INSTITUT FÜR INFORMATIK
Datenbanken und Informationssysteme

Universitätsstr. 1 D–40225 Düsseldorf

HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

Interrater-Reliabilitätskoeffizienten: Analysen und Simulationen

Daniel Laps

Bachelorarbeit

Beginn der Arbeit:	6. Juli 2017
Abgabe der Arbeit:	6. Oktober 2017
Gutachter:	Prof. Dr. Stefan Conrad
	Prof. Dr. Martin Mauve

Erklärung

Hiermit versichere ich, dass ich diese Bachelorarbeit selbstständig verfasst habe. Ich habe dazu keine anderen als die angegebenen Quellen und Hilfsmittel verwendet.

Düsseldorf, den 6. Oktober 2017

Daniel Laps

Zusammenfassung

In der Wissenschaft werden oft Untersuchungsgegenstände der Forschung durch mehrere Codierer kategorisiert. Nach Abschluss der Kategorisierung, auch Annotation genannt, ist von Interesse, ob die Codierer in ihren Urteilen übereinstimmen. Der Grad der Übereinstimmung ist ein wichtiger Indikator, um die Qualität der Kategorisierungen zu bewerten. Für die Messung der Übereinstimmung sind von verschiedenen Personen ein ganze Reihe von sogenannten Interrater-Reliabilitätskoeffizienten entwickelt worden. Einige der Koeffizienten werden in dieser Arbeit erläutert.

Im Rahmen der Bachelorarbeit ist ein Simulationsprogramm entwickelt worden. Das Programm simuliert die Annotation und die Berechnung der Reliabilitätskoeffizienten. Hierzu wird in einem ersten Schritt vom Benutzer des Programms angegeben, wie die Annotationsdatensätze generiert werden sollen. Der Benutzer kann verschiedene Parameter, die Eigenschaften der generierten Datensätze beschreiben, spezifizieren. Nach der Generierung der Datensätze berechnet das Programm die Werte der Koeffizienten. Anschließend speichert das Programm die Werte und gegebenenfalls auch die generierten Datensätze. Durch die Variation der Parameter bei Generierung der Datensätze lassen sich verschiedene Eigenschaften der Koeffizienten untersuchen.

Mit Hilfe des Simulationsprogramms sind eine ganze Reihe von Simulationen durchgeführt worden. Die Ergebnisse sind zur Analyse der Koeffizienten benutzt worden. Hier zeigte sich, dass, bis auf einer Ausnahme, alle der im Rahmen der Arbeit behandelten Koeffizienten in etwa das gleiche Verhalten zeigen. Dennoch konnte an Hand der Analysen anderer Autoren eine Empfehlung für zwei der Koeffizienten ausgesprochen werden. Es konnte demonstriert werden, dass die Analyse der Koeffizienten durch Simulationen für einige Fragestellungen eine gute Methode darstellt. Allerdings konnten einige andere Fragestellungen nicht mit dieser Methode bearbeitet werden.

Inhaltsverzeichnis

1 Einleitung

In diesem einleitenden Kapitel werden die Motivation und die zugrundeliegende Aufgabenstellung der Bachelorarbeit dargestellt, sowie der Aufbau der Arbeit beschrieben.

1.1 Motivation

In vielen Bereichen der Wissenschaft ist es oft notwendig, Untersuchungsgegenstände der Forschung zu kategorisieren: Sei es in der Computerlinguistik bei der Annotation von Texten, in der Medizin in der Diagnose von Krankheiten von Patienten oder zum Beispiel in der Informatik bei der Erstellung von Trainingsdaten für Machine Learning. Kategorisieren mehrere Personen die gleichen Daten, dann stellt sich die Frage nach der Übereinstimmung zwischen den Personen als ein wichtiger Indikator zur Bewertung der Qualität der Kategorisierungen.

Verschiedene Interrater-Reliabilitätskoeffizienten sind entwickelt worden, um die Übereinstimmung zu messen. Es stellt sich jedoch die Frage, welche Koeffizienten am besten die Übereinstimmung messen und daher den Anderen vorzuziehen sind. Für einen gewinnbringenden Einsatz in der Praxis ist zudem zu analysieren, welche Eigenschaften die Koeffizienten haben, worin sie sich unterscheiden und was es bei ihrem Einsatz zu beachten gibt.

In der Literatur finden sich bereits Analysen von Interrater-Reliabilitätskoeffizienten, die aber alle Aussagen über die Koeffizienten ableiten, indem die Berechnungsschritte der Koeffizienten analysiert werden. Eine andere Art die Analyse der Koeffizienten anzugehen kann in der Simulation durch das Generieren geeigneter Testdaten und anschließender Berechnung der Koeffizienten auf diesen Testdaten bestehen. Ein eigens für die Simulation entwickeltes Computerprogramm kann hier die Möglichkeit schaffen, diese Art der Analyse durchzuführen.

1.2 Aufgabenstellung

Die Aufgabe dieser Bachelorarbeit lässt sich in drei Teile unterteilen: Häufig verwendete Interrater-Reliabilitätskoeffizienten sollen beschrieben werden und ein geeignetes Simulationsprogramm entwickelt werden. Anschließend soll auf der Basis der Beschreibung der Koeffizienten und von Simulationen, die mit dem entwickelten Simulationsprogramm durchgeführt worden sind, die Koeffizienten analysiert werden und Handlungsempfehlungen gegeben werden.

1.3 Aufbau der Arbeit

Diese Bachelorarbeit gliedert sich in sieben Kapitel. Das erste Kapitel führt in die Arbeit ein. Im zweiten Kapitel werden die grundlegenden Begriffe und Konzepte im Kontext von Interrater-Reliabilitätskoeffizienten vorgestellt. Das folgende dritte Kapitel beschreibt im Detail die in der Bachelorarbeit vorkommenden Koeffizienten. Der Ablauf einer Simulation und die für das Simulationsprogramm entwickelten Konzepte werden

in Kapitel vier vorgestellt. Im Anschluss wird in Kapitel fünf die Implementierung des Programms beschrieben. Im sechsten Kapitel werden die mit dem Simulationsprogramm gewonnen Simulationsergebnisse vorgestellt, analysiert und daraus Schlüsse für die Benutzung der Koeffizienten gezogen. Das letzte Kapitel fasst die Ergebnisse der Arbeit zusammen und stellt mögliche weiterführende Arbeiten vor.

Der Bachelorarbeit liegt eine CD bei. Auf dieser CD ist der Quellcode des Simulationsprogramms, die Simulationsergebnisse und die Beschreibungen aller durchgeführten Simulationen zu finden. Eine weitergehende Beschreibung liefert enthaltene README-Datei.

2 Grundlagen

Folgendes Kapitel erläutert die grundlegenden Begriffe bezüglich der Interrater-Reliabilitätskoeffizienten.

Die Veröffentlichungen, auf die in dieser Bachelorarbeit Bezug genommen werden, sind in einem Zeitraum von über 50 Jahren von Autoren aus unterschiedlichen Fachrichtungen und mit unterschiedlichen Anwendungsfällen im Kopf geschrieben worden. Es verwundert daher nicht, dass die verwendeten Ausdrücke für ein und denselben Begriff von Autor zu Autor zum Teil stark voneinander abweichen. Aus diesem Grund werden in Tabelle 1 die einheitlich in der Bachelorarbeit verwendeten wichtigen Ausdrücke für einen Begriff in Deutsch und Englisch und die Ausdrücke für die Begriffe in der Literatur gegenübergestellt, um so Missverständnisse auszuschließen.

Deutscher Ausdruck	**Englischer Ausdruck**	**Weitere Ausdrücke in der Literatur**
Codierer	rater	observer [Kri04], coder [AP07], judge [Coh60]
Datensatz	data set	record [Kri04], sample [Sco55]
Kategorie	category	value [Kri04]
Element	token	unit [Kri04], item [AP07], subject [Fle71]
Einheit	unit	
Prozentuale Übereinstimmung/beobachtete Übereinstimmung	percentage agreement/observed agreement	percentage of judgments [Sco55]

Tabelle 1: Ausgewählte verwendete Ausdrücke und ihre Synonyme in der Literatur

2.1 Annotation von Daten

Annotation bedeutet im Kontext von Interrater-Reliabilitätskoeffizienten, dass einzelne Objekte, die untersucht werden sollen, von Codierern mit Kategorien ausgezeichnet werden [AP07]. Alle Objekte zusammen bilden mit den Zuordnungen der Kategorien zu den Objekten einen Datensatz. Ein Datensatz wären zum Beispiel Feststellungen durch Ärzte, welche Symptome die untersuchten Patienten haben. In diesem Fall wären die Patienten die Objekte, die einzelnen Symptome die Kategorien und die Ärzte die Codierer. Die Codierer können sowohl Personen sein, als auch Messgeräte oder zum Beispiel eine mathematische Funktion [Kri04, Abs. 11.2]. Notwendig und hinreichend ist nur die Fähigkeit, bei Betrachtung eines Objektes, urteilen zu können, zur welcher Kategorie das Objekt gehört.

Es kann bezüglich der Annotation zwischen Codierung und Unitizing unterschieden werden [Kri04, Abs. 11.2].

2.1.1 Codierung

Bei der Codierung von Daten beurteilen die Codierer die einzelnen Elemente eines Datensatzes. Sie ordnen, wenn sie zu einem Urteil gekommen sind, den Elementen Kategorien zu. Elemente wären zum Beispiel die Wörter eines Textkorpus oder im obigen Beispiel die Patienten für die Symptome festgestellt werden. Zu Unterscheiden ist zwischen dem Fall, dass einem Element je Codierer genau eine Kategorie zugeordnet werden kann (einfach-kategoriell) oder einem Element je Codierer mehrere Kategorien zugeordnet werden können (mehrfach-kategoriell) [Kri04, Abs. 11.2].

Die Abgrenzung der einzelnen Elemente zueinander ergibt sich aus der Beschaffenheit der Daten und dem Ziel der Codierung. Sie wird vor dem Beginn der Codierung festgelegt. Die Codierer entscheiden beim Codieren nicht über die Grenzen der einzelnen Elemente. Die Elementgrenzen sind so zu wählen, dass die Elemente für einen Codierer unterschiedliche Bedeutungen haben können [Kri04, Abs. 11.2].

Einzelne Wörter wären zum Beispiel geeignete Elemente, um einen Text bezüglich Wortarten zu codieren. Während einzelne Buchstaben als Elemente keine geeigneten Elemente für diese Codierungsaufgabe wären, weil Wortarten nur für einzelne Wörter definiert sind und daher die Codierer für einzelne Buchstaben eines Wortes stets die gleiche Kategorie vergeben würden. Ebenfalls wären größere Elementgrenzen nicht geeignet, weil schon zwei nebeneinander stehende Wörter verschiedene Wortarten haben können und die Codierer in diesem Fall dann nicht wissen, welche Kategorien sie diesem Element zuordnen sollen.

Tabelle 2 stellt ein Beispiel für eine einfach-kategorielle Codierung dar. Zwei Codierer haben den Satz „Das Kind spielt Fußball.“ nach Wortarten codiert. Als Kategorien standen die üblichen zehn Wortarten (Artikel, Nomen, Verb, Adjektiv etc.) der deutschen Schulgrammatik zur Verfügung. Beim ersten Element gibt es eine Abweichung zwischen den Codierern: Der erste Codierer hält das erste Element für ein Artikel. Der zweite Codierer identifiziert Artikel nicht als Wortart und hält den bestimmten Artikel „der“ für ein Pronomen.

	1	2	3	4
	Das	**Kind**	**spielt**	**Fußball**
Codierer 1	Artikel	Nomen	Verb	Nomen
Codierer 2	Pronomen	Nomen	Verb	Nomen

Tabelle 2: Beispiel für eine Codierung im einfach-kategoriellen Fall

Im Weiteren wird nur der einfach-kategorielle Fall weiter betrachtet, da alle betrachteten Koeffizienten nur für diesen Fall definiert sind.

2.1.2 Unitizing

Unter Unitizing wird die Unterteilung eines Datensatzes in Abschnitte durch zwei oder mehr Codierer für eine Kategorie oder mehrere gegebene Kategorien verstanden. Es gibt Abschnitte, die für eine Kategorie von Bedeutung sind, und andere Abschnitte, die es nicht sind. Die Abschnitte, die von Bedeutung sind, werden Einheiten genannt. Da der

Datensatz für jede Kategorie einzeln in Abschnitte unterteilt wird, können sich die Einheiten des gleichen Codierer aber für unterschiedliche Kategorien überlappen. Im Gegensatz zu Codierung im einfach-kategoriellen Fall kann beim Unitizing ein Teil des Datensatz mit mehr als einer Kategorie ausgezeichnet sein [Kri04, Kap. 11].

Die Daten müssen für das Unitizing in kleinste unterscheidbare Einheiten (kuE) zu unterteilen sein und diese Einheiten müssen sich ordnen lassen. Die Abschnitte sind dann durch ihren Startpunkt und ihre Länge eindeutig innerhalb des Datensatzes lokalisiert, wobei es ausreicht, nur die Einheiten zu spezifizieren, um festzulegen welche Abschnitte Einheiten sind und welche nicht. Alle Zwischenräume zwischen zwei Einheiten sind Abschnitte ohne Einheiten zu sein. Für einen Text, auf den Unitizing angewendet werden soll, wären zum Beispiel die einzelnen Buchstaben die kleinsten unterscheidbaren Einheiten [Kri04, Kap. 11].

Ein Beispiel für Unitizing durch zwei Codierer wird in Tabelle 3 dargestellt. Unitizing soll auf den Satz – „Die Turingmaschine ist deterministisch.“ – bezüglich der zwei Kategorien Fachsprachlich (der Teil des Textes enthält Fachsprache) und Fremdsprachlich (der Teil des Textes enthält Fremdwörter) angewendet werden. KuEs sind die einzelnen Buchstaben. Der erste Codierer ist der Meinung, dass „Turingmaschine ist deterministisch“ im Gesamten fachsprachlich ist und markiert den gesamten Teil als eine Einheit, während der zweite Codierer meint das nur „Turingmaschine“ und „deterministisch“ für sich genommen fachsprachlich ist und markiert entsprechend zwei Einheiten. Keine fremdsprachlichen Teile meint der erste Codierer zu erkennen und markiert gar keine Einheiten. Für den zweiten Codierer sind „Turingmaschine“ und „deterministisch“ fremdsprachlich und markiert wiederum zwei Einheiten.

	1	4	18	21
	Die	**Turingmaschine**	**ist**	**deterministisch**
Codierer 1 - Fachspr.		S: 4, L:31		
Codierer 2 - Fachspr.		S:4, L:14		S:21, L:15
Codierer 1 - Fremdspr.				
Codierer 2 - Fremdspr.		S:4, L:14		S:21, L:15

Tabelle 3: Beispiel für Unitizing. S=Startpunkt; L:Länge; Fachspr.=Fachsprachlich; Fremdspr.=Fremdsprachlich

2.2 Reliabilität

Annotierte Daten, die in der Forschung, zur Argumentation oder zum Beispiel als Grundlagen für Machine Learning benutzt werden sollen, sollten über Zweifel erhaben sein. Dies bedeutet, dass unter Ausschluss Verzerrungen die Daten annotiert werden. Denkbarer Verzerrungen sind

- die unterschiedlichen Wahrnehmungen der Codierer d. h. Codierer nehmen das gleiche Element als etwas unterschiedliches wahr,
- die einseitigen Neigungen der Codierer d. h. zum Beispiel, dass ein Codierer jene Kategorie einer anderen Kategorie vorzieht, während es bei einem anderen Codierer umgekehrt ist und

- das nicht Einhalten der Bedingung, dass alle das Gleiche unter den annotierten Daten und den Kategorien verstehen.

Reliabilität ist dann die Stärke mit der die Verzerrungen für die gegebenen, annotierten Daten minimiert worden sind und die empirisch gemessen wird [Kri04, Abs. 11.1].

Reliabilität kann empirisch gemessen werden, indem die Übereinstimmung der Codierung in der Zuweisung von Kategorien zu den Daten gemessen wird. Wenn verschiedene Codierer mit hoher Übereinstimmung annotieren, dann kann geschlussfolgert werden, dass diese Codierer eine gleiche Bedeut ung der Annotationsrichtlinien verinnerlicht haben und die Ergebnisse nicht von den Codieren abhängig sind [AP07].

Reliabilität macht keine Aussage über den Wahrheitsgehalt der Daten. Zwei Codierer können zum Beispiel in ihren Urteilen übereinstimmen, aber dennoch objektiv falsch liegen. [Kri04, Abs. 11.1]

Es kann zwischen verschiedene Arten von Reliabilität unterschieden werden: Stabilität, Reproduzierbarkeit und Genauigkeit.

Stabilität: Stabilität ist das Ausmaß wie stark die Übereinstimmung über die Zeit für einen gegebenen einzelnen Codierer ist – gemessen als Übereinstimmung zwischen zwei Annotationen verschiedener Zeitpunkte (auch Intrarater-Reliabilität oder Intrarater-Übereinstimmung genannt)[AP07].

Reproduzierbarkeit: Unter der Bedingung, dass die Codierer unabhägig von einander arbeiten, ist Reproduzierbarkeit das Ausmaß der Übereinstimmung der Ergebnisse zwischen den Codierern (Interrater-Reliabilität oder Interrater-Reliabilität).

Genauigkeit: Das Ausmaß der Übereinstimmung zwischen dem Ergebnis eines Coders und einem definierten Standard wird Genauigkeit genannt.

3 Erläuterung der Interrater-Reliabilitätskoeffizienten

Dieses Kapitel beschreibt alle im Simulationsprogramm implementierten Interrater-Reliabilitätskoeffizienten.

Aufgabe eines Koeffizienten ist es zu einem, die Interrater-Reliabilität zu messen und zum anderen, die Ergebnisse für verschiedene Datensätze vergleichbar zu machen.

Im Folgenden sei für die mathematische Definition der Koeffizienten ein Datensatz eine Menge $\mathbf{E} = \{e_1, \ldots, e_N\}$ bestehend aus N Elementen. Weiter sei jedes Element e eine Funktion. Die Funktion repräsentiert die Zuordnungen von Kategorien durch die Codierer zu dem Element e. Definiert ist die Funktion für die Menge der Codierer $\mathbf{C} = \{c_1, \ldots c_C\}$ bestehend aus C Codierern und sie bildet ab auf die Menge der Kategorien $\mathbf{K} = \{k_1, \ldots, k_K\}$ bestehend aus K Kategorien mit $\forall i \in \mathbb{N} : k_i \in \mathbb{R}$. Weiter sei $n_k = \sum_{i=1}^{C} |\{e \in \mathbf{E} | e(c_i) = k\}|$ die Anzahl, wie oft die Kategorie k von den Codierern einem Element zugeordnet worden ist, $n_{ik} = |\{c \in \mathbf{C} | e_i(c) = k\}|$ die Anzahl wie oft das Element e_i von den Codierern die Kategorie k zugeordnet worden ist und $n_{c_i k} = |\{e \in \mathbf{E} | e(c_i) = k\}|$, wie oft der Codierer c_i die Kategorie k_j Elementen zugeordnet hat.

