

Tabelle 4.2.: Beispieldaten für Korrelationsberechnung

4.2. Spearman-Pearson's Rangkorrelationskoeffizient

Für Datenreihen, die stark von der Normalverteilung abweichen, eignet sich der Rangkorrelationskoeffizient nach Spearman-Pearson besser. Er ist vor allem unempfindlicher gegenüber Ausreissern. Bei dieser Methode werden die Datenreihen ihrer Grösse nach sortiert und mit einer Rangzahl versehen. Somit wird über diese Ränge das Bravis-Pearson Korrelationsverfahren angewendet:

$$\rho = \frac{\sum_{i=1}^{n}(rg(x_i) - \overline{rg(x)})(rg(y_i) - \overline{rg(y)})}{\sqrt{\sum_{i=1}^{n}(rg(x_i) - \overline{rg(x)})^2}\sqrt{\sum_{i=1}^{n}(rg(y_i) - \overline{rg(y)})^2}} \tag{4.2}$$

Dabei sind $rg(x_i)$ und $rg(y_i)$ die jeweilige Rangzahl. Gibt es mehrere Werte mit dem gleichen Betrag, bekommen sie die gleiche Rangzahl, welches sich aus dem arithmetischen Mittel ihrer theoretischen Ränge ergibt. $\overline{rg(x)}$ und $\overline{rg(y)}$ sind die arithmetischen Mittel aller Rangzahlen (Wik14b).

x **Wert**	**fortlaufender Rang**	**Rangzahl**
0.55	1	1
0.46	2	$\frac{2+3}{2} = 2.5$
0.46	3	$\frac{2+3}{2} = 2.5$
0.4	4	$\frac{4+5+6}{3} = 5$
0.4	5	$\frac{4+5+6}{3} = 5$
0.4	6	$\frac{4+5+6}{3} = 5$
0.2	7	7

Tabelle 4.3.: Beispiel für die Rang-Korrelationsberechnung

Die durchschnittliche Rangzahl $\overline{rg(x)}$ wäre in diesem Beispiel: $\frac{1+2.5+2.5+5+5+5+7}{7} = 4$

4.3. Harmonische Mitte

Beide Koeffizienten sind nach Hassan und Mihalcea wichtig um die Ähnlichkeit zweier Wörter zu vergleichen (HM11). Um die Korrelationen zu berechnen, wird die harmonische Mitte zwischen dem Bravais-Pearson's Korrelationskoeffizient (r) und dem Spearman-Pearson's Rangkorrelationskoeffizient (ρ) genutzt. Somit werden beide gleich stark gewichtet. Die harmonische Mitte zwischen den beiden Koeffizienten berechnet sich wie folgt:

$$\mu = \frac{2r\rho}{r + \rho} \tag{4.3}$$

5. Salient Semantic Analysis

Das "*Salient Semantic Analysis*" Verfahren wird von Hassan et al. beschrieben (HM11). Dieses Verfahren wird in dieser Arbeit näher betrachtet und implementiert.

5.1. Vorberechnungen

Aus Wikipedia-Artikeln werden die Links auf andere Artikel als Schlüsselwörter für das Verständnis des Textes betrachtet. Anhand dieser Schlüsselwörter wird für jedes Wort einen Vektor erstellt. Dieser Vektor bildet die Häufigkeiten ab, welche Schlüsselwörter wie oft in Verbindung mit einem Wort vorkommen. Danach werden die Vektoren durch die Berechnung in der Gleichung 5.5 verändert. Die Ähnlichkeit dieser Vektoren wird als die Ähnlichkeit der Wörter, zu denen die Vektoren gehören, angesehen.

5.1.1. Datensammlung

Zuerst werden alle Links eines Artikel gesammelt. Diese repräsentieren die Schlüsselwörter für einen Artikel. Danach wird nach weiteren Vorkommen dieser Schlüsselwörter im Artikel gesucht und markiert. Nach der "*one sense per discourse*"-Heuristik von Gale et al. hat ein Wort, dass mehrmals in einem Diskurs vorkommt, in der Regel dieselbe Bedeutung (GCY92).

Markierte Schlüsselwörter werden mit den Wörtern ergänzt, die in den anderen Artikeln oft ($\geq 95\%$) verlinkt wurden. Diese werden mit einer Methode von Mihalcea et al. festgestellt. Dafür wird das Verhältnis der Anzahl Artikel in denen das Wort ein Schlüsselwort ist ($count(D_{key})$), zu der totale Anzahl Artikel, in denen das Wort vorkommt ($count(D_W)$), berechnet (MC07).

$$P(keyword|W) \approx \frac{count(D_{key})}{count(D_W)} \tag{5.1}$$

5.1.2. Datenaufbereitung

Es gibt einen Korpus C, der aus Wikipedia-Artikeln besteht, von m Wörtern mit einen Wortschatz von N Wörtern und W verschiedene Schlüsselwörtern. In einer $N \times W$ Matrix (E) wird die Häufigkeit des gemeinsamen Auftretens, von Wörtern mit Schlüsselwörtern, innerhalb der nächsten k Schlüsselwörter gespeichert:

$$E_{i,j} = f^k(w_i, c_j) \tag{5.2}$$

Dabei ist f^k die Anzahl des gemeinsamen Auftretens eines Wortes w_i und des Schlüsselwortes c_j innerhalb der Fenstergrösse k.

−3	−2	−1		+1	+2	+3
Haus	ist	ein	Gebäude,	in	dem	**Menschen**

Tabelle 5.1.: Beispiel Fenstergrösse

Wird im Beispiel der Tabelle 5.1 angenommen, dass die Wörter “Haus” und “Menschen” Schlüsselwörter sind, dann gilt für den Text die Gleichungen 5.3 und 5.4.

$$1 = f^k(\text{Gebäude}, \text{Haus}) \tag{5.3}$$

$$1 = f^k(\text{Gebäude}, \text{Menschen}) \tag{5.4}$$

Aus dieser Matrix wird eine weitere $N \times W$ Matrix (P) erstellt, welche die “*pointwise mutual information*”, kurz PMI, repräsentiert. PMI wurde von Church und Hanks beschrieben und liefert Informationen darüber, wie oft zwei Wörter zusammen auftauchen in der Verbindung mit der Angabe des unabhängigen auftauchens (CH90). In einer leicht modifizierten Gleichung berechnen Hassan et al. die Matrix wie folgt:

$$P_{i,j} = log_2 \frac{f^k(w_i, c_j) \times m}{f^C(w_i) \times f^C(c_j)} \tag{5.5}$$

Dabei gibt die Funktion $f^C(x)$ die Häufigkeit eines Wortes im Korpus zurück. Jede Zeile wird gefiltert, so dass nur die besten β_i-Werte weiterhin bestehen und die restlichen auf 0 gesetzt werden.

$$\beta_i = (log_{10}(f^C(w_i)))^2 \times \frac{log_2(N)}{\delta}, \delta \geq 1 \tag{5.6}$$

Dabei ist δ eine Konstante, die aufgrund der Grösse eines Korpus bestimmt werden muss.

5.2. Ähnlichkeit der Wörter

Um die semantische Ähnlichkeit zweier Wörter berechnen zu können, wird die Überlagerung ihrer semantischen Profile gemessen. Das semantische Profil eines Wortes ist der entsprechende Zeilenvektor der Matrix P.

5.2.1. Kosinus-Ähnlichkeit

Die Überlagerung der semantischen Profile wird mit einer modifizierten Variante der Kosinus-Ähnlichkeit der Zeilenvektoren berechnet:

$$Score_{cos}(A, B) = \frac{\sum_{y=1}^{N} (P_{iy} \cdot P_{jy})^\gamma}{\sqrt{\sum_{y=1}^{N} P_{iy}^{2\gamma} \cdot \sum_{y=1}^{N} P_{jy}^{2\gamma}}} \tag{5.7}$$

Mit der Konstante γ wird die Gewichtung der grossen Zahlen kontrolliert. Je grösser γ ist, desto wichtiger werden die Schlüsselwörter, die neben einem Wort sehr häufig auftreten.

