Nachnutzbarmachung von Forschungsdaten und Tools am Beispiel altäthiopischer Korpora

Druskat, Stephan
stephan.druskat@hu-berlin.de
Humboldt-Universität zu Berlin, Deutschland

Vertan, Cristina
fsha060@uni-hamburg.de
Universität Hamburg, Deutschland

Einleitung

In den letzten Jahren wurden verschiedene Werkzeuge zur Annotation von Dokumenten entwickelt, z.B. *WebAnno* (Eckart de Castilho et al. 2016), *Cor A* (Bollmann et al. 2014) und *CATMA* (Meister et al. 2016). Diese Werkzeuge bieten zahlreiche Funktionalitäten, die für viele Sprachen und Anwendungen ausreichend sind. Die Unterstützung verschiedener Skripten erweckt den Eindruck, dass diese Werkzeuge für die Annotation aller Sprachkorpora erfolgreich eingesetzt werden können, was solange korrekt ist, wie man sehr flache Annotation durchführt.

Tiefere Annotationen, insbesondere für Sprachen außerhalb der Europäischen Sprachfamilie oder für historische Sprachen, dagegen gestalten sich problematischer. Häufig jedoch wird die Verwendung bereits etablierter Annotationstools trotzdem bevorzugt, weil diese eine reibungslose Integration mit weiteren Analyse- und Visualisierungs-Tools wie z.B. *ANNIS* (Krause & Zeldes 2016) versprechen. In diesem Beitrag werden wir zeigen, dass die Entwicklung dedizierter Annotationswerkzeuge dann als Lösung in Betracht gezogen werden kann, wenn gleichzeitig Schnittstellen (z.B. zu Analyse-Tools) entwickelt werden, die die Neuentwicklung wiederum an vorhandene Software und Infrastrukturen anbinden. Auf diese Weise können die Nachteile einer Neuentwicklung unter Gesichtspunkten der nachhaltigen Entwicklung von Forschungssoftware gegenüber der Anpassung bestehender Tools - bei gleichzeitiger Ermöglichung eines spezialisierten Anwendungsfalles - weitgehend abgeschwächt werden. Der Vorteil eines solchen Verfahrens ist die Realisierung eines Annotationsmodells, das exakt den Besonderheiten der Sprache entspricht. Insbesondere für diachrone Analysen ist es häufig nötig, dass man eine sehr detaillierte Modellierung der morphologischen, syntaktischen und semantischen Merkmale vornimmt, da die Unterschiede häufig nur im Detail reflektiert sind. Im Einzelnen werden wir die Entwicklung des *GeTa*-Annotationstools, eines Mehrebenenannotationswerkzeugs für die Altäthiopische Sprache, und dessen Integration mit dem Such- und Visualisierungsframework *ANNIS* darstellen.

Spezielle Anforderungen für die Altäthiopische Sprache

Das Altäthiopische (Ge#ez), ist eine südsemitische Sprache. Es war bis ins 19. Jahrhundert hinein die bedeutendste Schriftsprache des christlichen Äthiopien. Die reiche christlich-äthiopische Literatur ist zunächst von Übersetzungen - anfangs aus dem Griechischen und später aus dem Arabischen - geprägt, bevor sich eine mannigfaltige indigene Literatur mit ganz eigenen Zügen entwickelt. Insbesondere Texte, die einzig im Altäthiopischen vollständig überliefert, und deren Textzeugen in anderen Sprachen entweder vollständig verloren, oder von denen nur Fragmente erhalten sind (z.B. das Henochbuch) erlangen dabei ganz besondere Bedeutung (Vertan et. al. 2016).

Das Altäthiopische hat aus einer südsemitischen Schrift ein eigenes Silbenalphabet entwickelt, das bis heute in mehreren modernen Sprachen Äthiopiens und Eritreas Verwendung findet. Innerhalb der semitischen Sprachen fällt es durch die verwendete Rechtsläufigkeit auf; außerdem werden die Vokale vollständig geschrieben. Das äthiopische Silbenalphabet bringt dabei mit sich, dass Morphemgrenzen in der Schrift nicht darstellbar sind, sodass beispielsweise ein einzelner Vokal als Bestandteil einer Silbe eine eigenständige Wortart darstellen kann und tokenisiert werden muss; z.B. ist im zweisilbigen Wort ### /be·tu/ das /u/ als pronominales Suffix zu bet- *u* (*sein* Haus) zu segmentieren. Eine solche Annotation kann nur auf der Transkription erfolgen. Annotationen auf anderen Ebenen (z.B. Seiten- Spaltenumbrüche, Textkorrekturen) müssen auf dem Fidal-Skript realisiert werden. Dies bedeutet, dass Original und Transkription synchronisiert im Annotationswerkzeug dargestellt werden müssen. Eine korrekte Darstellung einer Transliteration benötigt die Transkription. Während diese regelbasiert automatisch durchgeführt werden kann, ist korrekte Transliteration (z.B. Konsonantendopplung) in vielen Fällen nur zusammen mit morphologischer Analyse möglich. Daher muss das Annotationstool Korrekturen am Basistext während der Annotation zulassen und zuverlässig verarbeiten können. Keins der existierenden Annotationswerkzeuge erfüllt diese Voraussetzungen. CorA arbeitet mit Listen von Wertkombinationen von Attributen. Die benötigte morphologische Annotation im Fall der altäthiopischen Sprache umfasst 30 Merkmale, mit je mindestens drei möglichen Werten. Die Handhabung solch einer umfangreichen Liste macht manuelle Annotation praktisch unmöglich. Weder WebAnno noch CATMA ermöglichen die Korrektur des zugrunde liegenden Textes während der