In Tabelle 4 ist ein Beispieldatensatz dargestellt, der benutzt wird, um die einzelnen Koeffizienten an einem Beispiel zu erläutern. In diesem Beispiel haben die drei Codierer A, B und C (also $C = 3$; $\mathbf{C} = \{A, B, C\}$) zehn Elemente (also $N = 10$) codiert, indem sie den einzelnen Elementen eine Kategorie aus $\mathbf{K} = \{1, 2, 3\}$ zuordnen. Weiter gilt zum Beispiel $e_2(A) = 2$, da Codierer A das zweite Element der Kategorie 2 zuordnet. $n_{10,2} = 1$, da es einen Codierer (nämlich A) gibt, der dem zehnten Element die Kategoire 2 zuordnet und $n_{A,3} = 2$, da der Codierer A zweimal einem Element die Kategorie Zwei zugeordnet hat.

	1	2	3	4	5	6	7	8	9	10
Codierer A	1	2	3	2	2	3	2	2	2	2
Codierer B	1	2	3	1	2	3	1	2	2	3
Codierer C	1	2	3	1	2	2	2	2	2	3

Tabelle 4: Beispieldatensatz für eine Codierung

Tabelle 5 stellt eine sogenannte Übereinstimmungstabelle (agreement table) dar. In dieser Tabelle wird für jedes Element angeben, wie viele Codierer sich bei diesem Element für eine Kategorie entschieden haben. Mit dieser Tabelle lässt sich daher einfach den Wert für n_{ik} bestimmen: Der Wert steht in der Spalte der Kategorie k und dort in der Zeile des i-ten Elements.

3.1 Prozentuale Übereinstimmung als Koeffizient

Die naheliegendste Lösung, um die Übereinstimmung zwischen zwei Codierern zu messen, ist die Berechnung der prozentualen Übereinstimmung ($U_b^{C=2}$, später auch beobachtete Übereinstimmung genannt). Sie wird gemessen als Verhältnis zwischen der Anzahl an Elementen für die die beiden Codierer zum gleichen Urteil gekommen sind und die

Gesamtanzahl an Elementen:

$$U_b^{C=2} = \frac{|\{e_i \in \mathbf{E} | e_i(c_1) = e_i(c_2)\}|}{N} = \frac{\sum_{i=1}^{N} f(i)}{N} \text{ mit } f(i) = \begin{cases} 1 & \text{falls } e_i(c_1) = e_i(c_2) \\ 0 & \text{sonst} \end{cases} \quad (1)$$

Per Definition kommen die Codierer also für ein Element zum gleichen Urteil, wenn sie diesem Element die selbe Kategorie zuordnen [AP07].

Für den Beispieldatensatz und den Codierern A und B gilt beispielsweise:

$$U_b^{C=2} = \frac{7}{10}$$

Für die Erweiterung auf mehr als zwei Codierer werden zur Messung der Übereinstimmung für jedes Element Paare der Urteile gebildet, weil es möglicherweise Elemente gibt, bei denen ein Teil der Codierer in ihren Zuordnungen übereinstimmen und ein Teil nicht. Zum Beispiel gibt es bei drei Codierern drei verschiedene Paare: Ein Paar bestehend aus der Zuordnung des ersten und zweiten Codierers, ein Paar aus der Zuordnung des ersten und dritten Codierers und ein Paar aus der Zuordnung des zweiten und dritten Codierers. Die prozentuale Übereinstimmung (U_b) wird dann gemessen als das Verhältnis zwischen der Anzahl an Paaren aller Elemente, die übereinstimmen, und der Gesamtanzahl an Paaren [AP07],

Pro Element ist die Anzahl aller Paare gegeben durch $\binom{C}{2}$. Die Anzahl der Paare, die übereinstimmen, für das i-te Element ist gleich $\sum_{k \in \mathbf{K}} \binom{n_{ik}}{2}$. Die prozentuale Übereinstimmung ist dann definiert als [AP07]:

$$U_b = \frac{1}{N\binom{C}{2}} \sum_{i=1}^{N} \sum_{k \in \mathbf{K}} \binom{n_{ik}}{2} = \frac{1}{NC(C-1)} \sum_{i=1}^{N} \sum_{k \in \mathbf{K}} n_{ik}(n_{ik} - 1) \quad (2)$$

Kategorien:	1	2	3
Element 1	3	0	0
Element 2	0	3	0
Element 3	0	0	3
Element 4	2	1	0
Element 5	0	3	0
Element 6	0	1	2
Element 7	1	2	0
Element 8	0	3	0
Element 9	0	3	0
Element 10	0	1	2

Tabelle 5: Übereinstimmungstabelle für den Beispieldatensatz

Für den Beispieldatensatz gilt:

$$U_b = \frac{\overbrace{(3(3-1)+0+0)}^{=6}+6+6+\overbrace{(2(2-1)+1(1-1)+0)}^{=2}+6+2+2+6+6+2}{10 \cdot 3(3-1)}$$
$$= \frac{44}{60} = \frac{11}{15} \approx 0.73$$

Für den Fall, dass es nur zwei Codierer gibt, gilt $U_b^{C=2} = U_b$. Der Beweis ist im Anhang zu finden.

Prozentuale Übereinstimmung ist kein geeigneter Koeffizient, da er nicht ermöglicht, die Ergebnisse verschiedener Datensätze zu vergleichen. Der Grund ist, dass bereits bei völlig zufälliger Zuordnung durch die Codierer eine gewisse Übereinstimmung zu erwarten ist. Die erwartete Übereinstimmung ist abhängig von der Beschaffenheit des Datensatzes: Zum einen steigt die erwartete Übereinstimmung, desto weniger Kategorien zur Auswahl stehen und zum anderen steigt die erwartete Übereinstimmung, desto weniger die Kategorien gleich häufig verwendet werden. Ziel der folgenden Koeffizienten ist es daher die erwartete Übereinstimmung (U_e) zu bestimmen und diese dann ins Verhältnis zu der beobachten Übereinstimmung (U_b) zu setzen. Die Ergebnisse, auch zwischen Datensätze mit unterschiedlicher Anzahl von Kategorien und mit unterschiedlicher Häufigkeit des Auftretens von einzelnen Kategorien, werden so vergleichbar gemacht [AP07].

3.2 Zufallskorrigierte Koeffizienten für Datensätze mit zwei Codierern

Alle in diesem Abschnitt beschriebenen Koeffizienten gehen wie folgt vor, um die Übereinstimmung durch Zufall herausrechnen zu können: Zuerst wird die beobachtete Übereinstimmung ($U_b^{C=2}$) (gemessen als prozentuale Übereinstimmung) und die erwartete Übereinstimmung (U_e) berechnet. Anschließend drückt $1 - U_e$ aus, wie stark die beobachtete Übereinstimmung über der erwarteten Übereinstimmung liegen kann. $U_b^{C=2} - U_e$ beschreibt dann, wie stark tatsächlich die beobachtete Übereinstimmung über der erwarteten Übereinstimmung liegt. Das Verhältnis dieser beiden Werte ergibt dann den Wert der Koeffizienten S, π, κ:

$$S, \pi, \kappa = \frac{U_b^{C=2} - U_e^{S,\pi,\kappa}}{1 - U_e^{S,\pi,\kappa}} \tag{3}$$

Die Koeffizienten nehmen den Wert null an, wenn die beobachtete Übereinstimmung der erwarteten Übereinstimmung entspricht. Der Wert eins wird erreicht, wenn die beobachtete Übereinstimmung den Wert eins erreicht, d. h. total ist. Der Koeffizient wird nur dann einen Wert unter null annehmen, wenn die Übereinstimmung geringer ist als das zu erwarten wäre, wenn die Codierer zufällig den Elementen Kategorien zuordnen. Die Koeffizienten dieses Abschnitts unterscheiden sich nur darin wie die erwartete Übereinstimmung abgeschätzt wird [AP07].

Unter der Annahme, dass die beiden Codierer c_1 und c_2 unabhängig von einander die Kategorien zuordnen, ist die Wahrscheinlichkeit, dass die Beiden für eine Kategorie k übereinstimmen: $P(k|c_1)P(k|c_2)$, wobei $P(k|c_1)$ bzw. $P(k|c_2)$ die Wahrscheinlichkeit ist,

dass Codierer c_1 bzw. c_2 einem Element die Kategorie k zuordnet. Die erwartete Übereinstimmung ist dann die Summe über alle Kategorien [AP07]:

$$U_e = \sum_{k \in \mathbf{K}} P(k|c_1)P(k|c_2) \tag{4}$$

3.2.1 Bennetts S

Bennetts S ist ein Koeffizient, der zuerst in einem Artikel [BAG54] von E. M. Bennett, R. Alpert und A. C. Goldstein beschrieben worden ist. Der Schätzung der erwarteten Übereinstimmung liegt die Annahme zu Grunde, dass alle Kategorien gleich wahrscheinlich sind, d. h. es gilt $P(k|c_1) \approx P'(k|c_1) = \frac{1}{K} = P'(k|c_2) \approx P(k|c_2)$. Daraus folgt nach Einsetzen in die Gleichung 4, dass die erwartete Übereinstimmung wie folgt berechnet wird [AP07]:

$$U_e^S = \sum_{k \in \mathbf{K}} P'(k|c_1)P'(k|c_2) = \sum_{k \in \mathbf{K}} \frac{1}{K} \cdot \frac{1}{K} = \frac{K}{K^2} = \frac{1}{K} \tag{5}$$

Bennett, Alpert und Goldstein geben in ihrem Artikel eine leicht andere Formel für S an als die in Gleichung 3:

$$S' \frac{K}{K-1}(U_b^{C=2} - \frac{1}{K})$$

Es lässt sich aber leicht zeigen, dass diese sich in die Gleichung 3 umformen lässt:

$$\begin{aligned} S' =& \frac{K}{K-1}\left(U_b^{C=2} - \frac{1}{K}\right) = \frac{KU_b^{C=2}}{K-1} - \frac{K}{(K-1)K} \\ =& \frac{KU_b^{C=2} - 1}{K-1} = \frac{U_b^{C=2} - \frac{1}{K}}{1 - \frac{1}{K}} = \frac{U_b^{C=2} - U_e^S}{1 - U_e^S} = S \end{aligned}$$

Der Beispieldatensatz enthält drei Kategorien, womit Bennetts S folgenden Wert für den Datensatz annimmt:

$$S = \frac{U_b^{C=2} - U_e^S}{1 - U_e^S} = \frac{U_b^{C=2} - \frac{1}{K}}{1 - \frac{1}{K}} = \frac{\frac{7}{10} - \frac{1}{3}}{1 - \frac{1}{3}} = \frac{11}{20} = 0.55$$

3.2.2 Scotts Pi

Scotts Pi ist zuerst beschrieben worden durch William S. Scott [Sco55]. Anders als bei Bennetts S, wird nicht angenommen, dass alle Kategorien gleich wahrscheinlich sind. Stattdessen wird die Wahrscheinlichkeit, einem Element eine bestimmte Kategorie zuzuordnen, abgeschätzt durch die Häufigkeit mit der die Codierer Elementen diese Kategorie zuordnen. Aus diesem Grund gilt: $P(k|c_1) \approx P'(k|c_1) = \frac{n_k}{2N} = P'(k|c_2) \approx P(k|c_2)$. Für die erwartete Übereinstimmung gilt dann weiter:

$$U_e^\pi = \sum_{k \in \mathbf{K}} P'(k|c_1)P'(k|c_2) = \sum_{k \in \mathbf{K}} \frac{n_k}{2N} \cdot \frac{n_k}{2N} = \sum_{k \in \mathbf{K}} \frac{n_k^2}{4N^2} \tag{6}$$

Die beiden Codierer A und B haben im Beispielsatz viermal den Elementen die Kategorie Eins zugeordnet, elfmal die Kategorie Zwei und fünfmal die Kategorie Drei. Damit ergibt sich für die Codierer A und B folgende erwartete Übereinstimmung:

$$U_e^\pi = \sum_{k \in \mathbf{K}} \frac{n_k^2}{4N^2} = \frac{4^2 + 11^2 + 5^2}{4 \cdot 10^2} = \frac{162}{400} = \frac{81}{200} = 0.405$$

Insgesamt ergibt das für Scotts Pi und den Datensatz:

$$\pi = \frac{U_b^{C=2} - U_e^\pi}{1 - U_e^\pi} = \frac{\frac{7}{10} - \frac{81}{200}}{1 - \frac{81}{200}} = \frac{59}{119} \approx 0.4958$$

3.2.3 Cohens Kappa

Für Cohens Kappa, beschrieben durch Jacob Cohen [Coh60], wird die Wahrscheinlichkeit, dass der Codierer c_1 bzw. c_2 einem Element eine bestimmte Kategorie zuordnet abgeschätzt durch die Häufigkeit mit der dieser eine Codierer c_1 bzw. c_2 Elementen eine bestimmte Kategorie zuordnet. Es gilt $P(k|c_1) \approx P'(k|c_1) = \frac{n_{c_1 k}}{N}$ und entsprechend $P(k|c_2) \approx P(k|c_2) = \frac{n_{c_2 k}}{N}$. Es können sich also für die beiden Codierer unterschiedliche Wahrscheinlichkeiten ergeben. Das ist ein Unterschied zu Scotts Pi, bei dem die Schätzung der Wahrscheinlichkeit, dass eine bestimmte Kategorie einem Element zugeordnet wird gegeben ein Codierer, nur von der Kategorie abhängt. Bei Bennetts S hängt die Schätzung, weder vom Codierer, noch von der Kategorie ab. Mit der Gleichung 4 ergibt sich die erwartete Übereinstimmung für Cohens Kappa wie folgt:

$$U_e^\kappa = \sum_{k \in \mathbf{K}} P'(k|c_1) P'(k|c_2) = \sum_{k \in \mathbf{K}} \frac{n_{c_1 k}}{N} \cdot \frac{n_{c_2 k}}{N} = \sum_{k \in \mathbf{K}} \frac{n_{c_1 k} n_{c_2 k}}{N^2} \tag{7}$$

n_{ck}	$k=1$	$k=2$	$k=3$
$c=$ Codierer A	1	7	2
$c=$ Codierer B	3	4	3

Tabelle 6: Für den Beispieldatensatz die Anzahl der zu einer Kategorie zugeordneten Elemente aufgeschlüsselt nach den Codierern A und B

In Tabelle 6 stehen die Werte für n_{ck} bezüglich des Beispieldatensatzes und den Codierern A und B. Die erwartete Übereinstimmung wird dann wie folgt berechnet:

$$U_e^\kappa = \sum_{k \in \mathbf{K}} \frac{n_{c_1 k} n_{c_2 k}}{N^2} = \frac{1 \cdot 3 + 7 \cdot 4 + 2 \cdot 3}{10^2} = \frac{37}{100} = 0.37$$

Das ergibt insgesamt als Wert für Cohens Kappa und dem Beispieldatensatz:

$$\kappa = \frac{U_b^{C=2} - U_e^\kappa}{1 - U_e^\kappa} = \frac{\frac{7}{10} - \frac{37}{100}}{1 - \frac{37}{100}} = \frac{11}{21} \approx 0.5238$$

3.3 Zufallskorrigierte Koeffizienten für Datensätze mit beliebig vielen Codierern

In diesem Abschnitt werden die zufallskorrigierten Koeffizienten so erweitertet, dass sie auch für Datensätze mit mehr als zwei Codierern berechnet werden können.

Unglücklicherweise werden in der Literatur die erweiterten Koeffizienten alle mit dem Buchstaben Kappa bezeichnet, sodass in Formeln möglicherweise unklar bleibt, welcher Koeffizient gemeint ist. Aus diesem Grund werden in dieser Arbeit die Koeffizienten in Formeln nach folgender Konvention benannt: Als Buchstabe zur Identifikation für einen erweiterten Koeffizient wird der Buchstabe des Koeffizienten, der erweitert wurde, verwendet (also S, π und κ) und zusätzlich mit einem hochgestellten $C > 2$ versehen. Die Erweiterung von Bennetts S wird zum Beispiel mit $S^{C>2}$ bezeichnet.

Aus den bereits im letzten Abschnitt erläuterten Gründen gilt auch für die erweiterten Koeffizienten:

$$S^{C>2}, \pi^{C>2}, \kappa^{C>2} = \frac{U_b - U_e^{S^{C>2},\pi^{C>2},\kappa^{C>2}}}{1 - U_e^{S^{C>2},\pi^{C>2},\kappa^{C>2}}} \tag{8}$$

Für die beobachtete Übereinstimmung wird konsequenterweise jetzt die Erweiterung der prozentualen Übereinstimmung auf mehr als zwei Codierer verwendet [AP07].

Da die beobachtete Übereinstimmung für mehr als zwei Codierer paarweise berechnet wird, ist es sinnvoll, dies bei der erwarteten Übereinstimmung ebenfalls zu tun. Im Allgemeinen gilt:

$$U_e = \sum_{k \in \mathbf{K}} \frac{1}{\binom{C}{2}} \sum_{i=1}^{C-1} \sum_{j=i+1}^{C} P(k|c_i) P(k|c_j) \tag{9}$$

Für jedes Paar an Codierern wird die Wahrscheinlickeit berechnet, dass die diese beiden Codierer für eine bestimmte Kategorie übereinstimmen und die Summe der Wahrscheinlichkeiten aller Paare wird durch die Anzahl aller Paare geteilt [AP07].

3.3.1 Erweiterung von Bennetts S: Randolphs Kappa

Randolphs Kappa, vorgestellt durch Justus J. Randolph [Ran05], stellt die Erweiterung von Bennetts S auf mehr als zwei Codierer da. Wie bei Bennetts S setzt Randolph für die erwartete Wahrscheinlichkeit:

$$U_e^{S^{C>2}} = \frac{1}{K} \tag{10}$$

Es lässt sich zeigen, dass für $P'(k|c) = \frac{1}{K}$ mit k eine beliebige Kategorie und c ein beliebiger Codierer die Definition von Randolph konsistent zu der Gleichung 9 ist, dass also gilt:

$$U_e^{S^{C>2}} = \sum_{k \in \mathbf{K}} \frac{1}{\binom{C}{2}} \sum_{i=1}^{C-1} \sum_{j=i+1}^{C} P'(k|c_i) P'(k|c_j) = \frac{1}{K}$$

Auch dieser Beweis ist im Anhang zu finden.