5.2.2. SOC-PMI

Eine weitere Möglichkeit um die Überlagerung der semantischen Profile zu berechnen ist eine modifizierte Variante der "*Second Order Co-Occurrence Pointwise Mutual Information*" (SOC-PMI) von Islam et al. (II06):

$$Score_{soc}(A,B) = ln\left(\frac{\sum_{y=1}^{N}(P_{i,y})^{\gamma}}{\beta_i} + \frac{\sum_{y=1}^{N}(P_{j,y})^{\gamma}}{\beta_j} + 1\right) \tag{5.8}$$

5.2.3. Normierung

Die Funktionen der Gleichungen 5.7 und 5.8 tendieren dazu, für nahezu identische Wörter sehr niedrige Werte zu generieren. Dies würde zu einer grossen Lücke zwischen zwei gleichen Wörtern und zwei sehr ähnlichen Wörtern führen. Um die Lücke zu schliessen wird das Ergebnis um den Faktor λ normiert:

$$Sim(A,B) = \begin{cases} 1 & Score(A,B) > \lambda \\ Score(A,B)/\lambda & Score(A,B) \leq \lambda \end{cases} \tag{5.9}$$

6. Implementation

Die Implementation ist in zwei Teile unterteilt. Der erste Teil ist für die Datensammlung, Auswertung und Aufbereitung aus der Wikipedia zuständig. Die aufbereiteten Daten müssen im zweiten Teil schnell herausgelesen werden. Zusätzlich muss so wenig Speicherplatz auf der Festplatte wie möglich benötigt werden. Im zweiten Teil werden zwei Wörter als Input erwartet. Zu den beiden Wörtern werden die entsprechenden aufbereiteten Daten aus dem ersten Schritt verwendet um die Ähnlichkeit der zwei Wörtern zu berechnen.

6.1. Datensammlung und Aufbereitung

Der erste Teil der Implementation ist in drei Phasen unterteilt:

Abbildung 6.1.: Ablauf Teil 1

Als Input benötigt dieser Teil eine XML Datei von einem Datenbank-Backup der Wikipedia, das alle Artikel einer Sprache umfasst (Wik14c).

6.1.1. Vorbereitungsphase

In der ersten Phase wird jeder Wikipedia-Artikel aus der Wikipedia-XML Datei gelesen und mit der *Task Parallel Library* (TPL) (Mic14c) von C# gelesen und verarbeitet. Als Schlüsselwörter werden Links auf andere Wikipedia-Artikel erkannt. Es wird gezählt, welche Wörter und Schlüsselwörter wie häufig unabhängig von einander auftreten. Anschliessend wird berechnet, welche Wörter am häufigsten verlinkt sind.

Damit Varianten von Wörtern, wie “auffällig” und “auffälligen”, als ein Begriff erkannt werden, wird jeweils der Wortstamm verwendet. Deshalb werden alle Wörter auf ihren Wortstamm reduziert. Hierfür wird eine *Stemming Library* verwendet (Eas14). Sie basiert auf dem “*Snowball*” Algorithmus von Martin Porter (Por01).

Um die späteren Phasen zu beschleunigen und den Speicherverbrauch zu reduzieren, werden die Schlüsselwörter beschränkt. Es werden nur Schlüsselwörter genommen, die auch wirklich einen Wikipedia-Artikel in der aktuellen Sprache haben. Zusätzlich kann die Anzahl der Wörter auf die häufigsten n

Begriffe beschränkt werden. Erläuterungen zu den Gründen finden sich in den Abschnitten 7.1 und 7.2 wieder.

6.1.2. Zählung der Auftretenshäufigkeiten

In der zweiten Phase werden ein zweites Mal alle Wikipedia-Artikel aus der Wikipedia-XML Datei mit TPL (Mic14c) gelesen. Für jeden Artikel wird als erstes ermittelt, welche Wörter Schlüsselwörter im aktuellen Artikel sind. Hierfür werden alle Links auf andere Artikel genommen, die nicht verlinkten Wiederholungen sowie diejenigen Begriffe, die in anderen Artikeln häufig verlinkt werden.

Danach wird gezählt, welche Wörter wie oft in der Nähe von Schlüsselwörtern vorkommen. In der Nähe bedeutet innerhalb einer bestimmten Anzahl Wörter. Die Summen werden in einer Matrix mit den Summen der anderen Artikeln zusammengezählt. Anhand des Beispielsatzes mit den zwei fett markierten Schlüsselwörter ergibt sich bei einer Fenstergröße plus/minus drei Wörtern die Matrix in der Tabelle 6.1: "*Ein Haus ist ein **Gebäude**, in dem Menschen leben oder arbeiten, das heisst ein **Wohngebäude** oder Geschäftsgebäude.*"

Wort\Schlüsselwörter	**Gebäude**	**Wohngebäude**
ein	1	1
Haus	0	0
ist	1	0
Gebäude	0	0
in	1	0
dem	1	0
Menschen	0	0
leben	0	0
oder	0	0
arbeiten	0	0
das	0	0
heisst	0	1
Wohngebäude	0	0
oder	0	1
Geschäftsgebäude	0	1

Tabelle 6.1.: Häufigkeitsmatrix

Da viele Wörter oft nur in der Nähe weniger Schlüsselwörter vorkommen, betragen die meisten Zellen 0. Diese Matrix wird auch als eine Sprase-Matrix bezeichnet. In einer Sparse-Matrix werden nur Elemente ungleich 0 gespeichert, um Speicherplatz zu sparen. Es wird dafür die Implementation des ALGLIB Projekts verwendet (Pro14).

0	0	0	0	0	**5**	0
0	**1**	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	**4**	0	0	0
0	0	0	0	0	0	0

Tabelle 6.2.: Sprase-Matrix Beispiel

Um RAM-Speicher zu sparen, wird diese Matrix in einer temporären Datei für die Phase der Berechnung abgelegt.

6.1.3. Berechnung der Vektoren

In der letzten Phase werden die Auftretenshäufigkeiten fortlaufend aus der temporären Datei gelesen. Mit diesen Auftretenshäufigkeiten werden nach der Formel 5.5 die Vektoren für die Wörter berechnet. Anschliessend werden die Vektoren gefiltert um irrelevante Informationen zu entfernen. Hierzu werden die grössten β-Werte behalten. Wieviele Werte dies in einem Vektor sind, wird nach der Formel 5.6 berechnet.

In der Formel 5.5 wird der Logarithmus eines Bruchs berechnet. Dieser Bruch wird mit Hilfe der Logorithmusregeln aufgesplittet (Wik14a):

$$log_2 \frac{f^k(w_i, c_j) \times m}{f^C(w_i) \times f^C(c_j)} = log_2(f^k(w_i, c_j)) + log_2(m) - log_2(f^C(w_i)) - log_2(f^C(c_j)) \tag{6.1}$$

Zusätzlich werden die Logarithmus-Werte in einer *Lookup-Table* zwischengespeichert und neu berechnet, falls das Ergebnis noch nie benötigt wurde. So kann die Laufzeit dieser Phase massiv verkürzt werden.

Diese Berechnungen sind mit Hilfe der C# *Library Parallel LINQ* implementiert und parallelisiert (Mic14a). Genauso wie die Daten fortlaufend aus der temporären Datei gelesen werden, werden die Ergebnisse dieser Berechnungen fortlaufend in eine Datei gespeichert. Um Speicherplatz zu sparen, wird für jede Zeile der Matrix nur gespeichert, wie viele Werte nicht 0 entsprechen und die Werte mit ihrem Spaltenindex. Für den Vektor $[4, 0, 0, 0, 0, 0, 0]$ würden drei Werte gespeichert:

1. Die Anzahl Werte die nicht 0 sind: 1
2. Der Index des Werts: 0
3. Der Wert selbst: 4

Da diese Datei keine temporäre ist, sondern die Basis für spätere Ähnlichkeitsberechnungen darstellt, muss sie möglichst klein sein. Deshalb wird sie zusätzlich mit der *Compression Library* von C# komprimiert (Mic14b).

6.2. Berechnung der Ähnlichkeit

Im zweiten Teil der Implementation geht es um die Berechnung von zwei Begriffen. Bevor die Ähnlichkeit berechnet werden kann, muss die Datei mit den Berechnungen geladen werden. Die Daten werden in einer Sparse-Matrix gespeichert. In dieser Matrix repräsentiert jede Zeile ein Wort und die Spalten die Schlüsselwörter. Um die Ähnlichkeit zweier Begriffe zu berechnen, werden die beiden Zeilen der Matrix benötigt, die den beiden Begriffen entsprechen.