Annotation. Die Implementierung einer solchen Funktion auf Basis eines dieser Werkzeuge würde tief in die Architektur der Software eingreifen, was innerhalb der Laufzeit des Projektes nicht realisierbar war. Auch die semi-automatischen Annotationsmöglichkeiten unter Supervision des Benutzenden sind für diese beiden Werkzeuge nicht leicht erweiterbar.

Das GeTa Annotationstool

Im TraCES-Projekt [1], das als Hauptziel die Entwicklung eines diachronen, tief annotierten Korpus für die Altäthiopische Sprache hat, implementieren wir eine neuartige Architektur, die sowohl Änderungen im Text als auch eine Mehrebenenannotation ermöglicht.

Wir betrachten als Grundtext den Originaltext in der altäthiopischen Schrift. Die Transliteration bildet die erste, die morphologische Annotation die zweite Ebene, wobei die Transliteration und der Originaltext bei allen Arbeitsschritten synchronisiert bleiben. Im folgenden Abschnitt beschreiben wir das Datenmodell, das diese Architektur ermöglicht.

Die Basiseinheit in unserem System ist ein Wort, das eine einmalige ID zugewiesen erhält (graphische Einheit). Ein Wort hat folgende Komponenten:

- Eine Liste der einzelnen Fidal [2]-Objekte, wobei ein Fidal-Objekt aus einer ID und einem Label (dem Fidal-Buchstaben) besteht.
- Eine Liste einzelner Silben-Objekte, wobei ein Silben-Objekt aus einer ID und einer Liste von einzelnen Buchstaben-Objekten besteht.
- Ein Buchstaben-Objekt hat immer eine ID und ein Label (das graphische Symbol).

Graphische Einheiten können in mehreren Tokens geteilt werden. Die Tokens sind getrennt gespeichert (mit eigenenen IDs) und verlinkt mit der graphischen Einheit. Elemente der Textstruktur werden durch selbständige Objekte dargestellt, die mit den beinhalteten Tokens verlinkt werden. Editorische Annotationen werden mit den graphischen Einheiten verlinkt. Eine derartig stark vernetzte komplexe Struktur ist mit XML schwierig zu modellieren, insbesondere bei manueller Annotation. Auch die maschinelle Verarbeitung einer derartig tief vernetzten TEI-Struktur wäre sehr kompliziert. Alternativ bietet JSON für dieses Modell eine bessere Handhabbarkeit. Wir haben uns daher entschieden, die Daten in JSON zu speichern und zusätzlich XML-Export zu implementieren. Die Datenstruktur ist in vier JSON-Dateien gespeichert: eine für die graphischen Einheiten, eine für die Tokens, je eine für weitere Annotationsebenen (Textstruktur und Edition). Diese JSON-Dateisammlung wird via Konverterschnittstelle nach *ANNIS* importiert.

corpus-tools.org: *ANNIS*, *Pepper*, *Salt*

ANNIS (Krause & Zeldes 2016) ist eine Such- und Visualisierungsplattform für linguistische Daten, die mehrebenenfähig ist, d.h., verschiedene Annotationsebenen eines Korpus darstellen kann und dabei verschiedenste Annotationsarten berücksichtigt. Dies macht *ANNIS* zu einem hervorragenden Analysetool für die Daten des TraCES-Projekts. Eine besondere Rolle dabei spielt die Mächtigkeit der *ANNIS*-eigenen Abfragesprache AQL (ANNIS Query Language), die neben Freitextsuche und regulären Ausdrücken sich vor allem dadurch auszeichnet, linguistische Strukturen über mehrere Ebenen suchen zu können.