Der Beispieldatensatz enthält drei Kategorien d. h. es gilt $U_e^{S^{C>2}} = \frac{1}{K} = \frac{1}{3}$. Insgesamt gilt dann für Randolphs Kappa:

$$S^{C>2} = \frac{U_b - U_e^{S^{C>2}}}{1 - U_e^{S^{C>2}}} = \frac{\frac{11}{15} - \frac{1}{3}}{1 - \frac{1}{3}} = \frac{3}{5} = 0.6$$

3.3.2 Erweiterung von Scotts Pi: Fleiss Kappa

Eine Erweiterung von Scotts Pi auf mehr als zwei Codierer stellt Fleiss Kappa dar [Fle71]. Pro Element ist die Anzahl der Urteile gleich der Anzahl der Codierer d. h., dass die Anzahl aller Urteile gleich $N \cdot C$ ist. Analog zu Scotts Pi gilt daher $P'(k|c) = \frac{n_k}{NC}$ für eine beliebige Kategorie $k \in \mathbf{K}$ und einen beliebigen Codierer $c \in \mathbf{C}$. Unter Benutzung der bereits im Anhang bewiesenen Aussage

$$\sum_{i=1}^{C-1} \sum_{j=i+1}^{C} x = \frac{C(C-1)x}{2}$$

und setzen von $x := \frac{n_k}{NC}$ gilt weiter:

$$\begin{aligned}
U_e^{\pi^{C>2}} &= \sum_{k \in \mathbf{K}} \frac{1}{\binom{C}{2}} \sum_{i=1}^{C-1} \sum_{j=i+1}^{C} P'(k|c_i) P'(k|c_j) \\
&= \sum_{k \in \mathbf{K}} \frac{1}{\binom{C}{2}} \sum_{i=1}^{C-1} \sum_{j=i+1}^{C} \frac{n_k}{NC} \cdot \frac{n_k}{NC} \\
&= \sum_{k \in \mathbf{K}} \frac{\binom{C}{2} \frac{n_k^2}{N^2 C^2}}{\binom{C}{2}} \\
&= \frac{1}{N^2 C^2} \sum_{k \in \mathbf{K}} n_k^2
\end{aligned}$$

Die Codierer A, B und C haben im Beispielsatz sechsmal den Elementen die Kategorie Eins zugeordnet, 17-mal die Kategorie Zwei und siebenmal die Kategorie Drei. Damit ergibt sich für die Codierer folgende erwartete Übereinstimmung:

$$U_e^{\pi^{C>2}} = \frac{1}{NC} \sum_{k \in \mathbf{K}} n_k^2 = \frac{1}{10^2 \cdot 3^2} \cdot (6^2 + 17^2 + 7^2) = \frac{187}{450} \approx 0.4156$$

Insgesamt ergibt das für Fleiss Kappa und den Datensatz:

$$\pi^{C>2} = \frac{U_b - U_e^{\pi^{C>2}}}{1 - U_e^{\pi^{C>2}}} = \frac{\frac{11}{15} - \frac{187}{450}}{1 - \frac{187}{450}} = \frac{143}{263} \approx 0.5437$$

3.3.3 Erweiterung von Cohens Kappa

Die folgende Erweiterung von Cohens Kappa geht auf Ron Artstein und Massimo Poesio zurück [AP07]. Die Wahrscheinlichkeit für die Auswahl einer Kategorie, gegeben einen Codierer, ist analog definiert zu der Definiton von Cohens Kappa, da sie nicht abhängig ist von der Anzahl der Codierer. Es gilt also $P'(k|c) = \frac{n_{ck}}{N}$ für eine beliegige Kategorie $k \in \mathbf{K}$ und einen beliebigen Codierer $c \in \mathbf{C}$. Nach Gleichung 9 ist die erwartete Wahrscheinlichkeit dann gegeben durch:

$$U_e^{\kappa^{C>2}} = \sum_{k\in\mathbf{K}} \frac{1}{\binom{C}{2}} \sum_{i=1}^{C-1} \sum_{j=i+1}^{C} P(k|c_i)P(k|c_j) = \sum_{k\in\mathbf{K}} \frac{1}{\binom{C}{2}} \sum_{i=1}^{C-1} \sum_{j=i+1}^{C} \frac{n_{c_ik}}{N} \cdot \frac{n_{c_jk}}{N} \tag{11}$$

n_{ck}	$k=1$	$k=2$	$k=3$
$c=$ Codierer A	1	7	2
$c=$ Codierer B	3	4	3
$c=$ Codierer C	2	6	2

Tabelle 7: Für den Beispieldatensatz die Anzahl der zu einer Kategorie zugeordneten Elemente aufgeschlüsselt nach den Codierern

In Tabelle 7 stehen die Werte für n_{ck} bezüglich des Beispieldatensatzes. Die erwartete Übereinstimmung wird dann wie folgt berechnet:

$$\begin{aligned}
U_e^{\kappa^{C>2}} =& \sum_{k\in\mathbf{K}} \frac{1}{\binom{C}{2}} \sum_{i=1}^{C-1} \sum_{j=i+1}^{C} \frac{n_{c_ik}}{N} \cdot \frac{n_{c_jk}}{N} \\
=& \sum_{k\in\mathbf{K}} \frac{1}{\binom{3}{2}} \sum_{i=1}^{2} \sum_{j=i+1}^{3} \frac{n_{c_ik}}{10} \cdot \frac{n_{c_jk}}{10} \\
=& \sum_{k\in\mathbf{K}} \frac{\frac{n_{1k}}{10} \cdot \frac{n_{2k}}{10} + \frac{n_{1k}}{10} \cdot \frac{n_{3k}}{10} + \frac{n_{2k}}{10} \cdot \frac{n_{3k}}{10}}{3} \\
=& \frac{0.1 \cdot 0.3 + 0.1 \cdot 0.2 + 0.3 \cdot 0.2 + 0.7 \cdot 0.4 + 0.7 \cdot 0.6 + 0.4 \cdot 0.6}{3} + \\
& \frac{0.2 \cdot 0.3 + 0.2 \cdot 0.2 + 0.3 \cdot 0.2}{3} \\
=& \frac{0.03 + 0.02 + 0.06 + 0.28 + 0.42 + 0.24 + 0.06 + 0.04 + 0.06}{3} \\
=& \frac{121}{300} \approx 0.4033
\end{aligned}$$

Das ergibt insgesamt für den Beispieldatensatz:

$$\kappa^{C>2} = \frac{U_b - U_e^{\kappa^{C>2}}}{1 - U_e^{\kappa^{C>2}}} = \frac{\frac{11}{15} - \frac{121}{300}}{1 - \frac{121}{300}} = \frac{99}{179} \approx 0.5531$$

3.4 Gewichtete zufallskorrigierte Koeffizienten

Die bis hierhin beschriebenen Koeffizienten unterscheiden alle nicht, wie stark die Nichtübereinstimmung ist. Es macht zum Beispiel keinen Unterschied, ob ein Codierer einem Element die Kategorie „Adjektiv“ zuordnet und ein anderer Codierer diesem Element die Kategorie „Adverb“ (geringe Nichtübereinstimmung) oder ein Codierer einem Element die Kategorie „Verb“ zuordnet und ein anderer Codierer dem Element die Kategorie „Nomen“ (große Nichtübereinstimmung). Die im folgenden vorzustellenden Koeffizienten messen daher die Nichtübereinstimmung und gewichten, wie stark die Nichtübereinstimmung eines Paares von Urteilen in das Endergebnis einfließt. Die Nichtübereinstimmung wird durch Metriken gemessen (s. u.) [AP07].

Da bei gewichteten zufallskorrigierten Koeffizienten die Nichtübereinstimmung betrachtet wird und nicht mehr die Übereinstimmung, wie bei den nicht gewichteten Koeffizienten, wird der Wert der Koeffizienten aus der beobachteten Nichtübereinstimmung $\bar{U}_b$ und der erwarteten Nichtübereinstimmung $\bar{U}_e$ berechnet:

$$\alpha, \kappa^g = 1 - \frac{\bar{U}_b}{\bar{U}_e^{\alpha,\kappa^g}} \tag{12}$$

Nichtübereinstimmung ist das Gegenteil von Übereinstimmung, sodass im Allgemeinen gilt: $\bar{U}_b = 1 - U_b$ und $\bar{U}_e = 1 - U_e$. Unter diesen Annahmen lässt sich zeigen, dass die Gleichungen 12 und 8 äquivalent sind [AP07]:

$$1 - \frac{\bar{U}_b}{\bar{U}_e} = 1 - \frac{1 - U_b}{1 - U_e} = \frac{1 - U_e - (1 - U_b)}{1 - U_e} = \frac{(U_b - U_e)}{1 - U_e} \quad \square$$

3.4.1 Metriken

Metriken sind Abbildungen, die die Distanz zwischen zwei Kategorien messen und so eine Aussage über den Grad der Nichtübereinstimmung machen [AP07].

Formal sei eine Metrik d wie folgt als eine Abbildung definiert:

$$d : \mathbf{K} \times \mathbf{K} \to [0, \infty) \subset \mathbb{R} \tag{13}$$

Anders als in der Mathematik üblich werden für diese Metriken je nach Koeffizient unterschiedliche weitere Bedingungen an die Abbildung gestellt, um als Metrik fungieren zu können. Alle hier beschriebenen Koeffizienten fordern jedoch $d(k, k) = 0 \quad \forall k \in \mathbf{K}$, um die beobachtete Nichtübereinstimmung korrekt berechnen zu können.

Krippendorff beschreibt in seinen Buch [Kri04, Abschnitt 11.3.4] verschiedene oft benutzte Metriken:

Nominalmetrik: Die Metrik ist definiert durch

$$d_n(k_i, k_j) = \begin{cases} 1 & \text{falls } k_i = k_j \\ 0 & \text{sonst} \end{cases} \tag{14}$$

Die Metrik eignet sich für Kategorien, für die sich kein messbarer Unterschied in der Nichtübereinstimmung für zwei beliebige Kategorien feststellen lässt. Ein Beispiel wären zum Beispiel verschiedene Telefonnummern, die sich entweder unterscheiden oder nicht. Für eine gegebene Telefonnummer lässt sich nicht sagen, ob diese Nummer mehr von der einen oder anderen Nummer abweicht.

Ordinalmetrik: Für die Ordinalmetrik muss eine Rangfolge der verwendeten Kategorien aufgestellt werden können[1]. Die Distanz zwischen zwei Kategorien wird dann gemessen als die Anzahl der Ränge zwischen diesen Kategorien gewichtet mit der Häufigkeit, mit der die Kategorien jeweils Elementen zugeordnet worden sind:

$$d_o(k_i, k_j) = \frac{n_{k_i}}{2} + \sum_{g>i}^{g<j} n_g + \frac{n_{k_j}}{2} \text{ für } k_i < k_j \tag{15}$$

Intervallmetrik: Für die Intervallmetrik muss nicht nur eine Rangfolge der verwendeten Kategorien aufgestellt werden können, sondern zusätzlich muss es möglich sein, zwei beliebige Kategorien subtrahieren zu können. In diesem Fall ist die Distanz zwischen zwei Kategorien definiert als die quadratische Differenz dieser beiden Kategorien:

$$d_i(k_i, k_j) = (k_i - k_j)^2 \tag{16}$$

Während zum Beispiel die Kategorien „Unzufrieden“, „Zufrieden“ und „Sehr Zufrieden“ geeignet sind, um mit der Ordinalmetrik die Abstände zu messen, sind sie mangels Möglichkeit der Subtraktion nicht geeignet, um mit der Intervallmetrik die Abstände zu messen. Dagegen wären unterschiedliche Temperaturwerte gemessen in Celsius geeignet, um sowohl mit der Ordinalmetrik als auch mit der Intervallmetrik die Abstände zu messen.

Verhältnismetrik. Die Verhältnismetrik ist definiert durch:

$$d_v(k_i, k_j) = (\frac{k_i - k_j}{k_1 + k_j})^2 \tag{17}$$

Auch bei dieser Metrik müssen die Kategorien in eine Rangfolge einzuordnen und zu subtrahieren sein. Zusätzlich muss die Null ein Referenzpunkt für die Kategorien sein. Die Idee dahinter ist, dass die Distanz im Verhältnis zum Abstand zur Null gemessen wird d. h. zwei verschiedene Kategorien, die nah an der Null liegen, haben gemäß der Metrik einen größeren Abstand als zwei Kategorien, die weiter weg von der Null liegen. Eine sinnvolle Anwendung der Verhältnismetrik wäre zum Beispiel die Schätzung des Alters von Personen. Während bei Kindern ein Jahr Differenz bereits ein großer Unterschied darstellt, ist dies bei Senioren üblicherweise nicht mehr so.

[1] Da die Kategorien am Anfang des Kapitels als Elemente der Menge der reellen Zahlen definiert worden sind, ist das Kriterium hier für alle Mengen von Kategorien erfüllt. In der Praxis ist jedoch auch die Verwendung eines anschaulichen Namens für eine Kategorie üblich.

3.4.2 Krippendorffs Alpha

Krippendorffs Alpha ist ein Koeffizient, der wie Scotts Pi und Fleiss Kappa auf der Annahme basiert, dass die erwartete Übereinstimmung abhängig ist von der Häufigkeit der einzelnen Kategorien insgesamt im Datensatz. Die folgende Beschreibung folgt derjenigen von Artstein und Poesio [AP07] und nicht der von Krippendorff selber [Kri04, Abschnitt 11.3], da bei der letzteren der Koeffizient über sogenannte Koinzidenzmatrizen definiert ist und dies die Implementierung in ein Programm unnötig erschwert im Vergleich zu der Koinzidenzmatrizen freien Definition von Artstein und Poesio.

Ähnlich wie bei der beobachteten Übereinstimmung für mehr als zwei Codierer wird für die beobachtete Nichtübereinstimmung die Idee verfolgt je Element die einzelnen Paare unterschiedlicher Urteile, gewichtet nach dem Grad der Nichtübereinstimmung, zu betrachten. Die Anzahl der Paare, für ein bestimmtes Element e_h und die beiden unterschiedlichen Kategorien k_i und k_j ergibt sich aus $n_{hi}n_{hj}$. Wie bei der beobachteten Übereinstimmung, ist die Anzahl aller Paare gleich $NC(C-1)$. Die beobachtete Nichtübereinstimmung ist daher definiert als

$$\bar{U}_b = \frac{1}{NC(C-1)} \sum_{h=1}^{N} \sum_{k_i \in \mathbf{K}} \sum_{k_j \in \mathbf{K}} n_{hi} n_{hj} d(k_i, k_j) \tag{18}$$

Die erwartete Nichtübereinstimmung ist gegeben durch:

$$\bar{U}_e^{\kappa} = \frac{1}{NC(NC-1)} \sum_{k_i \in \mathbf{K}} \sum_{k_j \in \mathbf{K}} n_{k_i} n_{k_j} d(k_i, k_j) \tag{19}$$

Da hier die Nichtüberstimmung berechnet wird, im Gegensatz zu der Berechnung von Übereinstimmung bei Fleiss Kappa, wird hier über alle Paare von Kategorien aufsummiert und nicht nur über die Kategoriepaare mit zweimal der gleichen Kategorie. Zusätzlich unterscheidet sich die Formel von $1 - U_e^{\pi^{C>2}}$ um den Faktor $\frac{NC-1}{NC}$. Dieser Korrekturfaktor soll für Datensätze mit wenigen Elementen oder Kategorien folgende Verzerrung der Schätzung von erwarteter Übereinstimmung bzw. Nichtübereinstimmung verhindern: Die erwartete Übereinstimmung bzw. Nichtübereinstimmung wird basierend auf einem einzigen Datensatz geschätzt. Die Elemente eines Datensatzes neigen dazu, sich nicht so stark zu unterscheiden als die Elemente verschiedener Datensätze. Die erwartete Übereinstimmung sollte aber eigentlich per Definition auf die Wahrscheinlichkeit $P(k)$ basieren d. h. die Wahrscheinlichkeit des Auftretens einer Kategorie unabhängig vom konkreten Datensatz. Tatsächlich wird $P(k)$ aber nur abgeschätzt, was zu einer Abweichung zur wahren Wahrscheinlichkeit $P(k)$ führt. Mutmaßlich ist die Abweichung um so größer je weniger Elemente ein Datensatz enthält.

Für große N oder C konvergiert der Korrekturfaktor gegen eins d. h., wenn die Nominalmetrik verwendet wird, dann wird der Unterschied zwischen Fleiss Kappa und Krippendorffs Alpha immer kleiner, desto mehr Codierer oder Elemente der Datensatz enthält.

Für die Berechnung von Krippendorffs Alpha mit der Intervallmetrik für den Beispielsatz ist zuerst die beobachtete Nichtübereinstimmung zu berechnen. Für die Metrik ergeben

sich folgende Werte: $d_i(1,2) = (1-2)^2 = 1 = d_i(2,1)$, $d_i(1,3) = (1-3)^2 = 4 = d_i(3,1)$, $d_i(2,3) = (2-3)^2 = 1 = d_i(3,2)$ und $d_i(k,k) = (k-k)^2 = 0 \quad \forall k \in \mathbf{K}$. Es sei

$$x_h = \sum_{k_i \in \mathbf{K}} \sum_{k_j \in \mathbf{K}} n_{hi} n_{hj} d(k_i, k_j)$$

der Wert für das h-te Element in der Formel zur Bestimmung der beobachteten Nichtübereinstimmung. Für eine übersichtlichere Darstellung ist die Berechnung der x_j in Tabelle 8 zu finden. Die Summanden für die Paare mit zweimal der gleichen Kategorie wurden weggelassen, da sie durch die Metrik immer null sind.