Die Begriffe, die der Funktion übergeben werden, wie in Unterabschnitt 6.1.1 beschrieben, werden auf ihren Wortstamm zurückgeführt. Von beiden Wortstämmen wird die jeweilige Zeile in der Matrix gelesen. Diese beiden Zeilenvektoren bilden die Basis für die Ähnlichkeitsberechnungen.

Zur Berechnung der Ähnlichkeit kann zwischen zwei Verfahren ausgewählt werden. Auf der einen Seite kann die Kosinus-Ähnlichkeit, beschrieben im Unterabschnitt 5.2.1, berechnet werden, andererseits kann die Berechnung "*Second Order Co-Occurrence Pointwise Mutual Information*" angewendet werden, welche im Unterabschnitt 5.2.2 beschrieben wurde.

6.3. Parameterwahl

Die mathematischen Formeln haben einige Parameter, die Spielraum für Optimierungen bieten. Das sind einerseits die Fenstergrösse k und die Gewichtung δ, die bereits in den Unterabschnitten 5.1.2 und 6.1.2 thematisiert wurden. Diese haben Auswirkungen auf den ersten Teil der Implementation. Andererseits gibt es die Parameter γ und λ aus den Formeln in Abschnitt 5.2, die Auswirkungen auf den zweiten Teil der Funktion haben.

6.3.1. Fenstergrösse

Die Fenstergrösse k ist der einzige Parameter zu dem Hassan und Mihalcea keine Aussage machen (HM11). Deshalb wurde bei anderen Arbeiten und Verfahren, die Auftretenshäufigkeiten verwenden, betrachtet. Rapp verwendet in seinem Verfahren die Fenstergrösse 2 (Rap04), Kolb verwendet 3 (Kol08) und Islam et al. verwenden die Fenstergrösse 5 (II06).

Für alle drei Verfahren sind alle Wörter im Fenster relevant, im Gegensatz zum SSA-Verfahren, bei dem im Fenster nur die Schlüsselwörter relevant sind. In Anbetracht dessen, dass die Fenstergrösse nicht zu gross gewählt werden darf und das im SSA-Verfahren nicht alle Wörter innerhalb des Fensters relevant sind, wurde die grösste Fenstergrösse von den drei Vorschlägen gewählt: ± 5. Dies bedeutet, dass die Schlüsselwörter, die bis zu fünf Wörter vor dem aktuellen Begriff liegen, gezählt werden und in den fünf Wörtern danach.

6.3.2. Delta, Gamma, Lambda

Hassan und Mihalcea haben in ihrer Arbeit zwei Belegungen für die drei Parameter angegeben. Sie geben an, dass sie mit diesen Parametern die besten Korrelationen mit ihren Vergleichsdaten erreicht haben. Diese entsprechen den Werten $\delta = 0.3$, $\gamma = 0.125$, $\lambda = 1$ und $\delta = 0.4$, $\gamma = 0.05$, $\lambda = 0.01$ (HM11).

Obwohl diese für die englische Wikipedia und englische Referenzdaten optimiert sind, bilden sie den Ausgangspunkt dieser Arbeit. Da δ in der Phase der Vektorberechnungen, beschrieben in den Unterabschnitten 5.1.2 und 6.1.3, benötigt wird und diese Phase die längste Laufzeit von allen hat. Die Laufzeit dauert mehrere Tage. Deshalb können nicht viele Belegungen für diesen Parameter getestet werden.

7. Ergebnisse

7.1. Optimierungsbedarf

In diesem Kapitel werden die Resultate und die gewonnenen Erkentnisse betrachtet.

Der grösste Teil der Zeit, die in das Projekt investiert wurde, musste dafür eingesetzt werden, den Code des ersten Teils, also der Analyse der Wikipedia, zu optimieren.

In der ersten Version wurde der Algorithmus, der in Kapitel 5 beschrieben wird, implementiert. Wie sich herausstellte, dauerte ein Rechendurchlauf länger als die Projektdauer. Eine komplette Laufzeit würde mehrere Monate benötigtigen.

Deshalb wurde die Implementation parallelisiert, um alle Prozessorkerne optimal auslasten zu können. Bei dieser Optimierung musste zusätzlich sehr stark auf die Ressourcenauslastung des Zwischenspeichers geachtet werden. Viele ungewollte Abbrüche des Programmes konnten auf eine zu hohe Belsastung des Zwischenspeichers zurückgeführt werden.

7.2. Wörter und Schlüsselwörter

Obwohl die Parallelisierung die Implementation sehr viel schneller gemacht haben, hätte die Rechenzeit noch immer mehrere Wochen gedauert. Die Phase der Vektorberechnung, die in den Unterabschnitten 5.1.2 und 6.1.3 beschrieben ist, benötigt dabei mit Abstand die meiste Rechenzeit.

In dieser Phase wird die Matrix mit Auftretenshäufigkeit aller Wörter in der Verbindung mit den Schlüsselwörtern genommen und anhand der Formel 5.5 eine neue Matrix berechnet. Diese Matrix hat die Anzahl an Schlüsselwörtern als Spaltenanzahl und die Anzahl an Wörtern als Spaltenanzahl.

Anzahl Wörter:	7'830'713
Anzahl Schlüsselwörter:	8'696'267

Tabelle 7.1.: Anzahl Wörter und Schlüsselwörter

Dies bedeutet, dass die Funktion für jedes Wort knapp 8.7 Millionen Zahlen verwenden muss. Wenn von der Schätzung ausgegangen wird, dass für jede Zahl nur 0.01 Millisekunden benötigt werden, bräuchte der Rechner für jedes Wort immer noch 1.5 Minuten. Dies würde bei 7.8 Millionen Wörtern 22.5 Jahre Laufzeit bedeuten. Da diese Arbeit in einem Semester durchgeführt wird, ist eine solche Laufzeit nicht zielführend.

Um die Anzahl der Schlüsselwörter zu reduzieren, wird von der Annahme ausgegangen, dass diejenigen Schlüsselwörter, welche auf einen wirklich existierenden Wikipedia-Artikel verweisen, die wichtigsten sind. Wie Hassan und Mihalcea schreiben, sind bei der Wikipedia die Wörter verlinkt, die wichtig für das Verständnis des Textes sind (HM11). Diejenigen Wörter, die einen Wikipedia-Artikel haben, werden einer Person so wichtig gewesen sein, dass er oder sie einen Artikel erstellt hat. Wenn die Schlüsselwörter mit dieser Methode reduziert werden, bleiben noch **2'263'688** übrig. Somit konnte die Spaltenanzahl drastisch reduziert werden.

Um die Laufzeit weiter zu verkürzen, wurde beschlossen die Vektoren für nicht alle Wörter zu berechnen. Einerseits nimmt die Genauigkeit der Berechnungen mit der Auftretenshäufigkeit ab. Andererseits werden Wörter, die nicht so häufig auftreten auch seltener benötigt. Um eine geeignete Anzahl Wörter bestimmen zu können, wurde die Auftretenshäufigkeit der häufigsten Wörter untersucht:

