AaS - Annotation als Suchproblem

Rebekka Hubert Michael Staniek Simon Will

December 6, 2016

1 Introduction

In diesem Dokument werden die Funktion, die Funktionsweise und die Systemvoraussetzungen der Software „Annotation als Suche“ (AaS) beschrieben.

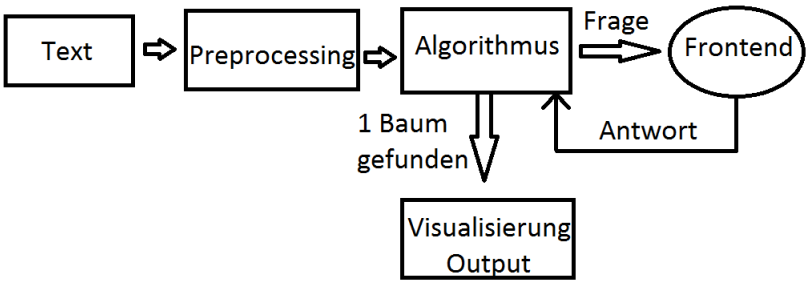
1.1 Motivation

Bei AaS handelt es sich um eine Software zur Unterstützung von Annotatoren, die auf Basis eines Parseforest Fragen an den Annotator stellt und anhand dieser den optimalen Parsebaum auswählt. Dafür benötigt der Annotator zwingend Grunderfahrung im Annotieren, aber keinerlei Programmierkenntnisse. Dies beschleunigt den Vorgang der menschlicher Dependenzannotation, die weiterhin die beste Qualität aufweist.

1.2 Architecture

2 System

Um die gröSStmögliche Kompatibilität und Flexibilität zu erreichen, besteht das System aus formal drei Teilbereichen, die durch Schnittstellen kommunizieren. Dies ermöglicht es, Konkret handelt es sich um eine Server-Client-Konstruktion, die es dem User ermöglicht, sowohl das gesamte System als auch lediglich Teilbereiche zu installieren: So ist das Preprocessing optional und auch der Client kann für einen bereits installierten Server aufgesetzt werden. Im Folgenden ein Überblick über das gesamte System:



Das System besteht bekanntlich aus drei Teilbereichen, die durch Schnittstellen kommunizieren:

- Preprocessing
  Generieren der möglichen Parsebäume und Übergabe der k-besten an die Fragengenerierung über die Schnittstelle Preprocessing - Fragegenerierung
- Algorithmus zur Fragengenerierung auf einem Server
  erhält die k-besten Parsebäume
- Client
  Übermittlung der Fragen an den Annotator und der Antwort des Annotators an den Algorithmus über die Schnittstelle Client - Server sowie vom Prozess unabhängige Darstellung des Parsebaums

2.1 AaS-Server

Der AaS-Server generiet die Fragen, nach welchen der optimale Parsebaum ausgeählt wird. Der Algorithmus folgt dabei folgendem Schema:

- Generiere aus Parseforest die Frage, welche bei ihrer Beantwortung die meisten Bäume rausfiltert.
  - Suche des Tupels, das den Suchraum am ehesten halbiert:

    a $\leftarrow$ len(parses) Anzahl der Parse-Bäume.
    For each tuple
      score $\leftarrow$ abs(count(tuple) $-\ \frac{a}{2}$)
    End For

- Nimm den Tupel als Frage

Näheres im AaSP.

2.1.1 Dependencies

Der Algorithmus benötigt folgende Software um korrekt zu laufen:

- Python $\geq$ 3.4
- Python3 packages: asyncio, Flask
- Parser, der k-best Parses liefert (Anders Bjorkelunds transition-based parser)
- TCP- oder UNIX-Sockets
- JSON

2.1.2 Algorithm

2.2 AaS-CLI-Client

2.2.1 Dependencies

2.3 AaS-Web-Client

2.3.1 Dependencies