Element	Berechnung
1	$x_1 = 3 \cdot 0 \cdot 1 + 3 \cdot 0 \cdot 4 + 0 \cdot 3 \cdot 1 + 0 \cdot 0 \cdot 1 + 0 \cdot 3 \cdot 4 + 0 \cdot 0 \cdot 1 = 0$
2	$x_2 = 0 \cdot 3 \cdot 1 + 0 \cdot 0 \cdot 4 + 3 \cdot 0 \cdot 1 + 3 \cdot 0 \cdot 1 + 0 \cdot 0 \cdot 4 + 0 \cdot 3 \cdot 1 = 0$
3	$x_3 = 0 \cdot 0 \cdot 1 + 0 \cdot 3 \cdot 4 + 0 \cdot 0 \cdot 1 + 0 \cdot 3 \cdot 1 + 3 \cdot 0 \cdot 4 + 3 \cdot 0 \cdot 1 = 0$
4	$x_4 = 2 \cdot 1 \cdot 1 + 2 \cdot 0 \cdot 4 + 1 \cdot 2 \cdot 1 + 1 \cdot 0 \cdot 1 + 0 \cdot 2 \cdot 4 + 0 \cdot 1 \cdot 1 = 4$
5	$x_5 = 0 \cdot 3 \cdot 1 + 0 \cdot 0 \cdot 4 + 3 \cdot 0 \cdot 1 + 3 \cdot 0 \cdot 1 + 0 \cdot 0 \cdot 4 + 0 \cdot 3 \cdot 1 = 0$
6	$x_6 = 0 \cdot 1 \cdot 1 + 0 \cdot 2 \cdot 4 + 1 \cdot 0 \cdot 1 + 1 \cdot 2 \cdot 1 + 2 \cdot 0 \cdot 4 + 2 \cdot 1 \cdot 1 = 4$
7	$x_7 = 2 \cdot 1 \cdot 1 + 2 \cdot 0 \cdot 4 + 1 \cdot 2 \cdot 1 + 1 \cdot 0 \cdot 1 + 0 \cdot 2 \cdot 4 + 0 \cdot 1 \cdot 1 = 4$
8	$x_8 = 0 \cdot 3 \cdot 1 + 0 \cdot 0 \cdot 4 + 3 \cdot 0 \cdot 1 + 3 \cdot 0 \cdot 1 + 0 \cdot 0 \cdot 4 + 0 \cdot 3 \cdot 1 = 0$
9	$x_9 = 0 \cdot 3 \cdot 1 + 0 \cdot 0 \cdot 4 + 3 \cdot 0 \cdot 1 + 3 \cdot 0 \cdot 1 + 0 \cdot 0 \cdot 4 + 0 \cdot 3 \cdot 1 = 0$
10	$x_{10} = 0 \cdot 1 \cdot 1 + 0 \cdot 2 \cdot 4 + 1 \cdot 0 \cdot 1 + 1 \cdot 2 \cdot 1 + 2 \cdot 0 \cdot 4 + 2 \cdot 1 \cdot 1 = 4$

Tabelle 8: Berechnung der Zwischenergebnisse je Element für die beobachtete Nichtübereinstimmung des Beispieldatensatzes

Für die beobachtete Nichtübereinstimmung gilt:

$$\begin{aligned}
\bar{U}_b =& \frac{1}{NC(C-1)} \sum_{h=1}^{N} \sum_{k_i \in \mathbf{K}} \sum_{k_j \in \mathbf{K}} n_{hi} n_{hj} d(k_i, k_j) \\
=& \frac{x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10}}{NC(C-1)} \\
=& \frac{0+0+0+4+0+4+4+0+0+4}{10 \cdot 3(3-1)} \\
=& \frac{16}{60} = \frac{4}{15} \approx 0.2667
\end{aligned}$$

Und für die erwartete Nichtübereinstimmung gilt:

$$\begin{aligned}
\bar{U}_e^{\kappa} =& \frac{1}{NC(NC-1)} \sum_{k_i \in \mathbf{K}} \sum_{k_j \in \mathbf{K}} n_{k_i} n_{k_j} d(k_i, k_j) \\
=& \frac{6 \cdot 6 \cdot 0 + 6 \cdot 17 \cdot 1 + 6 \cdot 7 \cdot 4 + 17 \cdot 6 \cdot 1 + 17 \cdot 17 \cdot 0 + 17 \cdot 7 \cdot 1}{10 \cdot 3(10 \cdot 3 - 1)} \\
& + \frac{7 \cdot 6 \cdot 4 + 7 \cdot 17 \cdot 1 + 7 \cdot 7 \cdot 0}{10 \cdot 3(10 \cdot 3 - 1)} \\
=& \frac{785}{870} = \frac{157}{174} \approx 0.9028
\end{aligned}$$

Insgesamt ergibt das:

$$\alpha = 1 - \frac{\bar{U}_b}{\bar{U}_e^{\alpha}} = 1 - \frac{\frac{4}{15}}{\frac{157}{174}} = \frac{553}{785} \approx 0.7045$$

3.4.3 Erweiterung für Cohens Kappa

Artstein und Poesio stellen eine Erweiterung als gewichteten Koeffizienten für Cohens Kappa und mehr als zwei Codierer vor. Die beobachtete Nichtübereinstimmung wird genauso berechnet wie bei Krippendorffs Alpha.

Die Wahrscheinlichkeit für eine Kategorie k gegeben ein Codierer c wird, wie bei Cohens Kappa, geschätzt durch $P'(k|c) = \frac{n_{ck}}{N}$. Die Wahrscheinlichkeit, dass zwei bestimmte Codierer c_h und c_l einem Element zwei verschiedene Kategorien k_i und k_j zuordnen ist $P'(k_i|c_h)P'(k_j|c_l) + P'(k_j|c_h)P'(k_i|c_j)$. Die Wahrscheinlichkeit für zwei beliebige Codierer ist dann der Durchschnitt über die Anzahl der Codiererpaare:

$$\frac{1}{\binom{C}{2}} \sum_{h=1}^{C-1} \sum_{l=h+1}^{C} P'(k_i|c_h)P'(k_j|c_l) + P'(k_j|c_h)P'(k_i|c_j) = \frac{1}{N^2\binom{C}{2}} \sum_{h=1}^{C-1} \sum_{l=h+1}^{C} n_{c_h i} n_{c_l j} + n_{c_h j} n_{c_l i}$$

Für die erwartete Übereinstimmung wird jetzt noch der Durchschnitt über alle Kategoriepaare gebildet und mit der Distanzfunkion gewichtet:

$$\begin{aligned}
\bar{U}_e^{\kappa^g} =& \frac{1}{N^2\binom{C}{2}} \sum_{k_i \in \mathbf{K}\setminus\{k_K\}} \sum_{k_j \in \mathbf{K}\setminus\{k_i\}} \sum_{h=1}^{C-1} \sum_{l=h+1}^{C} n_{c_h i} n_{c_l j} d(k_i, k_j) + n_{c_h j} n_{c_l i} d(k_i, k_j) \\
=& \frac{1}{N^2\binom{C}{2}} \sum_{k_i \in \mathbf{K}} \sum_{k_j \in \mathbf{K}} \sum_{h=1}^{C-1} \sum_{l=h+1}^{C} n_{c_h i} n_{c_l j} d(k_i, k_j)
\end{aligned}$$

3.5 Ein Koeffizient für Unitizing: Krippendorffs α_U

α_U ist ein von Krippendorff entwickelter Koeffizient für das Unitizing [Kri95].

Wie in Kapitel 2 erläutert, besteht beim Unitizing ein Datensatz aus kleinsten unterscheidbaren Einheiten (kuE) und einer Menge von Abschnitten $\mathbf{A}$. Die Menge der Abschnitte ist die Vereinigung der Abschnittsmengen aller Kategorien: $\mathbf{A} = \bigcup_{k \in \mathbf{K}} \mathbf{A}_k$. Die Menge der Abschnitte einer Kategorie wiederrum ist die Vereinigung der Abschnitte aller Codierer dieser Kategorie: $\mathbf{A}_k = \bigcup_{c \in \mathbf{C}} \mathbf{A}_{kc}$. Für einen beliebigen Abschnitt a bezeichne l_a die Länge dieses Abschnittes in kuE, b_a die Stelle in kuE im Datensatz an der a beginnt, d. h. der Startpunkt, und die Funktion u nimmt für a den Wert eins an, falls a eine Einheit ist und sonst null. L bezeichne die Gesamtlänge in kuE des Datensatzes.

α_U wird analog zu Krippendorff Alpha wie folgt berechnet:

$$\alpha_U = 1 - \frac{\sum_{k \in \mathbf{K}} \bar{U}_b^k}{\sum_{k \in \mathbf{K}} \bar{U}_e^k} \tag{20}$$

Es wird hier die beobachtete und erwartete Nichtübereinstimmung für jede Kategorie separat berechnet und dann aufsummiert.

Zwei Abschnitte gleicher Kategorie, aber von zwei verschiedenen Codierern, stimmen genau dann überein, wenn die beiden Abschnitte die gleiche Länge haben und an der gleichen Stelle beginnen. Wie stark zwei Abschnitte nicht übereinstimmen wird mit der Funktion d gemessen, die wie folgt definiert ist:

$$d(a_i, a_j) = \begin{cases} (b_{a_i} - b_{a_j})^2 + (b_{a_i} + l_{a_i} - b_{a_j} + l_{a_j})^2 & \text{falls } \star^1 \\ l_{a_i}^2 & \text{falls } \star^2 \\ l_{a_j}^2 & \text{falls } \star^3 \\ 0 & \text{sonst} \end{cases} \tag{21}$$

$$\begin{aligned} \star^1 \quad & u(a_i) = u(a_j) = 1 \wedge -l_{a_i} < b_{a_i} - b_{a_j} < l_{a_j} \\ \star^2 \quad & u(a_i) = 1 \wedge u(a_j) = 0 \wedge l_{a_j} - l_{a_i} \geq b_{a_i} - b_{a_j} \geq 0 \\ \star^3 \quad & u(a_i) = 0 \wedge u(a_j) = 1 \wedge l_{a_j} - l_{a_i} \leq b_{a_i} - b_{a_j} \leq 0 \end{aligned}$$

Im ersten Fall überlappen die beiden Abschnitte. In diesem Fall wird das Ausmaß der Nichtübereinstimmung gemessen als die Summe der quadrierten Länge die links überlappt und der quadrierten Länge die rechts überlappt. Im zweiten Fall ist der erste Abschnitt eine Einheit, die zweite nicht und die Einheit ist komplett vom zweiten Abschnitt überdeckt. Genau umgekehrt ist es im dritten Fall. Im letzten Fall stimmen die beiden Abschnitte genau überein, sind beides Abschnitte oder haben keine Gemeinsamkeiten.

Mit der Funktion d wird die beobachtete Übereinstimmung für eine bestimmte Kategorie k berechnet, indem für jeden Abschnitt der Kategorie die Unterschiede zu jedem anderen Abschnitt dieser Kategorie und der anderen Codierer berechnet wird und die Unterschiede aufsummiert werden. Anschließend wird die Gesamtsumme durch das mögliche Maximum der Gesamtsumme geteilt. Das Maximum wird berechnet als das Produkt aus der Anzahl der Codiererpaare und dem Quadrat der Datensatzlänge:

$$\bar{U}_b^k = \frac{1}{C(C-1)L^2} \sum_{c \in \mathbf{C}} \sum_{a_j \in \mathbf{A}_{kc}} \sum_{a_h \in \mathbf{A_k} \setminus \mathbf{A}_{kc}} d(a_j, a_h) \tag{22}$$

Für die Berechnung der erwarteten Übereinstimmung einer Kategorie wird der Unterschied zwischen allen möglichen Kombinationen der im Datensatz bestehenden Einheiten und der Lücken zwischen den Einheiten berechnet. Auf Grund der komplizierten Berechnung aller möglichen Kombinationen gibt Krippendorff folgende Formel zur Berechnung der erwarteten Übereinstimmung an und beweist, dass diese Formel korrekt

ist. Bei dieser Formel ist es nicht nötig alle Kombinationen und die Unterschiede direkt zu berechnen.

$$\bar{U}_e^k = \frac{\frac{1}{L}\sum\limits_{a_i \in \mathbf{A}_k} u(a_i)\left[\frac{A_k^u - 1}{3}\left(2l_{a_i}^3 - 3l_{a_i}^2 + l_{a_i}\right) + l_{a_i}^2 \sum\limits_{a_j \in \mathbf{A}_k} \overbrace{(1-u(a_j))(l_{a_j} - l_{a_i} + 1)}^{\text{falls } l_{a_j} \geq l_{a_i}}\right]}{CL(CL-1) - \sum\limits_{a_i \in \mathbf{A}_k} u(a_i) l_{a_i}(l_{a_i} - 1)} \tag{23}$$

mit $A_k^u = \sum\limits_{a_i \in \mathbf{A}_k} u(a_i)$ wird die Anzahl aller Einheiten einer Kategorie bezeichnet.

4 Ablauf der Simulationen

Diese Kapitel dient der Darstellung der allgemeinen Konzepte des entwickelten Simulationsprogramms an Hand des Ablaufs einer Simulation. In Abbildung 1 ist der Ablauf einer Simulation dargestellt.

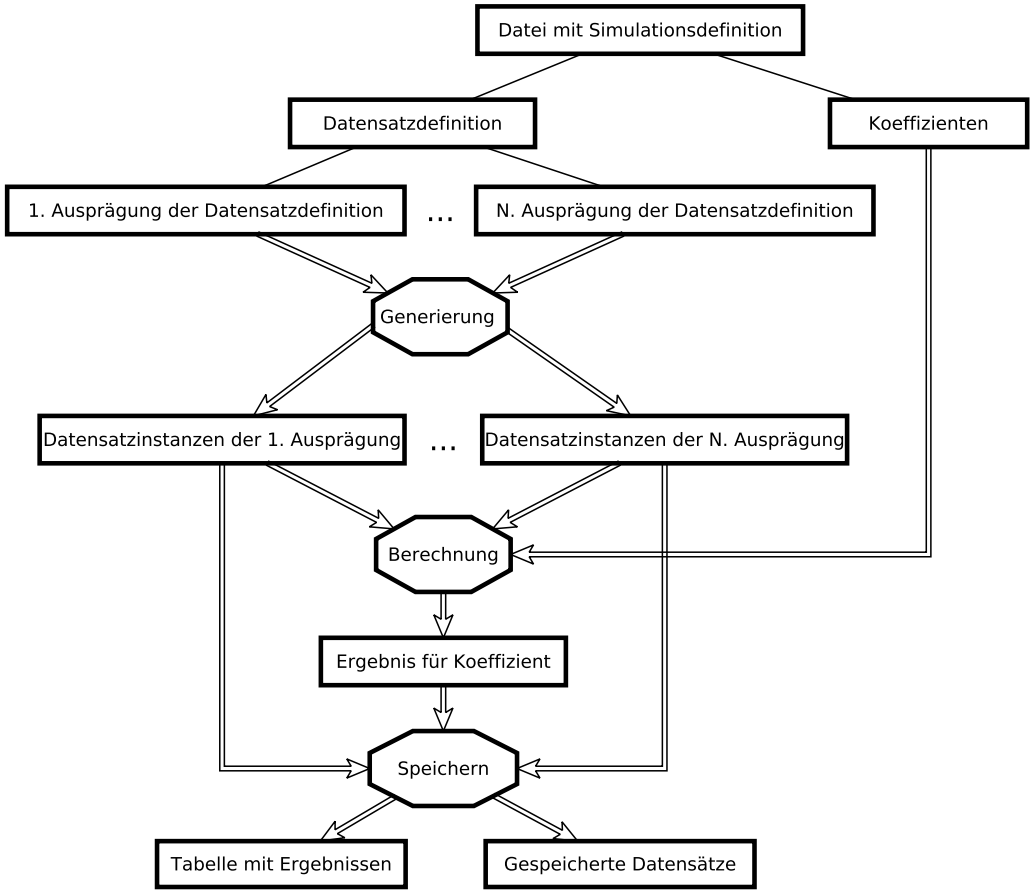

Abbildung 1: Schematische Darstellung des Ablaufs einer Simulation

4.1 Simulationsdefinition

Für die Ausführung einer Simulation muss der Benutzer zuerst definieren, was simuliert werden soll. Diese Simulationsdefinition schreibt er in eine Datei, die dann später durch das Simulationsprogramm eingelesen wird. Eine Simulationsdatei besteht aus zwei Teilen: Der eine Teil besteht aus der Definition der Datensätze und der andere Teil aus einer Liste mit den zu simulierenden Koeffizienten.

Die Datensatzdefinition besteht aus mehreren Variablen (zum Beispiel die Variable „Anzahl Elemente“), die als Parameter die zu generierenden Datensätze festlegen. Eine Variable kann mehrere durch den Benutzer festgelegte Werte, genannt Ausprägungen, anneh-

men – im Falle der Variable „Anzahl Elemente“ beispielsweise 100 und 200. Jede Kombination der verschiedenen Ausprägungen der Variablen bilden zusammen eine Ausprägung der Datensatzdefinition (Datensatzausprägung). Für jede dieser Ausprägung wird durch das Programm eine oder mehrere Instanzen der Ausprägung generiert. Eine Instanz ist ein Datensatz im Sinne des Kapitels zwei und drei, auf dem der Wert eines Koeffizienten berechnet werden kann.

Wenn es zum Beispiel die Variable „Anzahl Elemente“ mit den Ausprägungen 100 und 200 und die Variable „Anzahl Codierer“ mit den Ausprägungen zwei und drei gibt, dann würden vier verschiedenen Ausprägungen der Datensatzdefinition gebildet: Eine Ausprägung mit 100 Elementen und zwei Codierern, eine mit 100 Elementen und drei Codierern, eine mit 200 Elementen und zwei Codierern und eine Ausprägung mit 200 Elementen und drei Codierern. Für jede dieser Datensatzausprägungen würden dann eine oder mehrere Instanzen generiert. Diese Instanzen einer Ausprägung würden sich dann nicht in der Anzahl der Elemente und der Anzahl an Codierer unterscheiden, sondern beispielsweise in der Anzahl, wie oft der erste Codierer den Elementen eine bestimmte Kategorie zugeordnet hat.

Die Ausprägungen dienen der Analyse des Verhaltens eines Koeffizienten. Wenn zum Beispiel der Benutzer analysieren möchte, wie die Werte des Koeffizienten sich für eine steigende Anzahl an Codierern entwicklen, dann wird er die Variable „Anzahl Codierer“ mit den Ausprägungen 100 und 200 ausstatten.

4.2 Generierung der Datensätze

Bei der Generierung kann zum einen unterschieden werden zwischen Codierung und Unitizing und zum anderen zwischen den Methoden mit der die Datensätze generiert werden. Das Programm unterstützt zwei Methoden: Bei der Generierung durch vollständige Übereinstimmung wird zuerst ein Datensatz generiert, bei dem die Codierer vollständig übereinstimmen, d. h. im Codierungsfall ordnen die Codierer allen Elementen die selbe Kategorie zu. Anschließend werden zufällig einzelne Elemente bzw. Einheiten verändert und die Übereinstimmung so verschlechtert. Dagegen wird bei der Generierung durch Einlesen eines bereits generierten Datensatzes zuerst ein Datensatz, der durch eine vorhergehende Simulation gewonnen wurde oder durch den Benutzer erstellt worden ist, eingelesen und dann durch Kopieren der bereits vorhandenen Daten vergrößert oder durch Löschen verkleinert.

4.2.1 Generierung durch vollständige Übereinstimmung (Codierung)

In Tabelle 9 sind die Variablen, die der Benutzer für die Generierung eines Datensatzes durch vollständige Übereinstimmung im Falle von Codierung zur Verfügung hat, erläutert.

Der erste Schritt der Erzeugung von vollständiger Übereinstimmung sieht wie folgt aus: Es werden solange Elemente dem Datensatz hinzugefügt bis die passende Anzahl gemäß der entsprechenden Variable erreicht ist. Für jedes neue Element wird zufällig eine Kategorie aus Menge der Kategorien bestimmt gemäß der gegebenen Verteilung. Diese Kategorie wird dann für alle Codierer verwendet.