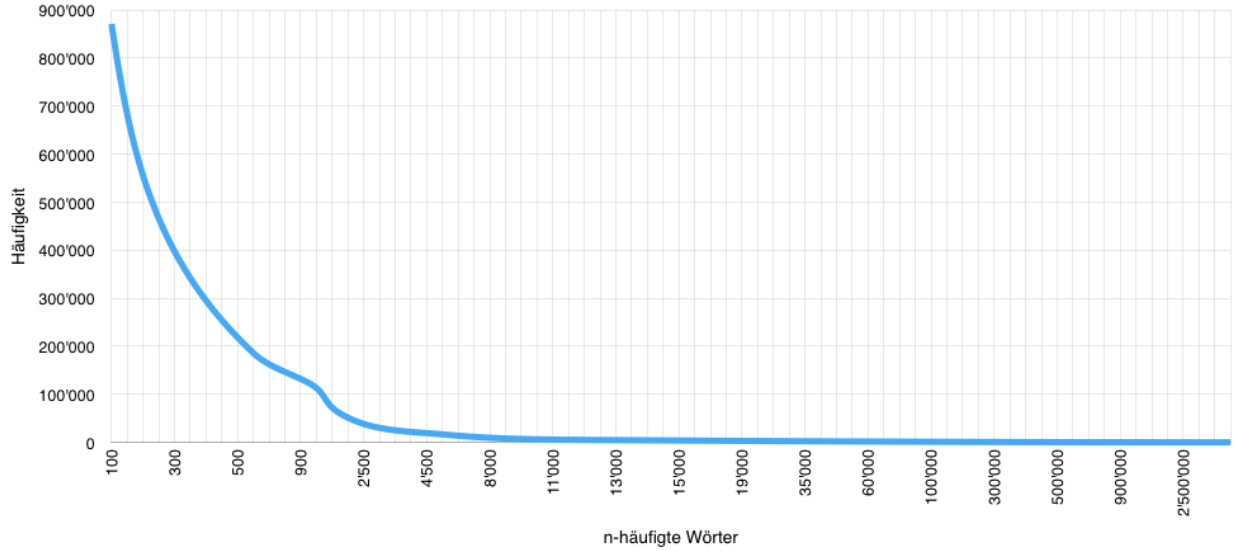

Abbildung 7.1.: Häufigkeitsverteilung Wörter

Wie in der obigen Abbildung erkennbar ist, kommen die häufigsten 300 Wörter sehr häufig vor, bei einer stark sinkenden Kurve. Nach den 100'000 häufigsten Wörtern ist die Kurve bereits sehr abgeflacht und die folgenden Wörter kommen in der deutschsprachigen Wikipedia keine 300 Mal mehr vor. Um jedoch einige Tests mit vielen Wörtern machen zu können, wurden die häufigsten 300'000 Wörter für die Berechnung genommen, da die Laufzeit mit etwa 2.5 Tagen noch vertretbar ist.

7.3. Korrelationen

In der zur Verfügung stehenden Zeit konnten die Wikipedia zwei Mal analysiert werden. Einmal mit dem Parameter $\delta = 0.3$ und einmal mit $\delta = 0.4$. Für beide wurde die Fenstergrösse 5 verwendet.

7.3.1. Standardkonfiguration

Wie in Abschnitt 6.3 beschrieben, wurden die Werte der Parameter so gesetzt, mit denen Hassan und Mihalcea die besten Ergebnisse bekommen haben (HM11). Mit diesen Parametern wurden die Korrelationen mit vier deutsprachigen Testdateien mit Referenzdaten der Technischen Universität Darmstadt berechnet (Dar14). Dafür wurde Bravais-Pearson's Korrelationskoeffizeint, Spearman-Pearson's Rangkorrelationskoeffizient und die harmonische Mitte der beiden Koeffizienten berechnet. Die genaue Berechnung ist in Kapitel 4 beschrieben.

Die Ergebnisse der Implementation der beiden Berechnungsvarianten $\mathbf{SSA}_{\cos}$ und $\mathbf{SSA}_{\text{soc}}$ – beschrieben in Abschnitt 5.2 – verglichen mit Zufallswerten zeigen die Tabellen 7.2 und 7.3.

Die Korrelationen von 0, vor allem bei dem SOC Verfahren, kommen daher, dass der berechnete Wert in allen Fällen grösser ist als das λ. Dieser Parameter ist dazu da, die Werte zwischen 0 und 1 zu

verteilen. Wie in der Gleichung 5.9 erkennbar ist, wird der Ähnlichkeits-Wert somit immer 1, wenn der Wert grösser als λ ist.

Testdaten	**SSA$_{soc}$**			**SSA$_{cos}$**			**Zufall**		
	r	ρ	μ	r	ρ	μ	r	ρ	μ
gur65.csv	0	0	0	0.1089	0.0719	0.0866	0.1006	0.0969	0.0964
gur350.csv	0	0	0	0.0730	0.0580	0.0646	0.0461	0.0458	0.0455
zg222.csv	0	0	0	0.0096	0.0573	0.0164	0.0523	0.0530	0.0510
umfrage.csv	0	0	0	0.1436	0.2154	0.1723	0.0950	0.0976	0.0939
Durchschnitt	0	0	0	0.0837	0.1006	**0.0850**	0.0735	0.0733	**0.0717**

Tabelle 7.2.: Korrelation der Kombination: $\delta = 0.3$, $\gamma = 0.125$, $\lambda = 1$

Testdaten	**SSA$_{soc}$**			**SSA$_{cos}$**			**Zufall**		
	r	ρ	μ	r	ρ	μ	r	ρ	μ
gur65.csv	0	0	0	0.0895	0.0020	0.0040	0.1080	0.1101	0.1072
gur350.csv	0	0	0	0.0661	0.0345	0.0453	0.0431	0.0417	0.0417
zg222.csv	0	0	0	0.1037	0.0313	0.0481	0.0569	0.0542	0.0541
umfrage.csv	0	0	0	0	0	0	0.0987	0.0984	0.0971
Durchschnitt	0	0	0	0.0864	0.0226	**0.0325**	0.0767	0.0761	**0.0750**

Tabelle 7.3.: Korrelation der Kombination: $\delta = 0.4$, $\gamma = 0.05$, $\lambda = 0.01$

Wie aus der Tabelle 7.2 ersichtlich wird, ergibt die Berechnung der SSA$_{cos}$ Verfahren mit den Parameter $\delta = 0.3$, $\gamma = 0.125$, $\lambda = 1$ die höchste Korrelation. Trotzdem ist er nur leicht besser als die Zufallswerte. Die Ergebnisse in der Tabelle 7.3 sind sogar schlechter als die Zufallswerte.

7.3.2. Lambda-Modifikation

Um zu verhindern, dass der λ-Parameter die Ähnlichkeitswerte verschwinden lässt, wird λ auf 10 erhöht und der Test wiederholt. Dies führt dazu, dass sowohl für die Beispiele aus der Umfrage als auch für das SSA$_{cos}$-Verfahren sinvollere Werte geliefert werden.

Wie in den Tabellen 7.4 und 7.5 ersichtlich wird, sind die Korrelationswerte der Umfrage höher als die Referenzdaten. In der Datei “gur65.csv” sind Werte, die die semantische Ähnlichkeit in einer diskreten Skala zwischen 0 und 4 bewerten. Die Referenzdaten der Dateien “gur350.csv” und “zg222.csv” bewerten die semantische Bezogenheit in einer diskreten Skala zwischen 0 und 4. In der Umfrage wurde zwischen 1 und 10 bewertet, wobei die Befragten den Unterschied von semantischer Ähnlichkeit und Bezogenheit nicht kannten (Dar14).

Zudem sind die Werte des SSA$_{soc}$-Verfahrens mit beiden Parameter-Kombinationen schlechter, als die des SSA$_{cos}$-Verfahrens. Mit 12.26% Korrelation findet sich in der Tabelle 7.5 den bisher besten Wert. Für die Beispiele in der Umfrage erreicht das SSA$_{cos}$-Verfahren sogar eine Korrelation von 28.55% mit den Bewertungen der Studenten.

7.3.3. Gamma-Optimierung

Um zu testen, wie stark die Werte noch optimierbar sind, wurden die beiden generierten Datensätze von $\delta = 0.3$ und $\delta = 0.4$ mit verschiedenen γ-Werten getestet. Da die von Hassan und Mihalcea verwendeten γ-Werte 0.05 und 1 betragen, wurde in Schritten von 0.05 getestet von 0 bis 2 (HM11).

