Softwareprojekt WS 2016/17
Robert Saß, Darmin Spahic

Universität Heidelberg
Institut für Computerlinguistik

Spezifikationsdokument
Negation Detection

„WUVV“

1. Ziel des Projekts

Das Ziel dieses Projekts ist es, ein Programm zu entwickeln, welches in einem Korpus Sätze findet, die Verneinungen enthalten, sowie Skopus und Fokus der jeweiligen Verneinung finden kann.

1.1. Fokus-Zuordnung als Schwierigkeit

Besonders die korrekte Fokus-Zuordnung stellt eine große Herausforderung dar. Hierzu betrachte man folgenden Beispielsatz: „Otto hat die Schere nicht in den Müll geworfen“

Je nach Betonung können unterschiedliche Implikationen „herausgehört“ werden:

Otto hat die Schere nicht in den Müll *geworfen*	>>	sondern vorsichtig reingelegt
Otto hat die Schere nicht *in den Müll* geworfen	>>	sondern aus dem Fenster
Otto hat *die Schere* nicht in den Müll geworfen	>>	sondern das Bastelpapier

2. Daten

Wir arbeiten auf dem manuell annotierten Korpus "Der Hund von Baskerville" von A. C. Doyle im Tiger XML Format. Dieses teilen wir in etwa so auf: 50% fürs Training, 50% zum Testen.

3. Werkzeuge

Verwendete Dritt-Werkzeuge werden zum Einen zwangsläufig ein XML-Parser sein, sowie ein Dependenzparser (Matetools).

4. Vorbereitung

Wir beginnen mit manuell aufgestellten Regeln zur Erkennung von Signalwörtern, Negationsskopus und -Fokus – zur Orientierung unter Zuhilfenahme von Parsern.

5. Unser Tool „WUVV“

5.1. Vorsätze

Unser Tool „WUVV“ soll möglichst plattformunabhängig und modular sowie möglichst vielfältig einsetzbar und komfortabel sein. Dabei soll es mindestens lokal ausführbar sein. Angedacht ist auch ein Webservice mit API sowie eine GUI oder App.

5.2. Architektur

WUVV besteht im Wichtigsten aus 5 Schritten: dem Import sowie Export von bzw. nach Tiger XML, einer Signalwort-Erkennung, einer Skopus-Erkennung sowie einer Fokus-Erkennung.

Die einzelnen Schritte sollen möglichst modular implementiert werden, um Subsysteme mit starker Kohäsion und loser Kopplung zu bilden, damit diese austauschbar und einzeln benutzbar sind.

Zum Zweck der losen Kopplung und Unabhängigkeit der Wahl der Programmiersprache werden diese Subsysteme über definierte Schnittstellen angesprochen, die von Shell-Skripten, die als Wrapper dienen, bereitgestellt werden.

Beispielsweise könnte die Funktion `doesSentenceContainNegation(sentence)` über einen Aufruf des Wrapper-Shell-Scripts so realisiert sein:
`./cue-detector.sh –isnegated "Ein Satz."`

6. Evaluation

Die Evaluation findet auf mehreren Ebenen statt. Es wird jeder der Arbeitsschritte einzeln, und für sich, aber auch insgesamt ausgewertet.

6.1. F-Score

Sowohl für die Auswertung gefundener Cue-Words als auch des Focus ist eine simple Hit-or-Miss Kontrolle möglich.

6.2. Dice-Similarity

Für die Größe des Scopes kann es zu unterschiedlich genauen Parses kommen weswegen hier anteilsmäßige Abzüge bei nicht deckungsgleichen Ergebnissen stattfinden.