Name	Wertebereich	Erläuterung
Anzahl Codierer	$\mathbb{Z}_{>1}$	Anzahl der Kategorien im generierten Datensatz
Anzahl Elemente	$\mathbb{Z}_{>0}$	Anzahl der Elemente im Datensatz
Anzahl Kategorien	$\mathbb{Z}_{>0}$	Anzahl der Kategorien im Datensatz
Wahrscheinlichkeitsverteilung der Kategorien	Liste mit Zahlen im Bereich $[0, 1] \subset \mathbb{R}$	Liste mit den Wahrscheinlichkeiten, dass bei der Generierung der vollständigen Übereinstimmung für ein bestimmtes Element und einem bestimmten Codierer eine bestimmte Kategorie zugeordnet wird. Das erste Element der Liste ist die Wahrscheinlichkeiten für die erste Kategorie usw.
Wahrscheinlichkeit für Änderung eines Elementes	$[0, 1] \subset \mathbb{R}$	Wahrscheinlichkeit, dass ein Element im zweiten Schritt geändert wird.
Wahrscheinlichkeiten für die Änderung der Kategoriezuordnung für die Codierer	Liste mit Zahlen im Bereich $[0, 1] \subset \mathbb{R}$	Liste mit den Wahrscheinlichkeiten für die Änderung der Kategoriezuordnung des entsprechenden Codierers für die Elemente, die im zweiten Schritt geändert werden sollen. Das erste Element der Liste ist die Wahrscheinlichkeit für eine Änderung beim ersten Codierer usw.

Tabelle 9: Variablen für die Generierung durch vollständige Übereinstimmung (Codierung).

Im zweiten Schritt werden zufällig Elemente aus dem im ersten Schritt erzeugten Datensatz ausgewählt, die geändert werden sollen. Die Wahrscheinlichkeit, dass ein bestimmtes Element tatsächlich geändert wird, ergibt sich aus der entsprechenden Variable. Für jedes Element, das geändert werden soll, werden wiederum zufällig Zuordnungen einzelner Codierer ausgewählt, denen eine andere Kategorie zugewiesen wird. Die entsprechende Variable gibt die Wahrscheinlichkeit für die einzelnen Codierer an, dass sich ihre Zuordnung ändert.

In Tabelle 10 ist ein generierter Datensatz nach dem ersten Schritt dargestellt. In diesem Fall sollte es zwei Codierer und drei Kategorien geben. Der Datensatz sollte zehn Elemente enthalten. Die Wahrscheinlichkeitsverteilung sei wie folgt: 0.6 für Kategorie Eins, 0.2 für Kategorie Zwei und 0.2 für Kategorie Drei. Dementsprechend sind auch sechs Elemente Kategorie Eins zugeordnet, jeweils zwei Elemente den Kategorien Zwei und Drei.

	1	2	3	4	5	6	7	8	9	10
Codierer 1	1	2	3	2	1	3	1	1	1	1
Codierer 2	1	2	3	2	1	3	1	1	1	1

Tabelle 10: Beispieldatensatz für vollständige Übereinstimmung (Codierung/1. Schritt)

Das Ergebnis nach dem zweiten Schritt ist in Tabelle 11 dargestellt. Die Wahrscheinlichkeit für die Änderung eines Elements sei in diesem Beispiel 0.4, daher sind in diesem Beispiel vier Elemente gelb markiert, um zu symbolisieren, dass diese zur Änderung ausgewählt worden sind. Weiter sei die Wahrscheinlichkeit für eine Änderung der Kategoriezuordnung für ein ausgewähltes Element für Codierer Eins 0.5 und für Codierer Zwei 0.75. Aus diesem Grund stehen bei den ausgewählten Elementen an zwei Stellen bei Codierer Eins eine andere Kategorie als in Tabelle 10 (Element Eins und Zehn) und bei Codierer Zwei an drei Stellen (Element Eins, Drei und Zehn).

	1	2	3	4	5	6	7	8	9	10
Codierer 1	3	2	3	2	1	3	1	1	1	2
Codierer 2	2	2	1	2	1	3	1	1	1	2

Tabelle 11: Beispieldatensatz für vollständige Übereinstimmung (Codierung/2. Schritt)

4.2.2 Generierung durch vollständige Übereinstimmung (Unitizing)

Die Variablen für die Generierung eines Datensatzes durch vollständige Übereinstimmung im Falle von Unitizing sind in Tabelle 12 erläutert.

Im ersten Schritt werden für jede Kategorie, die der Datensatz den entsprechenden Variablen nach enthalten soll, wie folgt dem Datensatz Einheiten hinzugefügt: Die Anzahl der Einheiten ergibt sich aus der passenden Variable. Die Länge des Datensatzes wird durch diese Anzahl geteilt. Dies ergibt den Abstand zwischen den Mittelpunkten der Einheiten einer Kategorie. Der Mittelpunkt der ersten Einheit hat die Hälfte des Abstands zwischen den Mittelpunkten Abstand vom Beginn des Datensatzes. Für die gleiche Kategorie ist bei jedem Codierer die Anzahl der Einheiten und die Länge der Einheiten gleich. Dieses Verfahren ergibt eine gleichmäßige Verteilung der Einheiten. Daraus folgt, dass für die gleiche Kategorie bei jedem der Codierer gleich viele Einheiten mit gleichem Mittelpunkt hinzugefügt werden, was eine vollständige Übereinstimmung ergibt. In der Abbildung 2 ist beispielhaft ein Ergebnis aus dem ersten Schritt für eine Kategorie dargestellt. Zwei Einheiten mit der Länge 100 sind zu erstellen. Der Datensatz hat insgesamt eine Länge von 300. 300 durch Zwei ergibt 150. Die Mittelpunkte sind daher jeweils 150 kuE[2] voneinander entfernt.

Für jeden Codierer und jede Kategorie werden die entsprechenden Einheiten im zweiten Schritt nacheinander durchgegangen und zufällig entschieden, ob die Einheit verändert wird oder nicht. Wenn eine Einheit verändert wird, wird die Einheit verschoben und/oder die Länge der Einheit verkleinert oder vergrößert, gemäß der beiden Variablen. Die Ausprägungen dieser Variablen sind Listen von Werten. Dabei werden die Elemente der Liste über die Codierergrenzen hinweg verwendet. Wenn es zum Beispiel zwei Kategorien und zwei Codierer gibt, dann werden bei vier Werten in der Liste der erste Werte bei der ersten Kategorie und dem ersten Codierer verwendet, der zweite Wert bei den Einheiten der zweiten Kategorie des ersten Codierers, dritte Wert für die Einheiten der ersten Kategorie des zweiten Codierers und der vierte Wert bei den Einheiten der

[2] kleinste unterscheidbare Einheit

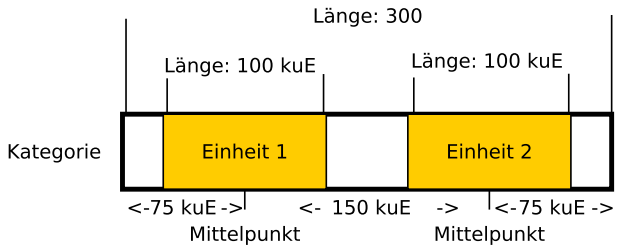

Abbildung 2: Darstellung des ersten Schritts

zweiten Kategorie des zweiten Codierers. Eine Verkleinerung der Einheit auf die Länge Null löscht die Einheit. Der zweite Schritt kann mehr als einmal durchgeführt werden. Wenn der zweite Schritt nur einmal durchgeführt würde, wären alle zur Änderung ausgewählte Einheiten gleich stark verschoben und verkleinert bzw. vergrößert worden. Wenn der zweite Schritt mehrmals ausgeführt wird, dann sind immer mehr Einheiten unterschiedlich stark verschoben und verkleinert bzw. vergrößert, was mehr dem entspricht, was bei einer Annotation durch einen Menschen zu erwarten wäre.

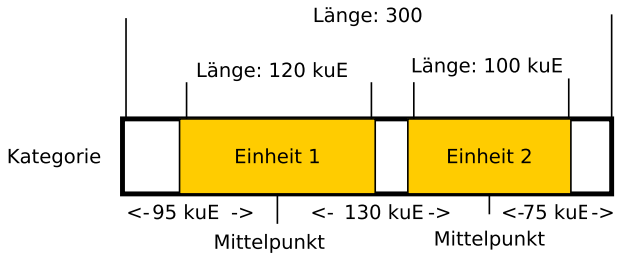

Abbildung 3: Darstellung des zweiten Schritts

In Abbildung 3 ist das obige Beispiel nach Schritt Zwei dargestellt. Die erste Einheit ist geändert worden und zwar 20 kuE nach rechts verschoben und um 20 kuE vergrößert.

4.2.3 Generierung durch Einlesen eines bereits generierten Datensatzes

Bei der Generierung eines Datensatzes durch Einlesen eines bereits generierten Datensatzes gibt es im Codierungsfall die beiden Variablen „Anzahl der Codierer“ und „Anzahl der Elemente“. Der Benutzer muss einen Ordner im Dateisystem angeben, in der die bereits generierten Datensätze gespeichert sind. Diese Datensätze werden vom Programm eingelesen. Es werden so viele Datensätze eingelesen, wie Datensatzinstanzen generiert werden sollen. Ist die vom Benutzer geforderte Anzahl an Instanzen höher als die Anzahl der im Ordner gespeicherten, dann werden die gespeicherten Datensätze wiederholt eingelesen, bis die gewünschte Anzahl erreicht ist. Die eingelesenen Datensätze werden dann so angepasst, dass die Anzahl der Codierer und die Anzahl der Elemente den Aus-

prägungen der entsprechenden Variablen entsprechen. Wenn die Anzahl der Codierer höher liegen soll als sie beim eingelesenen Datensatz ist, dann werden jedem Element die fehlenden Kategoriezuordnungen hinzugefügt, indem so viele Zuordnungen wie nötig der vorhandenen Codierer kopiert werden. Soll die Anzahl der Codierer niedriger sein, werden die Zuordnungen, die zu viel sind, gelöscht. Ebenso werden so viele Elemente wie nötig vom vorhandenen Datensatz dem zu generierenden Datensatz hinzugefügt bis die gewünschte Anzahl erreicht ist. Wenn die Anzahl der Elemente niedriger liegen soll als beim eingelesenen Datensatz, dann werden, beginnend mit dem letzten Element, die Elemente, die zu viel sind, gelöscht.

Im Unitizing-Fall gibt es ebenfalls die Variable „Anzahl der Codierer“ und zusätzlich „Anzahl der Kategorien“ und „Faktor der Datensatzlänge“. Wie im Codierungs-Fall werden bereits generierte Datensätze eingelesen und dann entsprechend der Variablen geändert. Die Anzahl der Codierer wird vergrößert, indem für jeden hinzuzufügenden Codierer die Einheiten eines bereits existierenden Codierers kopiert werden. Für eine Verringerung der Anzahl, werden die Einheiten so vieler Codierer wie nötig gelöscht. Analog wird die Anzahl der Kategorien vergrößert oder verkleinert. Zusätzlich kann der Benutzer über die Variable „Faktor der Datensatzlänge“ bestimmen, um welchen Faktor der Datensatz vergrößert werden soll. Es sind nur ganzzahlige Faktoren erlaubt, d. h. die Länge des Datensatzes kann verdoppelt, verdreifacht usw. werden. Auch hier wird der eingelesene Datensatz für die Vergrößerung kopiert und zwar so oft, wie durch den Wert des Faktors angegeben.

4.3 Berechnung der Koeffizienten

Für jede Datensatzausprägung findet ein Simulationsdurchlauf statt. Je Simulationsdurchlauf wird für jede Datensatzinstanz der Datensatzausprägung die Werte der Koeffizienten berechnet. Die Berechnung erfolgt wie in Kapitel drei beschrieben. Der Benutzer legt in der Simulationsdefinition fest, welche Koeffizienten berechnet werden sollen.

4.4 Speicherung der Berechnungsergebnisse und der generierten Datensätze

Nach der Berechnung können die generierten Datensatzinstanzen auf den Wunsch des Benutzers gespeichert werden und zum Beispiel in einer weiteren Simulation als Grundlagen für die Generierung neuer Datensätze dienen. Die Berechnungsergebnisse werden wie folgt gespeichert: Pro Koeffizient und eventuellen Zwischenergebnissen wird jeweils eine Tabelle gespeichert. Das bedeutet, wenn beispielsweise ein Koeffizient zwei Zwischenergebnisse berechnet, dann werden insgesamt drei Tabellen für diesen Koeffizienten gespeichert – zwei für die Zwischenergebnisse und eine für das Gesamtergebnis. Jede Datensatzausprägung wird repräsentiert durch eine Spalte und in den Zeilen stehen die berechneten Ergebnisse der Instanzen. Die gespeicherten Tabellen können dann vom Benutzer zur weiteren Analyse zum Beispiel mit einem Tabellenkalkulationsprogramm verwendet werden. Auch das Visualisieren durch ein Boxplot, wie im sechsten Kapitel dieser Arbeit, ist möglich.

Name	Wertebereich	Erläuterung
Anzahl Codierer	$\mathbb{Z}_{>1}$	Anzahl der Kategorien im generierten Datensatz
Anzahl Einheiten	$\mathbb{Z}_{>0}$	Anzahl der Einheiten im Datensatz
Anzahl Kategorien	$\mathbb{Z}_{>0}$	Anzahl der Kategorien im Datensatz
Länge Datensatz	$\mathbb{Z}_{>0}$	Länge des Datensatzes in kuE
Anzahl an Generierungsrunden	$\mathbb{Z}_{>0}$	Anzahl, wie oft Schritt 2 wiederholt wird.
Vergrößerungsfaktor	Liste mit Zahlen aus $\mathbb{R}$	Liste mit Faktoren, wie stark im zweiten Schritt die Einheiten vergrößert werden. Das erste Element der Liste ist der Faktor für die erste Kategorie und den ersten Codierer, das zweite Element ist der Faktor für die zweite Kategorie und den ersten Codierer usw.
Verschiebungswerte	Liste mit Zahlen aus $\mathbb{Z}$	Liste mit Werten, um wie viele kuE eine Einheit im zweiten Schritt nach rechts oder links verschoben wird. Das erste Element der Liste ist der Wert für die erste Kategorie und den ersten Codierer, das zweite Element ist der Faktor für die zweite Kategorie und den ersten Codierer usw.
Wahrscheinlichkeiten für die Änderung einer Einheit	Liste mit Zahlen im Bereich $[0, 1] \subset \mathbb{R}$	Liste mit den Wahrscheinlichkeiten für die Änderung (Verschiebung oder Änderung der Länge) einer Einheit im zweiten Schritt. Das erste Element der Liste ist die Wahrscheinlichkeit für die erste Kategorie und den ersten Codierer, das zweite Element ist der Faktor für die zweite Kategorie und den ersten Codierer usw.
Längen der Einheiten	Liste mit Zahlen im Bereich $\mathbb{Z}_{>0}$	Liste mit Längen der Einheiten in kuE im ersten Schritt. Es kann für jede Kategorie eine andere Länge festgelegt werden. Das erste Element der Liste ist die Länge für die erste Kategorie usw.

Tabelle 12: Variablen für die Generierung durch vollständige Übereinstimmung (Unitizing).

5 Implementierung

In diesem Kapitel wird die Implementierung des Simulationsprogramms beschrieben.

Das Simulationsprogramm ist in der Programmiersprache C# geschrieben und verwendet das .NET Framework als Klassenbibliothek. Das Simulationsprogramm ist in zwei Teile gegliedert: dem Simulationsframework[3] und einem Konsolenprogramm[4]. Das Simulationsframework besteht aus der Kernbibliothek (Core) und der Simulationsbibliothek (Simulation). Entsprechend sind auch zwei Projekte angelegt. Hinzu kommen noch zwei Projekte für das Unit Testing. Für die gesamte Implementierung des Simulationsframeworks ist ein objektorientierter Ansatz gewählt worden.

5.1 Kernbibliothek

Die Kernbibliothek besteht aus dem Namespace „Danlaps.InterRaterAgreement“ und enthält Klassen zur Repräsentation der Datensätze und der Berechnung der einzelnen Koeffizienten.

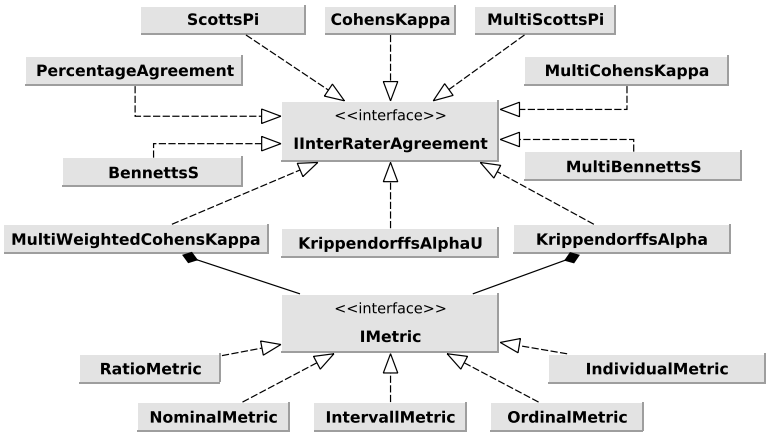

Abbildung 4: Klassendiagramm bezüglich der Koeffizienten

In Abbildung 4 ist das Klassendiagramm bezüglich der einzelnen Koeffizienten und ihren Metriken dargestellt. Alle Koeffizienten werden durch eine entsprechende Klasse repräsentiert. Die Koeffizientenklassen realisieren alle die Schnittstelle `IInterRaterAgreement`. Diese Schnittstelle stellt die Methode „Calculate“ bereit. Die Methode bekommt ein Objekt vom Typ `IDataSet` übergeben, das den Datensatz darstellt, auf dem der jeweilige Koeffizient berechnet wird. Die Klasse `KrippendorffsAlphaU` kann natürlich nur den Koeffizientenwert für Datensätze mit Unitizing berechnen (s. u.). Die Methode berechnet den Koeffizientenwert, wie in Kapitel 3 beschrieben, und gibt sowohl das Ergebnis als auch die Zwischenergebnisse (wie zum Beispiel den Wert der beobachteten Übereinstimmung) zurück. Die gewichteten Koeffizienten haben ein Attribut[5] vom

[3] siehe `src/InterRaterAgreementLibrary`

[4] siehe `src/InterRaterAgreementProgram`

[5] Wie in C# üblich sind alle hier genannten Attribute als Eigenschaften implementiert. Eine Eigenschaft ist

Typ `IMetric`. Die gewünschte Metrik wird gewählt, indem eine passende Instanz einer Klasse, die die Schnittstelle realisiert, dem Attribut zugewiesen wird.