Testdaten	**SSA$_{soc}$**			**SSA$_{cos}$**			**Zufall**		
	r	ρ	μ	r	ρ	μ	r	ρ	μ
gur65.csv	0.0538	0.0146	0.0230	0.1111	0.0720	0.0874	0.1086	0.1125	0.1085
gur350.csv	0.0177	0.0323	0.0229	0.0753	0.0715	0.0733	0.0453	0.0450	0.0446
zg222.csv	0.0529	0.0899	0.0666	0.1346	0.0108	0.0200	0.0594	0.0601	0.0586
umfrage.csv	0.0797	0.1148	0.0941	0.1355	0.2154	0.1664	0.0898	0.0923	0.0880
Durchschnitt	0.0510	0.0629	**0.0517**	0.1141	0.0924	**0.0868**	0.0758	0.0775	**0.0749**

Tabelle 7.4.: Korrelation der Kombination: $\delta = 0.3$, $\gamma = 0.125$, $\lambda = 10$

Testdaten	**SSA$_{soc}$**			**SSA$_{cos}$**			**Zufall**		
	r	ρ	μ	r	ρ	μ	r	ρ	μ
gur65.csv	0.0810	0.0338	0.0477	0.0773	0.0722	0.0747	0.0894	0.0874	0.0853
gur350.csv	0.0439	0.0323	0.0372	0.0797	0.0985	0.0881	0.0487	0.0490	0.0483
zg222.csv	0.1081	0.0177	0.0304	0.1316	0.0251	0.0421	0.0528	0.0498	0.0497
umfrage.csv	0.1947	0.2007	0.1977	0.2306	0.3745	0.2855	0.1031	0.1056	0.1020
Durchschnitt	0.1069	0.0711	**0.0782**	0.1298	0.1426	**0.1226**	0.0735	0.0729	**0.0713**

Tabelle 7.5.: Korrelation der Kombination: $\delta = 0.4$, $\gamma = 0.05$, $\lambda = 10$

Die Quelldaten finden sich für Abbildung 7.2 in Anhang B und für Abbildung 7.3 in Anhang C. Die Abbildungen zeigen auf der Y-Achse die durchschnittlichen Korrelationen über alle 4 Referenzdatensätze und auf der X-Achse die verschiedenen γ-Werte.

Bei fast allen Kombinationen, ist das SSA$_{cos}$-Verfahren das bessere. Wie in Abbildung 7.2 erkennbar ist, ist für $\delta = 0.3$ der beste γ-Wert $\gamma = 1.10$. Mit dieser Kombination kann eine Korrelation von 13.88% erreicht werden.

Einen noch besseren Korrelationswert kann aber mit der Kombination $\delta = 0.4$ und $\gamma = 1.70$ erreicht werden. Wie der Abbildung 7.3 und Anhang C entnommen werden kann,wird eine durchschnittliche Korrelation von 15.93% erreicht. Bei den Werten der Umfrage wurde sogar eine Korrelation von 45.55% erreicht.

Um die Werte gut zwischen 0 und 1 zu verteilen, stellte sich einen λ-Wert von 0.2 als gute Wahl heraus. Die höchsten Werte von unterschiedlichen Wörtern reichen bis knapp an 0.2 heran.

7.4. Vergleich mit anderen Implementationen

In der Tabelle 7.6 sind die erreichten Werte, im Vergleich mit beiden Varianten von DISCO (KG14) auf dieselben Testdatensätze. Die beiden unteren Zeilen sind die Werte von Hassan und Mihalcea in ihrer Arbeit (HM11). Diese können allerdings nur mit Vorsicht verglichen werden. Diese stützen sich einerseits auf andere Referenzdaten und beziehen sich andererseits auf die englischsprachige Wikipedia.

Die beiden Varianten von DISCO erreichen jeweils etwa 50%, während die Implementation von dieser Arbeit nur 15.93% erreicht. Im Vergleich dazu, belaufen sich die Werte von Hassan und Mihalcea auf 74.35% und 73.9% (HM11).

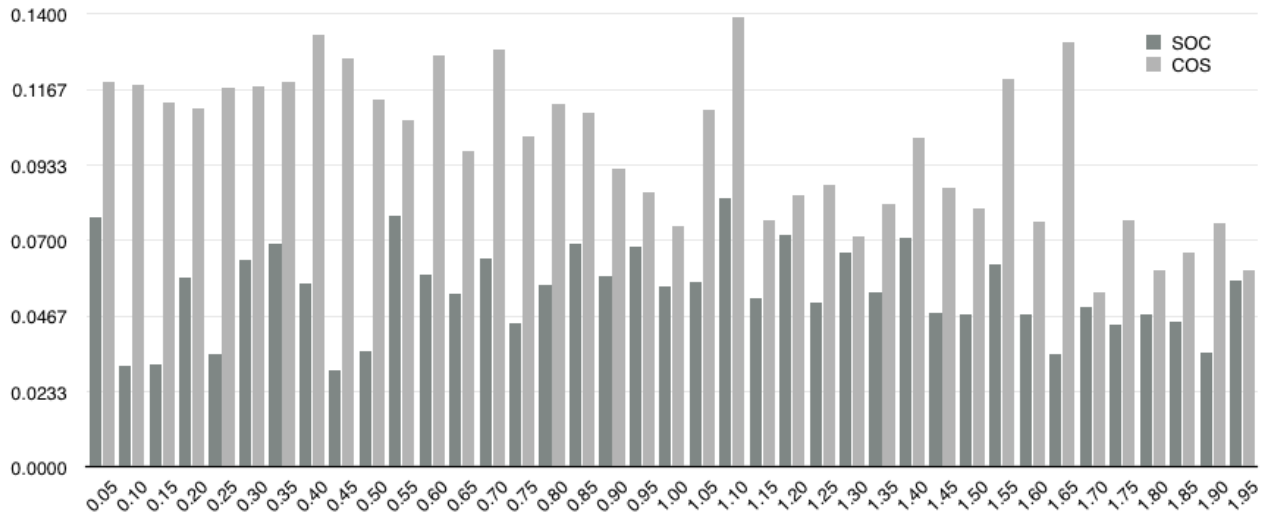

Abbildung 7.2.: Korrelationen mit den Parameter $\delta = 0.3$ und $\lambda = 10$ und verschiedenen γ

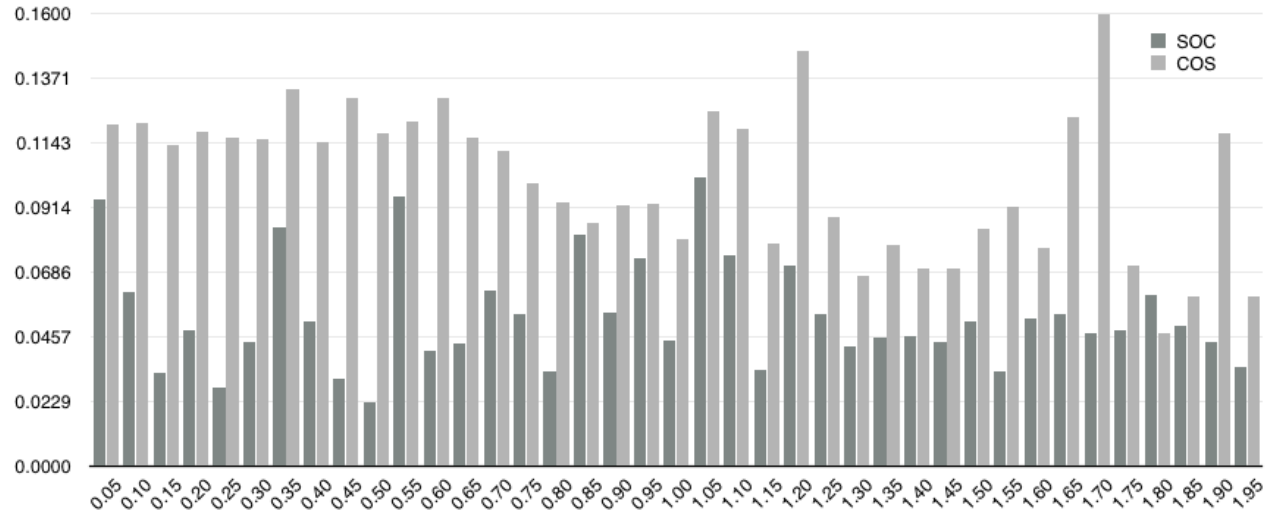

Abbildung 7.3.: Korrelationen mit den Parameter $\delta = 0.4$ und $\lambda = 10$ und verschiedenen γ

	Korrelation		
	r	ρ	μ
SSA	0.1833	0.1565	0.1593
Disco1	0.4719	0.5581	0.5077
Disco2	0.4674	0.5472	0.4986
SSA_{soc} (HM11)	0.7527	0.7347	0.7435
SSA_{cos} (HM11)	0.7447	0.7332	0.7390

Tabelle 7.6.: Ergebnisvergleich mit anderen Implementationen

8. Fazit

Mit dieser Arbeit konnte eine Funktion entwickelt werden, mit der die Ähnlichkeit von zwei Wörtern in einer numerischen Zahl ausgedrückt werden kann. Diese Zahl korreliert zu 15.93% mit den getesteten menschlichen Einschätzungen. Zufällige Werte, so wurde festgestellt, korrelieren mit den selbigen zu 7.5%. Das Potential konnte allerdings nicht ausgeschöpft werden, wenn andere Implementationen mit den gleichen menschlichen Einschätzungen zu etwa 50% korrelieren und die Erfinder des Verfahrens mit englischen Wörtern auf eine Korrelationen von 74% erreichen.