Durch *ANNIS*‘ beschriebene Eigenschaften und seine Konfigurierbarkeit ist die Software für die Verwendung in den unterschiedlichsten Analyseszenarien geeignet und wird in der Tat bereits in verschiedenen Communities verwendet, z.B. in historischer Linguistik, Typologie, Sprachdokumentation u.v.m.

Diese Eignung wird verstärkt durch eine hohe Kompatibilität mit vielen unterschiedlichen linguistischen Datenformaten, die erreicht wird durch Einsatz von *Pepper* (Zipser et al. 2011), einem Konvertierungsframework für liguistische Daten. *Pepper* nutzt ein generisches, graphbasiertes Zwischenmodell, *Salt* (Zipser & Romary 2010), das unterschiedlichste, auch über mehrere Ebenen eines Korpus verteilte, Annotationsarten aufnehmen kann und so weiterverarbeitbar macht. *Pepper* zeichnet sich weiterhin durch eine Plugin-Architektur aus, die es ermöglicht mit geringem Aufwand sowohl Import- als auch Manipulations- und Exportmodule zu entwickeln, die die Quelldaten in ein *Salt*-Modell im Hauptspeicher übertragen, das dort manipuliert werden und anschließend in ein Zielformat überführt werden kann.

ANNIS, *Pepper* und *Salt* sind Komponenten der Softwarefamilie *corpus-tools.org* (Druskat et al. 2016).

Schnittstelle zwischen *GeTa* und *ANNIS*

Diese Infrastruktur ermöglicht es ohne Weiteres, die in *GeTa* annotierten Daten über ein dediziertes *GeTa*-Importmodul für *Pepper* und die Verwendung der bereits existierenden *ANNIS*-Module für *Pepper* – hier das Exportmodul – in *ANNIS* verarbeitbar zu machen. Mit *GeTaModules* (Druskat 2018) haben wir einen solchen Importer entwickelt, dessen Funktionsweise hier kurz beschrieben werden soll. *GeTaModules* ist in Java implementiert und wird als OSGi-Bundle ausgeliefert, das von *Pepper*s OSGi-Plattform verwaltet werden kann. *GeTaModules* ist Open Source unter der Apache License, Version 2.0.

Um Performanz und Speicherökonomie zu optimieren nutzt der *GeTaModules*-Importer eine Kombination aus Streaming und Object-Mapping-Methoden, um die aus *GeTa* exportierten JSON-Dateien einzulesen. Dabei werden die kleinsten Einheiten der Transliteration auf den graphischen Einheiten genutzt, um eine Tokenisierung in *Salt* zu erstellen und einen virtuellen Primärtext aufzubauen. Auf den so erstellten Modelltokens werden rekursiv die Silben- und Fidal-Objekte als Spannen aufgebaut. Im Anschluss werden die linguistischen Annotationen der *GeTa*-Tokens sowie die weiteren Annotationen aus den JSON-Objekten per Identifikator auf die entsprechenden Einheiten projiziert.

In diesem Zustand hält *Pepper* einen kompletten *Salt*-Dokumentgraphen für das entsprechende Korpusdokument im Hauptspeicher. In einem weiteren Schritt werden Ordnungsrelationen jeweils zwischen den Knoten der Fidal und der transliterierten Wortebene erstellt, damit diese später in *ANNIS* als Segmentierungsgrundlage angezeigt werden können.

Im dritten Schritt werden mit Hilfe des *ANNIS*-Exportmoduls [3] die für den Import in *ANNIS* benötigten Dateien im nativen Format geschrieben. Diese können nun in *ANNIS* über dessen graphische Benutzeroberfläche importiert werden.

Die Konfiguration der Visualisierungen erfolgt durch Anpassung einer während des Exportvorgangs generierten Konfigurationsdatei. Im Fall der *GeTa*-Daten müssen hier lediglich die anzuzeigenden Annotationsebenen und ihre Reihenfolge eingestellt werden. In den *ANNIS*-Daten wird weiterhin ein dedizierter HTML-Visualisierer konfiguriert, der Wortartenannotationen auch graphisch Fidalwörtern zuordnet.

Mit *Pepper* erfolgt die Konvertierung auf der Kommandozeile und die Konfiguration in Textdateien. Da dies für einige potenzielle Anwendergruppen unbekanntes Terrain bedeuten könnte haben wir den Prototypen einer Desktopanwendung für *Pepper* für die Verwendung mit *GeTaModules* angepasst, die eine grafische Oberfläche für die Verwendung von *Pepper* bietet: *Pepper Grinder* (Druskat 2017). *Pepper Grinder (TraCES Edition)* bietet so den Workflow für die Konvertierung der *GeTa*-Daten nach *ANNIS* quasi per Knopfdruck an. *Pepper Grinder* wird in Zukunft in eine vollfunktionale Anwendung ausgebaut, die Modi für verschiedene Anwendergruppen anbieten wird.