Für die Überprüfung der Korrektheit der Berechnungen sind für jeden Koeffizienten mindestens zwei Unit Tests geschrieben worden. Ein Test mit einem Datensatz, bei dem die Codierer vollständig übereinstimmen, dann ist der erwartete Wert des Koeffizienten gleich eins und ein Test mit einem Datensatz, der das Beispiel aus dem Artikel darstellt, in dem der Koeffizient beschrieben ist.

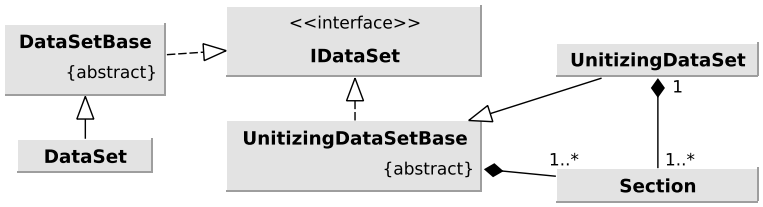

Abbildung 5: Klassendiagramm bezüglich Repräsentation der Datensätze (Kernbibliothek)

Das Klassendiagramm in Abbildung 5 zeigt die Klassen und die Schnittstellen, die für die Repräsentation der Datensätze verantwortlich sind. Ein Datensatz für die Codierung wird durch die Schnittstelle `IDataSet` dargestellt. Diese Schnittstelle stellt Attribute bereit, mit der auf die Menge der Kategorien und die Menge der Elemente zugegriffen werden kann. Zusätzlich enthält es noch die Methode „Save“ mit der ein Datensatz als XML-Datei[6] gespeichert wird. Für den Codierungsfall realisieren die Klassen `DataSetBase` und `DataSet` die Schnittstelle. `DataSet` implementiert dabei nur die Funktionalität des Hinzufügens von Elementen und des Einlesen eines Datensatzes. Der Rest wird in der abstrakten Klasse implementiert. Die Aufteilung der Funktionalität ist notwendig, damit Klassen in der Simulationsbibliothek von `DataSetBase` erben können. Die Klassen der Simulationsbibliothek sollen dabei keine Methoden habe mit denen Elemente hinzugefügt oder eingelesen werden können, da die Elemente vom Programm generiert werden. Bei den Klassen `UnitizingDataSetBase` und `UnitizingDataSet` für den Unitizingfall ist die Aufteilung der Funktionalität analog. Die beiden Klassen realisieren auch `IDataSet`, damit auch `KrippendorffAlphaU` die Schnittstelle `IInterRaterAgreement` realisieren kann.

Im Codierungsfall werden die einzelnen Elemente eines Datensatzes als eine Liste von Integern repräsentiert. Die einzelnen Integerwerte sind dabei die Zuordnungen der Codierer zu diesem Element. Die Elemente werden wiederum in einer Liste organisiert.

Im Unitizingfall werden die einzelnen Abschnitte als Objekte der Klasse `Section` ebenfalls in einer Liste organisiert. Es werden nur die Einheiten dieser Liste hinzugefügt. Die restlichen Abschnitte werden erzeugt, wenn nötig. Die Klasse `Section` besitzt Attribute über die Kategorie, Länge, Codierer, Startpunkt und Endpunkt des repräsentierten Abschnitts.

---

in C# syntaktischer Zucker für eine Realisierung eines Attributes durch eine Getter und eine Setter Methode.

[6] Die Formate sind in XML Schema spezifiziert; siehe `specs/DataSetSchema.xsd` und `specs/UnitizingDataSetSchema.xsd`

5.2 Simulationsbibliothek

Die Simulationsbibliothek besteht aus dem Namespaces „Danlaps.InterRaterAgreement.-Simulation“ und „Danlaps.InterRaterAgreement.Simulation.Util“. Der „Util“ Namespace enthält Hilfsklassen zur Speicherung der Simulationsergebnisse zur Kontrolle der Variablen während der Generierung der Datensätze. Der Namespace „Simulation“ enthält die restlichen Klassen der Implementierung.

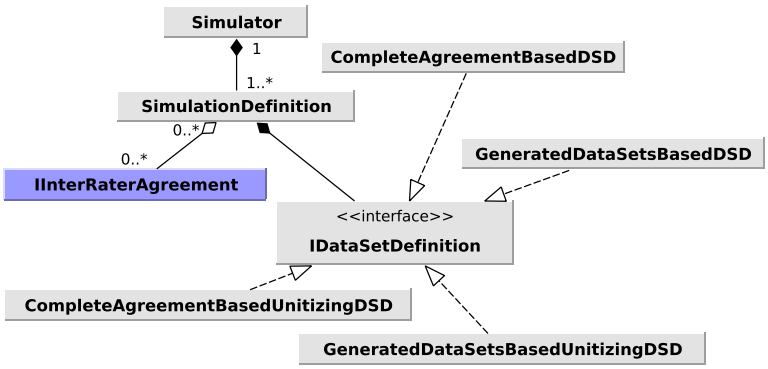

Abbildung 6: Klassendiagramm bezüglich der Simulationsdefinition

In Abbildung 6 sind die wichtigsten Klassen bezüglich der Simulationsdefinition dargestellt. Der Klasse `Simulation` können Objekte vom Typ `SimulationDefinition` hinzugefügt werden. Die Klasse `SimulationDefinition` enthält eine Liste mit den Koeffizienten der Simulation und ein Objekt von einer Klasse, die `IDataSetDefinition` realisiert. Die Simulationsdefinition wird durch den Benutzer in einer XML-Datei[7] geschrieben und dann vom Programm eingelesen. Die hinzugefügten Simulationsdefinitionen können alle zusammen oder auch einzeln ausgeführt werden. Die Ausführung erfolgt, wie in Kapitel vier beschrieben.

Die Klassen `CompleteAgreementBasedDSD` und `CompleteAgreementBasedUnitizingDSD` repräsentieren die Datensatzdefinitionen für die „Generierung durch vollständige Übereinstimmung“ (VÜ). Die Klassen `GeneratedDataSetBasedDSD` und `GeneratedDataSetBasedUnitizingDSD` repräsentieren die Datensatzdefinitionen für die „Generierung durch Einlesen eines bereits generierten Datensatzes“ (ED). Alle Klassen haben als Attribute die Variablen für die Einstellungen der Datensatzgenerierung. Bei der VÜ-Methode wird für jede Kombination der Ausprägungen der Variablen so viele Datensatzinstanzen erstellt, wie vom Benutzer gewünscht. Hierzu werden gemäß den Ausprägungen der Variablen Instanzen der Klasse `CompleteAgreementBasedDS` bzw. `CompleteAgreementBasedUnitizingDS` erstellt. Die beiden Klassen erben von `DataSetBase` bzw. `UnitizingDataSetBase` und in diesen Klassen wird dann gemäß den Vorgaben ein konkreter Datensatz erzeugt. Bei der ED-Methode werden die bereits generierten Datensätze eingelesen und dann gemäß der Ausprägungen der Variablen direkt neue Datensätze erstellt. Hierzu werden die Klassen `DataSet` und `UnitizingDataSet` verwendet.

[7] Das Format ist in XML Schema spezifiziert; siehe `specs/SimulationDefinitonSchema.xsd`

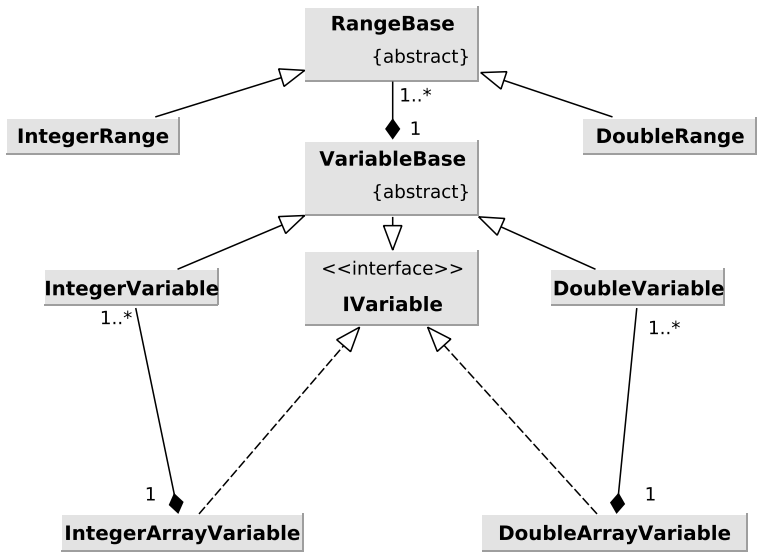

Abbildung 7: Klassendiagramm für die Klassen bezüglich der Variablen zur Datensatzgenerierung

Die Klassen in dem Diagramm von Abbildung 7 werden zur Beschreibung der Variablen der Datensatzdefinitionen verwendet. Es gibt vier verschiedene Typen vor Variablen, die jeweils durch eine Klasse repräsentiert werden: `DoubleVariable`, `IntegerVariable`, `DoubleArrayVariable`, `IntegerArrayVariable`. Die Ausprägungen werden durch entsprechende Zahlenbereich beschrieben. Jedes Range-Objekt repräsentiert einen Zahlenbereich. Ein Bereich ist definiert durch einen Startpunkt, einen Endpunkt und eine Schrittweite. Bei der Generierung der Datensätze werden die Ausprägungen der variablen erzeugt, indem nacheinander die Zahlenbereiche durchgegangen werden. Wenn zum Beispiel ein Zahlenbereich bei eins beginnt und bis fünf geht mit Schrittweite zwei, dann werden für diesen Zahlenbereich die Ausprägungen eins, drei und fünf erzeugt.

5.3 Konsolenprogramm

Das Konsolenprogramm besteht nur aus der Klasse `Start`. Zu Beginn der Ausführung des Programms wird vom Benutzer abgefragt, wo die Simulationsergebnisse gespeichert werden sollen. Anschließend kann der Benutzer Befehle eingeben. Es können Simulationsdefinitionen geladen werden und die definierten Simulationen ausgeführt werden. Das Ergebnis der Simulationen werden dem Benutzer auf der Konsole in Tabellenform dargestellt.

6 Darstellung und Bewertungen der Simulationsergebnisse

Für die Analyse der Interrater-Reliabilitätskoeffizienten sind verschiedene Simulationen durchgeführt worden. Das Kapitel fasst die Ergebnisse der Simulationen zusammen, stellt einen Zusammenhang mit den Analyseergebnissen anderer Autoren her und zieht daraus Schlüsse für die Benutzung der Koeffizienten.

Es ist eine so große Anzahl an Simulationen durchgeführt worden, dass es den Rahmen einer Bachelorarbeit nicht angemessen wäre, hier jede einzelne Simulation vorzustellen. Aus diesem Grund werden im folgenden die Ergebnisse der Simulationen zusammengefasst und nur exemplarisch einige Simulationen vorgestellt. Alle Definitionen und alle Ergebnisse sind auf der beiliegenden CD gespeichert.

6.1 Größe des Datensatzes

Mit der Größe des Datensatzes ist im Codierungsfall die Anzahl der Elemente gemeint und im Unitizingfall die Länge des Datensatzes.

Im Codierungsfall lässt sich als Ergebnis der Simulationen feststellen, dass bei nahezu keinem Koeffizienten eine Änderung seines Wertes zu beobachten war, wenn die Anzahl der Elemente zunehmen oder abnehmen. Als Beispiel für Simulationen, bei der die Datensätze mit der Methode „Generierung durch vollständige Übereinstimmung“ (VÜ) generiert worden sind, sei eine Simulation[8] betrachtet, bei der die Datensätze zwei Codierer und zwei Kategorien (Wahrscheinlichkeitsverteilung: 0.4 erste Kategorie, 0.6 zweite Kategorie) enthalten. Die Ausprägungen der Datensatzdefinition sind die verschiedenen Anzahlen an Elementen. Das Ergebnis ist in Abbildung 8 für Fleiss Kappa dargestellt.

Pro Ausprägung sind bei dieser Simulation 500 Datensatzinstanzen erstellt worden. Die jeweils 500 Ergebnisse je Ausprägung der Koeffizientenberechnung sind als Boxplot dargestellt. In der Abbildung ist keine signifikante Änderung des Koeffizientenwertes festzustellen. Lediglich die Streuung der Ergebnisse nimmt mit zunehmender Anzahl der Elemente ab. Dies war zu erwarten, da mit zunehmender Anzahl an Elementen die Datenmenge des Datensatzes zunimmt. Aus diesem Grund spielen einzelne Ausreißer von der vorgebenen Beschaffenheit der Elemente, die bei der zufälligen Generierung der Elemente entstehen, immer weniger für das Gesamtergebnis der Koeffizientenberechung eine Rolle.

Auch alle anderen Simulationen mit der VÜ-Methode, aber anderen Parametern bezüglich der Beschaffenheit der Datensätze, sowie die Ergebnisse der anderen Koeffizienten, zeigen das gleiche Bild. Eine Ausnahme besteht bei Krippendorffs Alpha und der Erweiterung von Cohens Kappa bezüglich der Ordinalmetrik. Während bei einigen Simulationen die Ergebnisse mit zunehmender Anzahl an Elementen besser werden, werden sie bei anderen Simulationen schlechter. Letztlich muss festgestellt werden, dass die VÜ-Methode hier eher ungeeignet ist. Das Problem ist, dass bei der Definition der Datensätze nicht die tatsächlichen Werte der Kategorien festgelegt werden können, sondern dass die Kategorien beginnend mit der Null durchnummeriert werden. Die tatsächlichen Werte sind aber bei der Berechnung der Ordinalmetrik entscheidend. Es kann nicht getestet

[8] siehe `testNumberOfTokens/2R2C0406CP0_1RCP02TCP`

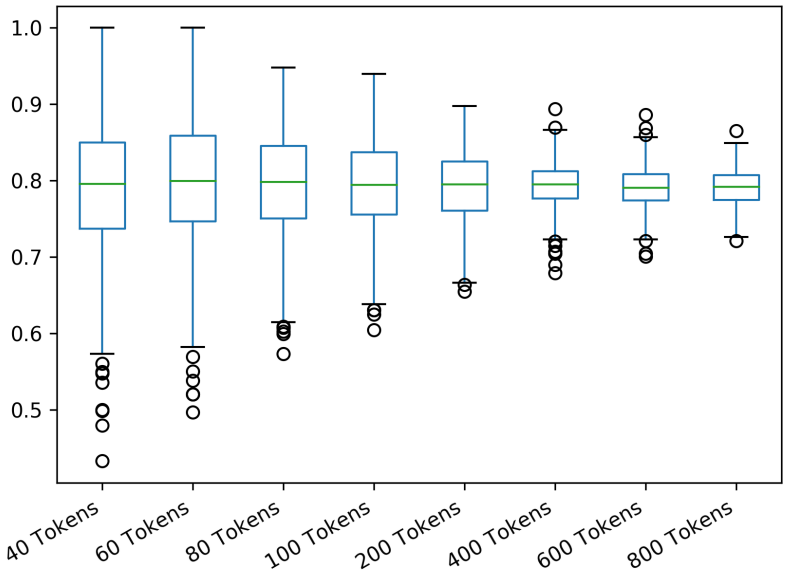

Abbildung 8: Darstellung der Simulationsergebnisse für Fleiss Kappa und steigener Anzahl an Elementen

werden, wie die Ergebnisse aussähen, wenn die Kategorien zum Beispiel weiter auseinander lägen. Aus diesem Grund kann an dieser Stelle keine abschließende Bewertung der Ergebnisse vorgenommen werden. Analog gilt das auch für die Intervallmetrik und die Verhältnismetrik. Aus diesem Grund wird auf die Ergebnisse, die unter Benutzung von Metriken mit Ausname der Nominalmetrik, gewonnen worden sind, nicht mehr eingegangen.

Elementanzahl:	40	60	80	100	200	400	600	800
1. Datensatz	0.52	0.52	0.52	0.52	0.52	0.52	0.52	0.52
2. Datensatz	0.52	0.52	0.52	0.52	0.52	0.52	0.52	0.52

Tabelle 13: Simulationsergebnisse für Fleiss Kappa und steigener Anzahl an Elementen

Für die Simulationen mit der Methode „Generierung durch Einlesen eines bereits generierten Datensatzes“ (ED) sind zwei per Hand erstellte Datensätze[9] verwendet worden. Beide Datensätze enthalten zwei Codierer und drei Kategorien. Die Datensätze wurden so erstellt, dass die prozentuale Übereinstimmung für beide Datensätze gleich ist. Bei dem einem Datensatz sind die Gesamtanzahlen, wie oft ein Codierer eine bestimmte Kategorie Elementen zuordnet, für jede Kategorie und beide Codierer gleich. Bei dem anderen Datensatz sind die Gesamtanzahlen für die beiden Codierer unterschiedlich. Mit Ausnahme von Krippendorffs Alpha sind die Koeffizientenwerte unabhängig von der Anzahl der Elemente immer gleich (vgl. exemplarisch die Werte in Tabelle 13 für Fleiss

[9] siehe `testNumberOfTokens/dataSets`

Kappa). Die gewonnen Erkenntnisse aus der VÜ-Methode werden damit bestätigt.

Elementanzahl:	40	60	80	100	200	400	600	800
1. Datensatz	0.526	0.524	0.523	0.5224	0.5212	0.5206	0.5204	0.5203
2. Datensatz	0.526	0.524	0.523	0.5224	0.5212	0.5206	0.5204	0.5203

Tabelle 14: Simulationsergebnisse für Krippendorffs Alpha mit der Nominalmerik und steigener Anzahl an Elementen

Die Ergebnisse für Krippendorffs Alpha mit der Nominalmetrik sind in Tabelle 14 dargestellt. Die Werte nehmen mit zunehmender Anzahl ganz leicht ab und nähern sich den Werten von Fleiss Kappa bzw. Scotts Pi an. Die leichte Veränderung ist daher auf den bereits beschriebenen Korrekturfaktor zurückzuführen.

Für Unitizing lässt sich festhalten, dass die Ergebnisse, die durch Simulation mit der VÜ-Methode gewonnen worden sind, nicht besonders aussagekräftig sind. Der Grund liegt in der Tatsache begründet, dass im ersten Schritt der Generierung eine feste Anzahl von Einheiten mit fester Länge auf dem Datensatz verteilt werden. Dies führt dazu, dass mit steigender Länge des Datensatzes das Verhältnis von kuEs[10], die einer Einheit zugeordnet sind, zu denen, die einem Abschnitt, der keine Einheit ist, zugeordnet sind, nicht konstant ist. Der Grund ist, dass der Abstand zwischen den Einheiten immer größer wird. Es ist aber nicht zu erwarten, dass sich dieses Verhältnis bei einem größer werdenden Datensatz verändert, wenn jeweils die gleichen Codierer kodieren. Zwar könnte dieser Umstand ausgeglichen werden, indem bei steigender Länge des Datensatzes auch die Länge der Einheiten zu nimmt, aber die Länge der Einheiten hat Auswirkungen auf die Messung der Distanz durch die Distanzfunktion bei Krippendorffs α_U.