Aus zeitlichen Gründen konnte nicht alle Möglichkeiten ausgeschöpft werden. Die Laufzeit der Analyse musste stark reduziert werden. Anstatt mit den gefunden ca. acht Millionen Schlüsselwörtern, wurde mit gut zwei Millionen gearbeitet. Es kann davon ausgegangen werden, dass bessere Werte erreicht werden können, wenn alle Schlüsselwörter verwendet werden.

Weiter besteht noch Potential in der Optimierung der Parameter. Die Ergebnisse haben gezeigt, das eine Veränderung des γ-Parameters starke Unterschiede in der Qualität ergeben. Der δ-Parameter konnte allerdings nur mit zwei Belegungen getestet werden. Es wurden die Belegungen genommen, mit denen Hassan und Mihalcea die besten Resultate erreicht haben. Der Parameter δ hängt, laut ihnen, jedoch von der Grösse der Datenbasis ab (HM11). Da davon ausgegangen werden kann, dass sie mit der englischen Wikipedia gearbeitet haben und diese Arbeit mit der deutschen Wikipedia arbeitet, könnten andere Werte durchaus bessere Resultate liefern.

Ein weiterer Ansatz um die Resultate zu verbessern wäre die Vergrösserung der Fenstergrösse. Wie in Unterabschnitt 6.3.1 erörtert, wurde diese aufgrund anderer Implementationen festgelegt, weil Hassan und Mihalcea dazu keine Aussage treffen. Diese andernen Implementationen berücksichtigen jedoch jedes Wort innerhalb des Fensters. In dem in dieser Arbeit beschriebenen Verfahren, spielen jedoch nur die Schlüsselwörter in dem Fenster eine Rolle. Es könnte daher durchaus eine Verbesserung geben, wenn das Fenster vergrössert wird.

Abschliessend lässt sich sagen, dass diese Methode die Ähnlichkeit zwischen zwei Wörtern zu ermitteln durchaus Potential hat. Es benötigt jedoch viel Zeit um die Varianten und Verfahren zu testen. Wie einige wissenschaftliche Publikationen zeigen, kann die Ähnlichkeit mittlerweile jedoch gut ermittelt werden.

9. Literaturverzeichnis

[BH06] Budanitsky, Alexander ; Hirst, Graeme: Evaluating wordnet-based measures of lexical semantic relatedness. In: Computational Linguistics 32 (2006), Nr. 1, S. 13–47

[CH90] Church, Kenneth W. ; Hanks, Patrick: Word association norms, mutual information, and lexicography. In: Computational linguistics 16 (1990), Nr. 1, S. 22–29

[Dar14] Darmstadt, Technische U.: German Relatedness Datasets. `http://www.ukp.tu-darmstadt.de/data/semantic-relatedness/german-relatedness-datasets/`, März 2014

[DW03] Dorow, Beate ; Widdows, Dominic: Discovering corpus-specific word senses. In: Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 2 Association for Computational Linguistics, 2003, S. 79–82

[Eas14] EasyWay: Word stemming for German on .NET Framework. `http://www.codeproject.com/Articles/3701/Word-stemming-for-German-on-NET-Framework`, Juli 2014

[GCY92] Gale, William A. ; Church, Kenneth W. ; Yarowsky, David: One sense per discourse. In: Proceedings of the workshop on Speech and Natural Language Association for Computational Linguistics, 1992, S. 233–237

[GM07] Gabrilovich, Evgeniy ; Markovitch, Shaul: Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. In: IJCAI Bd. 7, 2007, S. 1606–1611

[HM11] Hassan, Samer ; Mihalcea, Rada: Semantic Relatedness Using Salient Semantic Analysis. In: AAAI, 2011

[II06] Islam, Aminul ; Inkpen, Diana: Second order co-occurrence PMI for determining the semantic similarity of words. In: Proceedings of the International Conference on Language Resources and Evaluation (LREC 2006), 2006, S. 1033–1038

[KG14] Kolb, Peter ; GbR, Prochazkova: DISCO compute semantic similarity beetween words. `http://www.linguatools.de/disco/disco.html`, März 2014

[Kol08] Kolb, Peter: Disco: A multilingual database of distributionally similar words, 2008

[Kol09] Kolb, Peter: Experiments on the difference between semantic similarity and relatedness. (2009)

[LD97] Landauer, Thomas K. ; Dumais, Susan T.: A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. In: Psychological review 104 (1997), Nr. 2, S. 211

[MC07] Mihalcea, Rada ; Csomai, Andras: Wikify!: linking documents to encyclopedic knowledge. In: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management ACM, 2007, S. 233–242

[Mic14a] Microsoft: Parallel LINQ (PLINQ). `http://msdn.microsoft.com/library/dd460688(v=vs.110).aspx`, Juli 2014

[Mic14b] Microsoft: System.IO.Compression Namespace. http://msdn.microsoft.com/library/System.IO.Compression(v=vs.110).aspx, Juli 2014

[Mic14c] Microsoft: Task Parallel Library (TPL). http://msdn.microsoft.com/library/dd460717(v=vs.110).aspx, Juli 2014

[Por01] Porter, Martin: German stemming algorithm. http://snowball.tartarus.org, Juli 2001

[Pro14] Project, ALGLIB: ALGLIB - numerical analysis library. http://www.alglib.net/aboutus.php, Juli 2014

[Rap04] Rapp, Reinhard: A Freely Available Automatically Generated Thesaurus of Related Words. In: LREC, 2004

[Res95] Resnik, Philip: Using information content to evaluate semantic similarity in a taxonomy. In: arXiv preprint cmp-lg/9511007 (1995)

[Tur01] Turney, Peter: Mining the web for synonyms: PMI-IR versus LSA on TOEFL. (2001)

[Wik14a] Wikibooks: Beweisarchiv: Arithmetik: Erweiterte Rechenarten: Logarithmus: Logarithmengesetze — Wikibooks, Die freie Bibliothek. http://de.wikibooks.org/w/index.php?title=Beweisarchiv:_Arithmetik:_Erweiterte_Rechenarten:_Logarithmus:_Logarithmengesetze&oldid=673454, Juli 2014

[Wik14b] Wikibooks: Mathematik: Statistik: Korrelationsanalyse — Wikibooks, Die freie Bibliothek. http://de.wikibooks.org/w/index.php?title=Mathematik:_Statistik:_Korrelationsanalyse&oldid=651914, Juli 2014

[Wik14c] Wikimedia: Wikimedia Downloads. http://dumps.wikimedia.org/backup-index.html, Juli 2014

[Wik14d] Wikipedia: Latent semantic analysis — Wikipedia, The Free Encyclopedia. http://en.wikipedia.org/w/index.php?title=Latent_semantic_analysis&oldid=609640347, Juni 2014

10. Abbildungsverzeichnis

11. Tabellenverzeichnis

12. Ehrlichkeitserklärung

Hiermit erkläre ich die vorliegende Bachelorthesis selbständig, ohne Hilfe Dritter und nur unter Benutzung der angegebenen Quellen verfasst zu haben.