Für den Unitizingfall werden daher nur die Ergebnisse, die durch Simulationen mit der ED-Methode gewonnen worden sind, vorgestellt. Es sind per Hand sieben ganz verschiedene Datensätze erstellt worden. Anschließend sind für diese Datensätze mehrere Simulationen durchgeführt worden. Die Simulationen unterscheiden sich in der Frage, wie hoch die Anzahl der Codierer und der Kategorien der mit diesen Simulationen neu generierten Datensätze sind. Exemplarisch sind die Werte für den Fall, dass die Datensätze zwei Codierer und eine Kategorie beinhalten, dargestellt[11] (Tabelle 15). Es lässt sich feststellen, dass die Werte für alle Datensätze geringfügig steigen, wenn die Länge des Datensatzes zunimmt. Dieses Ergebnis ist bei allen Simulationen zu beobachten.

Es bleibt die Frage, was eine gute Größe für einen Datensatz ist. Aus der Tatsache, dass die Koeffizientenwerte, abhängig von der Größe des Datensatzes, gar nicht oder nur sehr geringfügig ändern, kann abgeleitet werden, dass die Größe keine allzu große Relevanz hat unter der Bedingung, dass die Codierer unabhängig von der Größe des Datensatzes annotieren. Krippendorff gibt als Faustregel für den Codierungsfall an, dass die Größe des Datensatzes mindestens so groß sein sollte, dass jede Kategorie mindestens so oft verwendet wird, dass sich mindestens fünf Übereinstimmungen durch Zufall ergeben [Kri04, Abs. 11.4.3]. Daraus folgt, dass der Datensatz umso größer sein sollte, desto mehr Kategorien verwendet werden und desto ungleichmäßiger die einzelnen Kategorien Verwendung finden.

[10] kleinste unterscheidbare Einheit

[11] siehe `testNumberOfRaters/2R1C`

Vergrößerungsfaktor:	1	2	3	4	5	6	7	8	9	10
1. Datensatz	0.78426	0.78849	0.78985	0.79052	0.79092	0.79118	0.79137	0.79151	0.79162	0.79171
2. Datensatz	0.10599	0.11471	0.11753	0.11893	0.11976	0.12031	0.12071	0.12100	0.12123	0.12141
3. Datensatz	0.10599	0.11471	0.11753	0.11893	0.11976	0.12031	0.12071	0.12100	0.12123	0.12141
4. Datensatz	-0.29404	-0.29122	-0.29032	-0.28989	-0.28963	-0.28946	-0.28933	-0.28924	-0.28917	-0.28911
5. Datensatz	0.98937	0.98986	0.99001	0.99007	0.99011	0.99014	0.99016	0.99017	0.99019	0.99019
6. Datensatz	0.98689	0.98723	0.98734	0.98740	0.98743	0.98745	0.98746	0.98747	0.98748	0.98749
7. Datensatz	0.37837	0.38355	0.38524	0.38607	0.38657	0.38690	0.38713	0.38731	0.38745	0.38756

Tabelle 15: Simulationsergebnisse für Krippendorffs Alpha U und steigender Anzahl an Elementen

6.2 Anzahl der Kategorien

Wenn der Datensatz codiert wird, dann konnte in allen durchgeführten Simulationen und bei fast allen Koeffizienten keine signifikante Änderung des Koeffizientenwertes festgestellt werden. Nur bei prozentualer Übereinstimmung aufgefasst als Koeffizient war eine Verschlechterung mit zunehmender Anzahl an Kategorien festzustellen. Dies deckt sich mit den Ausführungen am Ende von Abschnitt 3.1. Für die Simulation von unterschiedlicher Kategorienanzahlen ist es nicht möglich, die ED-Methode zu verwenden. Der Grund ist, dass es keine passende Variable bei der Methode gibt, die es ermöglichen würde, Ausprägungen mit verschiedenen Kategorienanzahlen zu generieren.

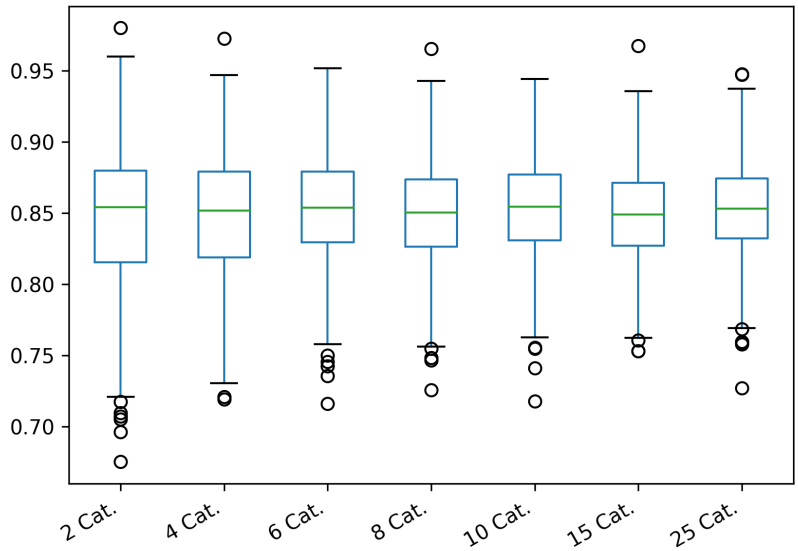

Abbildung 9: Darstellung der Simulationsergebnisse für Krippendorffs Alpha mit der Nominalmetrik und steigener Anzahl an Kategorien

In Abbildung 9 sind die Ergebnisse beispielhaft bezüglich Krippendorffs Alpha für eine Simulation[12], bei der die Datensätze zwei Codierer und 100 Elemente enthalten. Wie bereits geschrieben, ist keine Änderung des Koeffizientenwertes bei steigender Kategorienanzahl festzustellen. Die Ergebnisse für prozentuale Übereinstimmung sind in Abbildung 10 zu finden. Hier zeigt sich die abnehmende Übereinstimmung bei steigender Kategorienanzahl.

Im Falle von Unitizing ist die aus den Simulationen gewonnene Erkenntnis, dass hier ebenfalls keine Änderung eintritt, wenn die Kategorienanzahl wächst. Zu Beachten ist aber, dass sowohl die beobachtete Übereinstimmung als auch die erwartete Übereinstimmung bei α_U für jede Kategorie einzeln berechnet wird. Wenn die Abschnitte der zusätzlichen Kategorien eine größere oder kleinere Übereinstimmung haben als die anderen

[12] siehe `testNumberOfCategories/2R100T0406CP0505RCP02TCP`

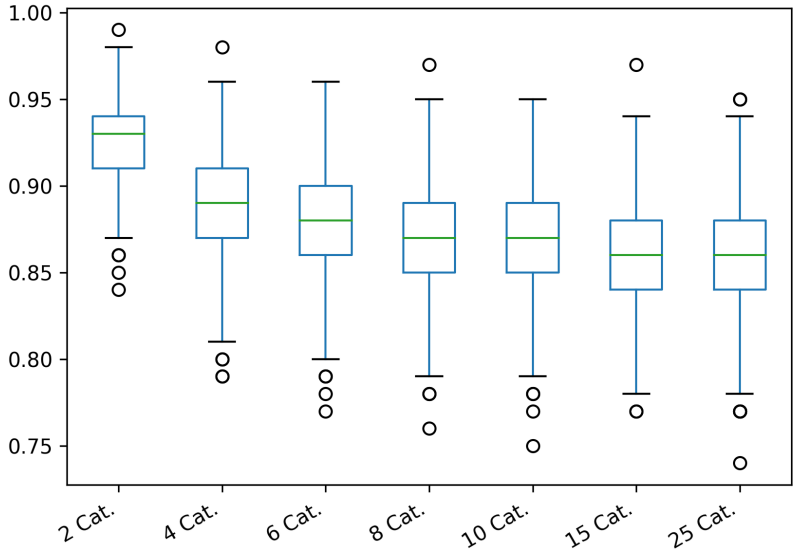

Abbildung 10: Darstellung der Simulationsergebnisse für prozentuale Übereinstimmung und steigender Anzahl an Kategorien

Kategorien, dann ist der Wert von Krippendorffs α_U nur von der Übereinstimmung der Abschnitte dieser Kategorien abhängig und nicht mehr von der Anzahl der Kategorien.

Für die VÜ-Methode ist der beschriebene Sachverhalt in Abbildung 11 dargestellt. Abgebildet sind die Ergebnisse einer Simulation[13], bei der die Datensätze die Länge 200 haben, 4 Codierer beinhalten und es gibt nach dem ersten Schritt zehn Einheiten je Kategegorie und Codierer mit der Länge zehn. Es ist zu erkennen, dass die Koeffizientenwerte gleich bleiben, wenn nur die Ausprägungen mit gerader bzw. ungerader Kategorienanzahl betrachtet werden. Wenn alle Ausprägungen betrachtet werden, existieren beträchtliche Unterschiede von Kategorieanzahl zu Kategorieanzahl. Der Grund ist folgender: Bei ungerader Anzahl werden die Einheiten innerhalb der Kategorien unterschiedlich stark verändert, was zu einer niedrigeren Übereinstimmung bezüglich der einzelnen Kategorien und damit auch des Gesamtergebnisses führt. Bei gerader Anzahl werden dagegen die Einheiten der Kategorien untereinander weniger stark geändert. Stattdessen sind Parameter für die Generierung der Datensätze so gewählt, dass die Veränderungen bei einzelnen Kategorien stärker ausfallen. Dies führt aber nicht zu einer Verringerung der Übereinstimmung, weil die Übereinstimmung für jede Kategorie einzeln berechnet wird.

Die Ergebnisse, die durch die ED-Methode gewonnen worden sind, entsprechen den Ergebnissen, die mit der VÜ-Methode gewonnen worden sind. Es sind für die Generierung der neuen Datensätze die gleichen sieben Datensätze, wie für die Analyse der Größe des Datensatzes verwendet worden. Die Ergebnisse als Beispiel für die durchgeführten

[13] siehe `testNumberOfCategories/ 200L4R10UC1_12EF2GR2_-2TV05UCP10UL`

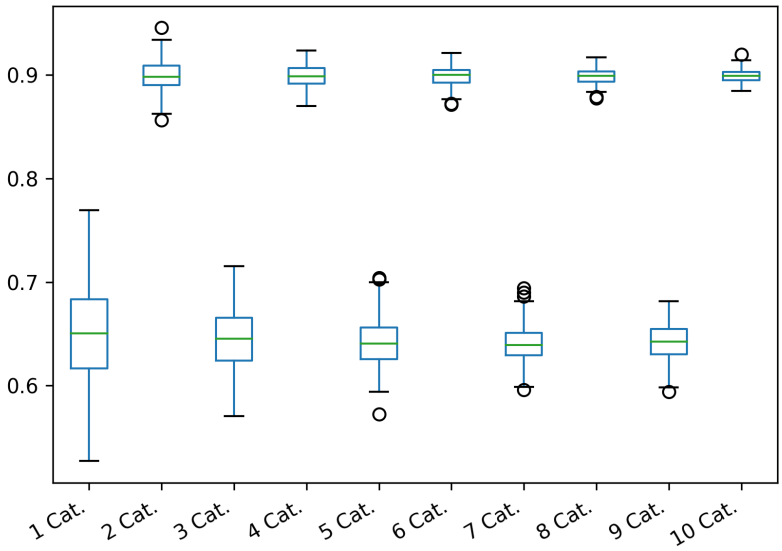

Abbildung 11: Darstellung der Simulationsergebnisse für α_U und steigender Anzahl an Kategorien

Simulationen ist in Tabelle 16 dargestellt. Es ist keine Änderung des Koeffizientwertes feststellbar.

Kategorieanzahl:	1	3	6	9
1. Datensatz	0.86078	0.86078	0.86078	0.86078
2. Datensatz	0.41400	0.41400	0.41400	0.41400
3. Datensatz	0.41400	0.23112	0.18987	0.20183
4. Datensatz	0.14048	0.14048	0.14048	0.14048
5. Datensatz	0.99345	0.99345	0.99345	0.99345
6. Datensatz	0.99165	0.99165	0.99165	0.99165
7. Datensatz	0.59144	0.59144	0.59144	0.59144

Tabelle 16: Simulationsergebnisse für Krippendorffs α_U und steigener Anzahl an Kategorien (ED-Methode)

Zumindest was den Codierungsfall betrifft, liegt die Vermutung nahe, zu behaupten, dass die Anzahl der Kategorien nicht von Belang ist. Stattdessen ist nach Krippendorff[Kri04, Abs. 11.4.3] als auch nach Artstein und Poesio[AP07] darauf zu achten, dass die Kategorien so gewählt werden, dass sie sich gegenseitig ausschließen und alle Kategorien in etwa gleich häufig verwendet werden.

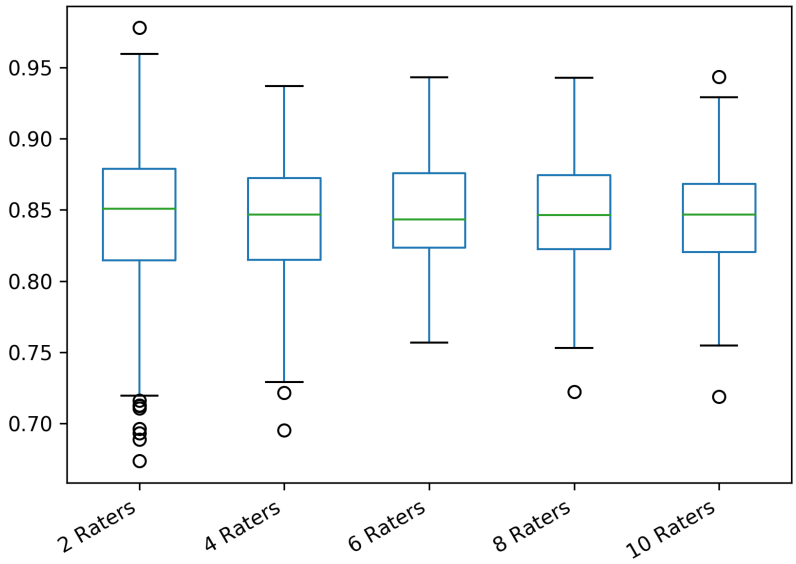

Abbildung 12: Darstellung der Simulationsergebnisse für Fleiss Kappa und steigender Anzahl an Codierern (VÜ-Methode/alle Codierer gleich gut)

6.3 Anzahl der Codierer

In Abbildung 12 ist examplarisch das Ergebnis einer Simulation[14] für Fleiss Kappa dargestellt für den Fall, dass die zusätzlichen Codierer gleich gut codieren wie die bereits enthaltenen Codierer. Die Datensätze sind mit der VÜ-Methode generiert worden. Das bedeutet, dass die Wahrscheinlichkeit, dass sich im zweiten Schritt der Generierung die Zuordnung eines Elementes ändert, für alle Codierer gleich ist. Es ist festzustellen, dass keine signifikante Änderungen der Werte zu beobachten ist. Das gleiche gilt auch für alle anderen Koeffizienten. Etwas anderes gilt für die Ergebnisse der Simulationen der ED-Methode in diesem Fall. In der Tabelle 17 sind die Ergebnisse bezüglich Fleiss Kappa für eine Simulation[15] dargestellt bei der wieder die beiden bereits in Abschnitt 6.1 besprochenen Datensätze zur Generierung verwendet worden sind. Hier kann eine Erhöhung der Koeffizientenwerte bei zunehemder Codiereranzahl beobachtet werden, wobei die Erhöhung immer weiter abflacht.

Codiereranzahl:	2	4	6	8	12	16
1. Datensatz	0.5200	0.6800	0.7120	0.7257	0.7382	0.7440
2. Datensatz	0.5200	0.6800	0.7120	0.7257	0.7382	0.7440

Tabelle 17: Darstellung der Simulationsergebnisse für Fleiss Kappa und steigender Anzahl an Codierern (ED-Methode/alle Codierer gleich gut)

[14] siehe `testNumberOfRaters/2C100T0406CP05RCP02TCP`
[15] siehe `testNumberOfRaters/100T`

Für den Fall, dass die Codierer, die hinzu kommen, schlechter codieren, lässt sich für alle Koeffizienten feststellen, dass dann auch die Koeffizientenwerte schlechter werden. Für die VÜ-Methode zeigen das zum Beispiel die Ergebnisse einer Simulation[16], bei der die Elemente der ersten beiden Codierer im zweiten Schritt nur mit niedrieger Wahrscheinlichkeit geändert werden und die Elemente der restlichen Codierer mit deutlich höherer Wahrscheinlichkeit geändert werden. Die Ergebnisse dieser Simulation für Fleiss Kappa finden sich in Abbildung 13.

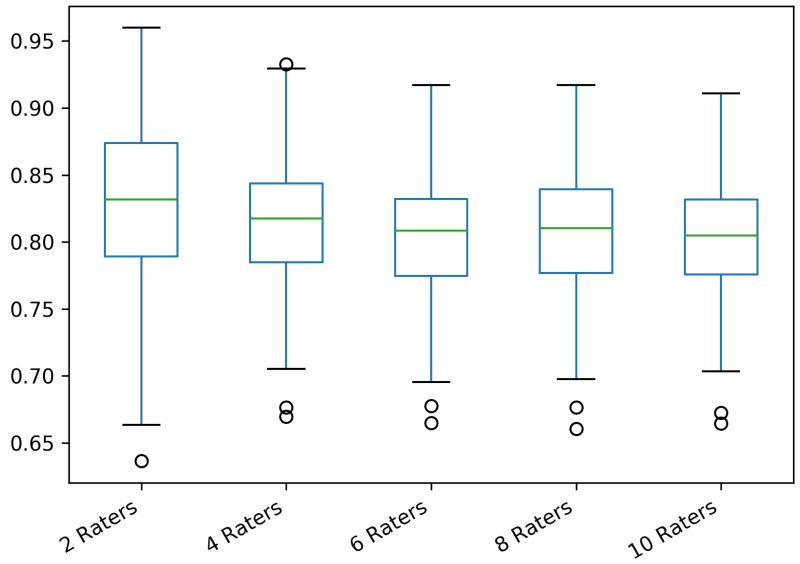

Abbildung 13: Darstellung der Simulationsergebnisse für Fleiss Kappa und steigender Anzahl an Codierern (VÜ-Methode/alle Codierer nicht gleich gut)

Bei der ED-Methode ist in diesem Fall wie folgt vorgegangen worden: Es sind mit der VÜ-Methode auf den Fall passende Datensätze mit drei oder fünf Codierern generiert worden. Anschließend sind in zwei Simulationen[17] jeweils Ausprägungen mit drei bzw. fünf und mit jeweils einem Codierer weniger generiert worden. Das Ergebnis der Simulation ist exemplarisch für den Koeffizienten Fleiss Kappa Abbildung 14 dargestellt. Die Ergebnisse der Simulation mit der VÜ-Methode werden bestätigt.

Auf Grund der Ergebnisse lässt sich feststellen, dass es Sinn machen kann, schlechte Codierer auszusortieren, um eine höhere Reliabilität zu erreichen. Zu Beachten ist jedoch, dass Reliabilität keine Aussage über den Wahrheitsgehalt der Codierungen macht und daher bei verminderter Codiereranzahl die Gefahr steigt, dass zwar die Codierer eine hohe Übereinstimmung erzielen, aber sie objektiv vermehrt falsch codieren.

Im Unitizingfall zeigt sich das gleiche Bild wie im Codierungsfall: Es ist keine Änderung der Koeffizientwerte festzustellen, wenn die Codierer alle gleich gut codieren und die

[16] siehe `testNumberOfRaters/2C100T0406CP0108RCP02TCP`
[17] siehe `testNumberOfRaters/eleminate1RaterOf3` und `testNumberOfRaters/eleminate1RaterOf5`

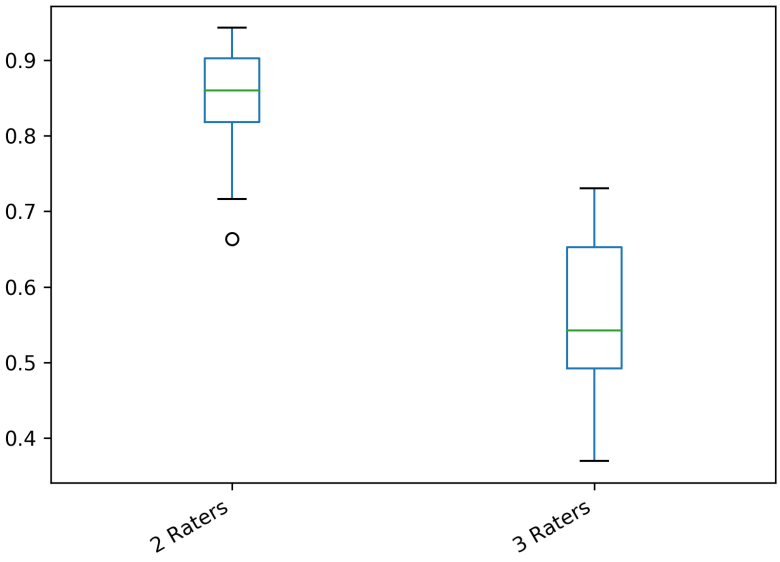

Abbildung 14: Darstellung der Simulationsergebnisse für Fleiss Kappa und steigender Anzahl an Codierern (ED-Methode/alle Codierer nicht gleich gut)

Werte werden schlechter, wenn die hinzukommenden Codierer schlechter codieren als die nicht hinzukommenden Codierer. Bei Simulationen mit der ED-Methode ist es analog zum Codierungsfall so, dass bei gleich guten Codierern die Werte von α_U ansteigen mit steigender Codiereranzahl.

6.4 Weitergehende Fragestellungen

6.4.1 Zusammenstoßende Einheiten

Beim Unitizing stellt sich die Frage, welchen Wert Krippendorffs α_U annimmt, wenn die Einheiten des gleichen Codierers und Kategorie direkt zusammenstoßen und in diesem Fall die Einheiten nicht zu einer Einheit zusammengefasst werden. Für die Untersuchung dieser Frage sind in einem ersten Schritt drei Datensätze verschiedenster Art vom Autor der Bachelorarbeit erstellt worden. In einem zweiten Schritt ist für jeden Datensatz ein weiterer Datensatz erstellt worden, indem jede Einheit in zwei Einheiten unterteilt worden ist. Die Datensätze des erste Schritts werden mit A bezeichnet und die des zweiten Schritts mit B. Anschließend ist sowohl für die A als auch für die B Datensätze eine Simulation mit der ED-Methode durchgeführt worden, die für verschiedene Anzahlen von Codierer α_U berechnet. In allen Fällen zeigt sich, dass die berechneten Werte von α_U für die B Datensätze ausgesprochen schlechter sind als für die A Datensätze. Die Ergebnisse sind in Tabelle 18 dargestellt. Auf Grund dieser Erkenntnis ist es sinnvoll, die Codierer anzuhalten, Einheiten, die zusammenstoßen, zu verbinden.

Codiereranzahl:	2	3	5
1. Datensatz A	-0.77316	-0.04555	0.01830
1. Datensatz B	-0.60115	-0.03607	0.06275
2. Datensatz A	0.78714	0.88030	0.88578
2. Datensatz B	0.42841	0.62733	0.66238
3. Datensatz A	0.98740	0.99179	0.99258
3. Datensatz B	0.38607	0.59851	0.63641

Tabelle 18: Darstellung der Simulationsergebnisse für die Analyse von zusammenstoßenden Einheiten

6.4.2 Auswahl eines Koeffizienten

Die Frage, welcher der analysierten Koeffizienten für die Bewertung der Reliabilität besonders geeignet ist, ist nicht so einfach aus den Ergebnissen der Simulationen zu beantworten. Nur die prozentuale Übereinstimmung aufgefasst als Koeffizient kann nicht als guter Koeffizient angesehen werden auf Grund der Problematik bei zunehmender Anzahl an Kategorien. Alle andern Koeffizienten haben bei der Analyse keine Schwächen gezeigt, sodass man einen Koeffizienten ausschließen könnte. Die Frage nach der Auswahl eines Koeffizienten stellt sich im Übrigen nur für den Codierungsfall, da im Unitizingfall sowieso nur Krippendorffs α_U allgemein anerkannt ist.

Bei den gewichteten Koeffizienten ist zu beachten, dass die berechneten Werte und ihr Verhalten für verschiedene Ausprägungen von Datensätzen stark von der verwendeten Metrik abhängen. Aus diesem Grund ist es sehr schwierig die Werte zu interpretieren. Artstein und Poesio empfehlen dann auch, nur gewichtete Koeffizienten zu benutzen, wenn dies unbedingt notwendig ist [AP07].

Sowohl Krippendorff als auch Artstein und Poesio empfehlen die Verwendung von Fleiss Kappa bzw. Krippendorffs Alpha. Bennetts S und seine Erweiterung wird abgelehnt, weil Bennetts S darauf basiert, dass alle Kategorien gleich häufig verwendet werden. Diese Annahme ist aber eher unrealistisch für die Praxis. So gut wie nie werden in der Praxis alle Kategorien gleich häufig verwendet. Hinzu kommt das Bennetts S Werte steigen, wenn dem Datensatz Kategorien hinzugefügt werden, die gar nicht oder nur sehr wenig verwendet werden. Bei Cohens Kappa ist das Problem, dass der berechnete Wert davon abhängt, bei welchem Codierer eine Nichtübereinstimmung zu den anderen Codierern aufgetreten ist und welche Kategorien besonders oft falsch zugeordnet werden. Alle diese Probleme treten bei Fleiss Kappa bzw. Krippendorffs Alpha nicht auf, da sie die erwartet Übereinstimmung unabhängig von Zuordnungen der Codierer bestimmen [AP07, Kri04].

6.4.3 Notwendige Höhe des Koeffizientenwertes für die Akzeptanz der Annotation

Zum Schluss sei noch die die Frage besprochen, ab welchen Koeffizientenwert eine Annotation akzeptiert werden sollte, weil die Reliabilität ausreicht. Auch in diesem Fall geben die Simulationen keinen Aufschluss über diese Frage und es existieren auch in der Literatur verschiedene Ansichten über die Antwort dieser Frage. Krippendorff empfiehlt

keine der Annotationen zu akzeptieren, bei denen der Wert von Krippendorff Alpha unter 0.8 fällt. Nicht unwichtig ist bei der Frage, welcher Wert zu akzeptieren ist, für was die Annotation verwendet wird. Zum Beispiel ist fü die Annotation, an deren Korrektheit Menschenleben abhängen, höhere Anforderungen an die Reliabilität zu stellen als bAnnotationen deren Inkorrektkeit deutlich weniger extreme Folgen hat [Kri04, Abs. 11.4.4].

Artstein und Poesio schließen sich der Empfehlung von Krippendorff an. Zusätzlich halten sie es nicht für ausreichend bei Veröffentlichungen nur den Wert eines oder auch mehrerer Koeffizienten anzugeben. Statdessen sollte zusätzlich die Art angegeben werden wie die Annotationen zustande gekommen sind (z. B. Angaben über die Anzahl der Codierer oder wie und welche Anleitung bei der Annotation verwendet worden ist) und Details über die gemachte Annotation über eine Übereinstimmungstabelle zum Beispiel.

7 Zusammenfassung

Zum Abschluss wird eine Zusammenfassung der wichtigsten gewonnen Erkenntnisse gegeben und ein Ausblick auf weiterführende Arbeiten gewagt.

7.1 Fazit

Das Ziel der Bachelorarbeit bestand in der Entwicklung eines geeigneten Simulationsprogramms und eine anschließenden Analyse der Interrater-Reliabilitätskoeffizienten mit Hilfe des entwickelten Simulationsprogramms.

Es konnte demonstriert werden, dass das entwickelte Simulationsprogramm geeignet ist, um zu verschiedenen Fragestellungen passende Simulationen durchzuführen. Allerdings offenbarte es auch bezüglich gewichteten Koeffizienten seine Schwächen. Insgesamt hat sich herausgestellt, dass die Methode der Simulation für die Analyse der Koeffizienten gewinnbringend sein kann und im Vergleich zu einer mathematischen Analyse nicht von vornherein schlechter ist. Festzustellen ist aber auch, dass es Fragestellungen gibt, die mindestens bei der vorgestellten Implementierung nicht einer Analyse durch Simulation zugänglich sind.

Bei der Analyse der Koeffizienten konnte festgestellt werden, dass bis auf prozentuale Übereinstimmung alle Koeffizienten bezüglich der untersuchten Eigenschaften gleich gut abgeschnitten haben. Trotzdem konnte basierend auf der Analyse anderer Autoren eine Empfehlung für Krippendorffs Alpha bzw. Fleiss Kappa ausgesprochen werden.

7.2 Ausblick

Eine interessante Erweiterung des Simulationsprogramms wäre die Möglichkeit, Datensätze einzulesen, die mit existierenden Annotationstools erstellt worden sind. Mit solch einer Erweiterung ergäben sich weitere Möglichkeiten für Simulationen und der Analyse. Es wäre dann einfach, Datensätze von tatsächlich in der Forschung erstellten Annotationen zu verwenden. Zum einen könnten potenzielle Benutzer untersuchen, wie die Werte ihrer Datensätze auf Veränderungen der Daten reagieren, beispielsweise, wenn einer der Codierer aus dem Datensatz genommen wird. Zum anderen könnten Datensätze verschiedener Autoren miteinander verglichen werden.

Die Analyse dieser Arbeit beschränkt sich auf die generellen Eigenschaften der Interrater-Reliabilitätskoeffizienten unabhängig davon, was für Daten annotiert werden. Weiterführende Analysen und Simulationen könnten darin bestehen zu untersuchen, was es bei den einzelnen Typen von Annotationsaufgaben zu beachten gibt. Es macht möglicherweise einen Unterschied, ob einzelne Wörter eines Textes oder ganze Texte annotiert werden oder auch, ob Ärzte Patientendiagnosen stellen. Vorstellbar wäre, dass für verschiedene Arten von Annotationsaufgaben jeweils andere Koeffizienten geeignet wären oder das je nach Aufgabe andere Anforderungen an die Höhe der Koeffizienten gestellt werden, um eine Annotation zu akzeptieren.

Anhang

Beweis der Gleichheit der beobachteten Übereinstimmung für zwei Codierer und beliebig viele Codierer

Zu zeigen ist $U_b^{C=2} = U_b$.

Zuerst ist zu zeigen, dass

$$\sum_{k \in \mathbf{K}} n_{ik}(n_{ik} - 1) = 2f(i)$$

für jedes i-te Element und der Funktion f aus der Gleichung 1. Entweder stimmen die beiden Codierer überein, dann existiert eine Kategorie k_1, für die gilt $n_{ik_1} = 2$ und für alle restlichen Kategorien gilt dann $n_{ik} = 0$, oder sie stimmen nicht überein, dann existieren zwei Kategorien k_1 und k_2, für die gilt $n_{ik_1} = 1$ und $n_{ik_2} = 1$, sowie $n_{ik} = 0$ für die restlichen Kategorien.Für den ersten Fall ergibt das

$$n_{ik_1}(n_{ik_1} - 1) + \sum_{k \in \mathbf{K} \setminus \{k_1\}} n_{ik}(n_{ik} - 1) = 2(2-1) + \sum_{k \in \mathbf{K} \setminus \{k_1\}} 0(0-1) = 2 = 2 \cdot 1 = 2f(i)$$

und für den zweiten Fall

$$\sum_{k \in \{k_1, k_2\}} n_{ik}(n_{ik}-1) + \sum_{k \in \mathbf{K} \setminus \{k_1, k_2\}} n_{ik}(n_{ik}-1) = 2 \cdot 1(1-1) + \sum_{k \in \mathbf{K} \setminus \{k_1, k_2\}} 0(0-1) = 0 = 2 \cdot 0 = 2f(i)$$

Weiter gilt dann:

$$U_b = \frac{1}{N \cdot 2(2-1)} \sum_{i=1}^{N} \sum_{k \in \mathbf{K}} n_{ik}(n_{ik} - 1) = \frac{1}{2N} \sum_{i=1}^{N} 2 \cdot f(i) = \frac{1}{N} \sum_{i=1}^{N} f(i) = U_b^{C=2} \quad \square$$

Beweis der Gleichheit der erwarteten Übereinstimmung für Randolphs Kappa und der allgemeinen Definition der erwarteten Übereinstimmung

Es ist zeigen, dass für $P'(k|c) = \frac{1}{K}$ mit k eine beliebige Kategorie und c ein beliebiger Codierer die Definition von Randolph konsistent zu der Gleichung 9 ist, dass also gilt:

$$U_e^{S^{C>2}} = \sum_{k \in \mathbf{K}} \frac{1}{\binom{C}{2}} \sum_{i=1}^{C-1} \sum_{j=i+1}^{C} P'(k|c_i) P'(k|c_j) = \frac{1}{K}$$

Zuerst ist per vollständiger Induktion über die Anzahl der Codierer C für ein beliebiges $x \in \mathbb{R}$ zu zeigen:

$$\sum_{i=1}^{C-1} \sum_{j=i+1}^{C} x = \frac{C(C-1)x}{2}$$

Induktionsanfang. Sei $C = 1$, dann gilt:

$$\sum_{i=1}^{0} \sum_{j=i+1}^{1} x = 0 = \frac{1(1-1)x}{2}$$

Induktionsvoraussetzung. Nach der Induktionsvoraussetzung gilt:

$$\sum_{i=1}^{C-1} \sum_{j=i+1}^{C} x = \frac{C(C-1)x}{2}$$

Induktionsschritt. $C \to C+1$:

$$\begin{aligned}
\sum_{i=1}^{C+1-1} \sum_{j=i+1}^{C+1} x &= \left(\sum_{i=1}^{C} \sum_{j=i+1}^{C} x \right) + \sum_{i=1}^{C} x \\
&= \left(\sum_{i=1}^{C-1} \sum_{j=i+1}^{C} x \right) + \left(\sum_{i=1}^{C} x \right) + \overbrace{\sum_{j=C+1}^{C} x}^{=0} \\
&\overset{IV}{=} \frac{C(C-1)x}{2} + \sum_{i=1}^{C} x \\
&= \frac{C(C-1)x}{2} + Cx \\
&= \frac{x(C(C-1)+2C)}{2} \\
&= \frac{x(C^2+C)}{2} \\
&= \frac{xC(C+1)}{2} \\
&= \frac{(C+1)((C+1)-1)x}{2}
\end{aligned}$$

Setze jetzt $x := \frac{1}{K^2}$, dann gilt:

$$\begin{aligned}
U_e^{S^{C>2}} &= \sum_{k \in \mathbf{K}} \frac{1}{\binom{C}{2}} \sum_{i=1}^{C-1} \sum_{j=i+1}^{C} P(k|c_i) P(k|c_j) \\
&= \sum_{k \in \mathbf{K}} \frac{1}{\binom{C}{2}} \sum_{i=1}^{C-1} \sum_{j=i+1}^{C} \frac{1}{K^2} \\
&= \sum_{k \in \mathbf{K}} \frac{\frac{C(C-1)}{2} \cdot \frac{1}{K^2}}{\binom{C}{2}} \\
&= \sum_{k \in \mathbf{K}} \frac{\binom{C}{2} \frac{1}{K^2}}{\binom{C}{2}} \\
&= \sum_{k \in \mathbf{K}} \frac{1}{K^2} \\
&= K \frac{1}{K^2} \\
&= \frac{1}{K} \quad \square
\end{aligned}$$

Literatur

[AP07] ARTSTEIN, Ron ; POESIO, Massimo: *Inter-Coder Agreement for Computational Linguistics*. `http://cswww.essex.ac.uk/Research/nle/arrau/icagr.pdf`. Version: 2007, Abruf: 01. 09. 2017. – Die Quelle bezieht sich auf die verlinkte erweiterte Version eines im Computational Linguistics veröffentlichen Artikel.

[BAG54] BENNETT, E. M. ; ALPERT, R. ; GOLDSTEIN, A. C.: Communications Through Limited - Response Questioning. In: *The Public Opinion Quarterly* (1954)

[Coh60] COHEN, Jacob: A coefficient of agreement for nominal scales. In: *Educational and Psychological Measurement* (1960)

[Fle71] FLEISS, Joseph L.: Measuring nominal scale agreement among many raters. In: *Psychological Bulletin* (1971)

[Kri95] KRIPPENDORFF, Klaus: On the reliability of unitizing contigugous data. In: *Sociological Methodology* (1995)

[Kri04] KRIPPENDORFF, Klaus: *Content Analysis: An Introduction to Its Methodology*. Second Edition. Sage Publications, 2004

[Ran05] RANDOLPH, Justus J.: Free-Marginal Multirater Kappa (multirater κ_{free}): An Alternative to Fleiss' Fixed-Marginal Multirater Kappa. In: *Proceedings of the 5th Joensuu University Learning and Instruction Symposium* (2005)

[Sco55] SCOTT, William A.: Reliability of Content Analysis: The Case of Nominal Scale Coding. In: *The Public Opinion Quarterly* (1955)

Abbildungsverzeichnis

Tabellenverzeichnis