Jan Fässler

Ort, Datum Unterschrift

A. Umfrage

			Testpersonen									
Wort1	**Wort2**	**Ø**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Offerte	Angebot	**8.2**	9	8	10	8	10	9	10	5	7	6
Aufwand	Leistung	**5.5**	6	6	5	7	10	5	5	5	2	4
Angebot	Leistung	**4.4**	4	4	8	7	4	2	1	7	3	4
Mitarbeiter	Angestellter	**7.6**	5	10	7	7	8	10	8	9	7	5
Arbeiter	Angestellter	**6.6**	7	10	3	7	5	9	4	10	4	7
Verzeichnis	Offerte	**1.7**	2	1	1	1	1	2	3	1	3	2
Energie	Leistung	**3.7**	4	8	1	5	3	5	4	1	4	2
Zustimmung	Gültigkeit	**3.2**	3	7	1	5	4	4	1	2	3	2
Kostendach	Spesen	**2.8**	2	6	3	1	1	4	3	1	4	3
Lehrperson	Schüler	**4.4**	3	8	6	8	1	4	5	2	3	4
Stadt	Dorf	**7.2**	5	10	8	8	5	6	8	10	6	6
Gemeinde	Stadt	**7.8**	7	10	6	9	8	7	8	10	6	7
Optimierung	Verbesserung	**8.3**	9	10	8	8	9	9	9	8	5	8
Uhr	Stopuhr	**7.8**	5	10	7	8	7	8	7	10	8	8
Modus	Variante	**7.6**	10	10	10	7	2	9	6	10	8	4
Status	Sprache	**2**	1	1	4	1	2	1	3	1	4	2
Anzeige	Start	**1.7**	2	1	2	1	1	1	4	1	3	1
Gerät	Projekt	**1.9**	3	1	1	1	1	2	3	1	3	3
Eigenschaft	Projektzeit	**1.7**	4	1	1	1	1	1	2	1	2	3
Resultat	Verbesserung	**2.4**	3	4	1	1	1	1	2	5	4	2
Sprache	Mitarbeiter	**1.3**	3	2	1	1	1	1	1	1	1	1
Offerte	Spesen	**2.4**	4	5	3	1	1	1	4	1	2	2
Server	Service	**5.5**	7	8	4	7	4	1	7	7	6	4
Optimierung	Resultat	**3.7**	4	7	0	6	1	1	5	5	5	3
Dorf	Gemeinde	**7.4**	7	10	7	7	2	9	8	10	7	7
Funktion	Freund	**1.8**	4	1	1	1	1	1	4	2	1	2
Lehrperson	Kontakt	**2.8**	5	1	1	5	1	1	5	6	1	2
Variante	Status	**1.5**	1	1	1	1	1	1	4	1	1	3
Stadt	Service	**1.7**	4	1	1	1	1	1	3	2	1	2
Verbesserung	Eigenschaft	**2**	4	1	2	1	1	1	4	3	1	2
Mitarbeiter	Stadt	**2**	5	1	1	1	1	1	2	6	1	1
Person	Spesen	**2.7**	5	4	1	4	1	1	5	4	1	1
Schüler	Kontakt	**2.7**	5	1	1	6	1	1	5	4	1	2
Android	iPhone	**5.9**	7	9	8	9	1	4	7	1	7	6
WindowsPhone	iOS	**6.8**	8	9	8	8	4	6	7	1	7	10
Facebook	Twitter	**7.4**	8	10	8	9	7	6	7	5	6	8
iPhone	iOS	**7.4**	7	10	9	10	9	4	7	10	3	5
Facebook	iPhone	**3.7**	5	5	1	6	1	2	8	1	2	6
Suche	Kontakt	**2.4**	4	2	1	2	1	1	5	4	1	3
Netzwerk	Angestellter	**2.5**	3	1	1	1	1	1	6	8	1	2
Applikation	Android	**5.4**	8	7	3	7	1	3	8	10	4	3
Applikation	Netzwerk	**2.9**	6	1	1	6	1	1	6	3	1	3
Basis	Funktion	**1.7**	2	1	1	1	1	1	5	1	2	2
Produkt	Service	**5.5**	6	5	4	6	5	6	7	5	5	6
Version	Variante	**5.2**	7	8	2	2	5	7	5	6	4	6
Massnahme	Optimierung	**4.8**	6	6	1	6	3	4	6	5	4	7
Rolle	Person	**5**	6	5	2	8	1	1	7	10	6	4
Mittarbeiter	Rolle	**5.1**	6	5	3	6	1	4	7	10	6	3
Angestellter	Rolle	**5.1**	6	5	2	6	1	4	7	10	6	4
Verkehr	Auto	**6.9**	8	10	8	8	3	3	8	10	5	6
Fahrad	Auto	**6.2**	8	10	5	8	1	5	8	4	5	8
Schulleiter	Lehrperson	**6.1**	7	8	3	6	5	6	7	6	6	7
Stärkung	Optimierung	**5.1**	6	8	1	7	4	7	4	4	3	7
Gerät	Auto	**5.2**	7	4	1	4	8	4	4	10	5	5
Aufruf	Schrei	**5.7**	6	9	8	2	6	3	5	10	4	4
Aufgabe	Angabe	**2.1**	3	1	1	1	1	2	2	3	3	4
Absender	Person	**4.7**	7	2	1	5	1	6	7	10	3	5
Kündigung	Anstellung	**6.2**	6	9	7	7	1	4	7	10	6	5
Haftung	Angebot	**2.7**	3	2	1	1	1	1	2	10	3	3
Haftung	Anhang	**1.7**	3	1	1	1	1	2	2	1	2	3
Umfang	Grösse	**5.5**	7	8	8	7	1	4	7	4	4	5

B. Gamma Test mit $\delta = 0.3$ und $\lambda = 10$

γ	umfrage	gur65	zg222	gur350	SOC	umfrage	gur65	zg222	gur350	COS
0.05	0.1736	0.0501	0.0329	0.0521	**0.0772**	0.2729	0.0850	0.0424	0.0755	**0.1190**
0.10	0.0467	0.0409	0.0180	0.0191	**0.0312**	0.2703	0.0893	0.0330	0.0793	**0.1180**
0.15	0.0320	0.0375	0.0425	0.0142	**0.0315**	0.2701	0.0896	0.0378	0.0527	**0.1126**
0.20	0.0435	0.0199	0.0624	0.1086	**0.0586**	0.2703	0.0717	0.0306	0.0700	**0.1107**
0.25	0.0102	0.0290	0.0690	0.0324	**0.0352**	0.2659	0.0901	0.0437	0.0695	**0.1173**
0.30	0.0254	0.0179	0.0706	0.1415	**0.0638**	0.2703	0.0960	0.0274	0.0768	**0.1176**
0.35	0.0260	0.0708	0.0285	0.1511	**0.0691**	0.2605	0.0996	0.0408	0.0743	**0.1188**
0.40	0.0371	0.0404	0.0574	0.0912	**0.0565**	0.2577	0.0826	0.0849	0.1090	**0.1335**
0.45	0.0464	0.0051	0.0510	0.0174	**0.0300**	0.2502	0.0872	0.0973	0.0704	**0.1263**
0.50	0.0547	0.0345	0.0313	0.0221	**0.0356**	0.2463	0.0860	0.0486	0.0727	**0.1134**
0.55	0.1308	0.1211	0.0496	0.0082	**0.0774**	0.1504	0.0856	0.1224	0.0698	**0.1071**
0.60	0.0661	0.0412	0.0534	0.0779	**0.0597**	0.2299	0.0828	0.1249	0.0717	**0.1273**
0.65	0.0710	0.0498	0.0745	0.0198	**0.0538**	0.2208	0.0740	0.0366	0.0585	**0.0975**
0.70	0.0738	0.0403	0.0739	0.0697	**0.0644**	0.3426	0.0818	0.0201	0.0719	**0.1291**
0.75	0.0776	0.0283	0.0644	0.0072	**0.0444**	0.2048	0.0887	0.0393	0.0758	**0.1021**
0.80	0.0819	0.0402	0.0719	0.0319	**0.0565**	0.1979	0.1457	0.0305	0.0749	**0.1122**
0.85	0.0512	0.0303	0.0713	0.1237	**0.0691**	0.2695	0.0884	0.0038	0.0762	**0.1095**
0.90	0.1126	0.0296	0.0697	0.0243	**0.0591**	0.1928	0.0952	0.0278	0.0527	**0.0921**
0.95	0.0850	0.0225	0.0681	0.0972	**0.0682**	0.1863	0.0665	0.0111	0.0765	**0.0851**
1.00	0.0888	0.0495	0.0536	0.0315	**0.0559**	0.1751	0.0173	0.0256	0.0806	**0.0746**
1.05	0.1115	0.0072	0.0650	0.0460	**0.0574**	0.2628	0.0678	0.0310	0.0802	**0.1105**
1.10	0.1003	0.0159	0.0630	0.1530	**0.0831**	0.3704	0.0605	0.0411	0.0834	**0.1388**
1.15	0.1045	0.0770	0.0093	0.0188	**0.0524**	0.1530	0.0634	0.0190	0.0704	**0.0764**
1.20	0.1066	0.0009	0.0592	0.1201	**0.0717**	0.1354	0.0641	0.0476	0.0883	**0.0838**
1.25	0.1103	0.0208	0.0571	0.0149	**0.0508**	0.1366	0.0723	0.0648	0.0760	**0.0874**
1.30	0.1126	0.0846	0.0549	0.0126	**0.0662**	0.1247	0.0826	0.0015	0.0763	**0.0713**
1.35	0.1142	0.0399	0.0526	0.0102	**0.0542**	0.0994	0.0681	0.0807	0.0772	**0.0813**
1.40	0.2207	0.0128	0.0420	0.0077	**0.0708**	0.2504	0.0730	0.0068	0.0771	**0.1018**
1.45	0.1152	0.0220	0.0483	0.0050	**0.0476**	0.0718	0.1228	0.0729	0.0783	**0.0864**
1.50	0.1166	0.0249	0.0460	0.0022	**0.0474**	0.0598	0.0834	0.0988	0.0785	**0.0801**
1.55	0.1387	0.0730	0.0390	0.0007	**0.0628**	0.3159	0.0643	0.0200	0.0794	**0.1199**
1.60	0.1261	0.0172	0.0418	0.0035	**0.0472**	0.0598	0.0517	0.1109	0.0806	**0.0757**
1.65	0.0754	0.0193	0.0396	0.0060	**0.0351**	0.2628	0.0708	0.1100	0.0819	**0.1314**
1.70	0.1282	0.0229	0.0375	0.0084	**0.0493**	0.0271	0.0723	0.0343	0.0825	**0.0540**
1.75	0.1072	0.0231	0.0356	0.0106	**0.0441**	0.0272	0.0749	0.1182	0.0844	**0.0762**
1.80	0.1124	0.0275	0.0356	0.0127	**0.0471**	0.0174	0.0909	0.0488	0.0859	**0.0608**
1.85	0.1099	0.0233	0.0315	0.0147	**0.0449**	0.0176	0.0692	0.0912	0.0874	**0.0663**
1.90	0.0724	0.0236	0.0296	0.0165	**0.0355**	0.0217	0.0663	0.1251	0.0891	**0.0755**
1.95	0.1283	0.0245	0.0591	0.0182	**0.0575**	0.0041	0.0828	0.0648	0.0907	**0.0606**

C. Gamma Test mit $\delta = 0.4$ und $\lambda = 10$

γ	umfrage	gur65	zg222	gur350	SOC	umfrage	gur65	zg222	gur350	COS
0.05	0.1977	0.0477	0.0797	0.0511	**0.0940**	0.2855	0.0736	0.0392	0.0830	**0.1203**
0.10	0.0980	0.0472	0.0764	0.0241	**0.0614**	0.2847	0.0757	0.0375	0.0854	**0.1208**
0.15	0.0419	0.0494	0.0299	0.0097	**0.0327**	0.2825	0.0799	0.0357	0.0555	**0.1134**
0.20	0.0524	0.0025	0.0477	0.0897	**0.0481**	0.2847	0.0797	0.0245	0.0838	**0.1182**
0.25	0.0302	0.0545	0.0192	0.0075	**0.0278**	0.2800	0.0741	0.0490	0.0606	**0.1159**
0.30	0.0392	0.0398	0.0623	0.0341	**0.0438**	0.2847	0.0884	0.0069	0.0821	**0.1155**
0.35	0.0242	0.1074	0.0513	0.1534	**0.0841**	0.2934	0.0941	0.0563	0.0889	**0.1332**
0.40	0.0151	0.0687	0.0174	0.1017	**0.0507**	0.2668	0.0874	0.0628	0.0402	**0.1143**
0.45	0.0018	0.0505	0.0570	0.0127	**0.0305**	0.2579	0.0822	0.1040	0.0748	**0.1297**
0.50	0.0115	0.0532	0.0223	0.0035	**0.0226**	0.2520	0.0755	0.0665	0.0763	**0.1176**
0.55	0.2562	0.0642	0.0331	0.0274	**0.0952**	0.2102	0.0822	0.1201	0.0728	**0.1213**
0.60	0.0299	0.0215	0.0354	0.0762	**0.0408**	0.2378	0.0792	0.1288	0.0729	**0.1297**
0.65	0.0375	0.0829	0.0388	0.0132	**0.0431**	0.2284	0.0785	0.0921	0.0641	**0.1158**
0.70	0.0441	0.0715	0.0682	0.0640	**0.0619**	0.2697	0.0778	0.0245	0.0726	**0.1112**
0.75	0.0502	0.0708	0.0751	0.0191	**0.0538**	0.2115	0.0767	0.0346	0.0765	**0.0998**
0.80	0.0553	0.0047	0.0686	0.0051	**0.0335**	0.2046	0.0840	0.0094	0.0745	**0.0931**
0.85	0.0961	0.0444	0.0684	0.1171	**0.0815**	0.1794	0.0863	0.0017	0.0767	**0.0860**
0.90	0.0640	0.0598	0.0688	0.0245	**0.0543**	0.1872	0.0804	0.0364	0.0641	**0.0920**
0.95	0.0753	0.0529	0.0673	0.0969	**0.0731**	0.1872	0.0950	0.0107	0.0772	**0.0925**
1.00	0.0840	0.0208	0.0513	0.0204	**0.0441**	0.1752	0.0388	0.0263	0.0806	**0.0802**
1.05	0.2377	0.0398	0.0657	0.0650	**0.1021**	0.2713	0.1377	0.0103	0.0815	**0.1252**
1.10	0.0673	0.0405	0.0651	0.1250	**0.0745**	0.2800	0.0712	0.0410	0.0837	**0.1190**
1.15	0.0683	0.0055	0.0391	0.0227	**0.0339**	0.1466	0.1086	0.0211	0.0380	**0.0786**
1.20	0.0705	0.0432	0.0631	0.1068	**0.0709**	0.3282	0.1025	0.0666	0.0879	**0.1463**
1.25	0.0886	0.0433	0.0616	0.0202	**0.0534**	0.1225	0.0877	0.0645	0.0774	**0.0880**
1.30	0.0893	0.0001	0.0605	0.0187	**0.0421**	0.0994	0.0822	0.0097	0.0774	**0.0672**
1.35	0.0901	0.0153	0.0589	0.0170	**0.0453**	0.0717	0.0800	0.0831	0.0779	**0.0782**
1.40	0.0924	0.0292	0.0460	0.0151	**0.0457**	0.1185	0.0768	0.0049	0.0782	**0.0696**
1.45	0.0948	0.0115	0.0563	0.0132	**0.0439**	0.0875	0.0618	0.0512	0.0785	**0.0697**
1.50	0.0981	0.0399	0.0548	0.0112	**0.0510**	0.0789	0.0760	0.1014	0.0790	**0.0838**
1.55	0.0079	0.0895	0.0258	0.0091	**0.0331**	0.1862	0.0792	0.0200	0.0802	**0.0914**
1.60	0.1015	0.0465	0.0522	0.0069	**0.0518**	0.0369	0.1041	0.0866	0.0806	**0.0770**
1.65	0.1318	0.0271	0.0509	0.0046	**0.0536**	0.2194	0.0772	0.1143	0.0820	**0.1232**
1.70	0.1054	0.0295	0.0496	0.0022	**0.0467**	0.4355	0.0447	0.0740	0.0831	**0.1593**
1.75	0.1130	0.0293	0.0481	0.0002	**0.0476**	0.0276	0.0492	0.1214	0.0849	**0.0708**
1.80	0.1172	0.0767	0.0449	0.0025	**0.0604**	0.0130	0.0402	0.0475	0.0860	**0.0467**
1.85	0.1260	0.0216	0.0453	0.0047	**0.0494**	0.0185	0.0642	0.0696	0.0878	**0.0600**
1.90	0.1002	0.0242	0.0439	0.0067	**0.0438**	0.2446	0.0065	0.1293	0.0899	**0.1176**
1.95	0.0895	0.0136	0.0284	0.0087	**0.0350**	0.0192	0.0647	0.0645	0.0911	**0.0599**