Neben der projektinternen Nutzung von *GeTaModules* für die Konvertierung in das *ANNIS*-Format eröffnet die Software weitere Nachnutzungsszenarien für die annotierten Daten. Durch die Existenz von Manipulator- und Export-Modulen für *Pepper* für die verschiedensten Datenformate können die Daten mit anderen Werkzeugen nachgenutzt und etwa um zusätzliche Annotationsebenen angereichert, oder mit anderen Visualisierungen zu weiteren Forschungsfragen analysiert werden. Gleichzeitig leistet die Nachnutzung und Anpassung von *Pepper* durch *GeTaModules* einen Beitrag zur Vernachhaltigung dieses Frameworks. Durch die Ermöglichung des Imports von GeTa-Daten in ANNIS wird aus dem GeTa-Modell eine durchaus attraktive Modellierungsalternative für andere Sprachen mit ähnlichen Merkmalen. Derzeit wird das Modell in laufenden Projekten zu Mayasprachen und Jiddisch erprobt.

Zusammenfassend stellt unser Ansatz eine Alternative dar zu Tendenzen der „Format-Hoheit“, also der Modellierung auf Grundlage eines – eventuell de facto standardisierten – Datenformats im Gegensatz zur Modellierung auf Grundlage der Forschungsfrage und der vorliegenden Daten. Die Einrichtung von Schnittstellen, wie z.B. *GeTaModules*, die eine Nachnutzbarkeit der Daten auch über die ursprüngliche Erstellung hinaus gewährleisten können, ermöglicht eine optimierte Modellierung und die Entwicklung spezifischer, den Bedürfnissen der Forschung und ihrer Daten hochgradig angepasster Werkzeuge, wie etwa *GeTa*.

Fußnoten

1. https://www.traces.uni-hamburg.de/ .
2. "Fidal" ist der Terminus technicus für das äthiopische Silbenalphabet.
3. https://github.com/korpling/pepperModules-ANNISModules/ .

Bibliographie

Bollmann, Marcel / Petran, Florian / Dipper, Stefanie / Krasselt, Julia (2014): „CorA: A web-based annotation tool for historical and other non-standard language data“, in: *Proceedings of the 8th Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH)*. Gothenburg, Sweden: 86-90.

Druskat, Stephan (2018): „GeTaModules (Version 0.9.0)“. Zenodo. DOI: 10.5281/zenodo.1146985. http://doi.org/10.5281/zenodo.1146985

Druskat, Stephan (2017): „Pepper Grinder (Version 0.1.7)“. Zenodo. DOI: 10.5281/zenodo.1041735. https://doi.org/10.5281/zenodo.1041735

Druskat, Stephan / Gast, Volker / Krause, Thomas / Zipser, Florian (2016): "corpus-tools.org: An Interoperable Generic Software Tool Set for Multi-layer Linguistic Corpora", in: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016):* 23–28.

Eckart de Castilho, Richard / Mújdricza-Maydt, Éva / Yiman, Seid Muhie / Hartmann, Silvana / Gurevych, Iryna / Frank, Anette / Biemann, Chris (2016): „A Web-based Tool for the Integrated Annotation of Semantic and Syntactic Structures“, in: *Proceedings of the LT4DH workshop at COLING 2016, Osaka, Japan:* 76-84.

Krause, Thomas / Zeldes, Amir (2016): "ANNIS3: A new architecture for generic corpus query and visualization", in: *Digital Scholarship in the Humanities*: 118-139.

Meister, J.C. / Petris, M. / Gius, E. / Jacke, J. (2016): CATMA 5.0 [Software for text annotation and analysis] http://www.catma.de [letzter Zugriff 10.01.2018]

Vertan, Cristina / Ellwardt, Andreas / Hummel, Susanne (2016): "Ein Mehrebenen-Tagging-Modell für die Annotation altäthiopischer Texte", in: *Proceedings der DHd-Konferenz 2016* http://www.dhd2016.de/abstracts/vortr%C3%A4ge-061.html [letzter Zugriff 25.09.2017].

Zipser, Florian / Romary, Laurent (2010): "A model oriented approach to the mapping of annotation formats using standards", in: *Proceedings of the Workshop on Language Resource and Language Technology Standards (LREC 2010)* https://hal.inria.fr/inria-00527799/ [letzter Zugriff 25.09.2017].

Zipser, Florian / Zeldes, Amir / Ritz, Julia / Romary, Laurent / Leser, Ulf (2011): "Pepper: Handling a multiverse of formats", Poster, *33. Jahrestagung der Deutschen Gesellschaft für Sprachwissenschaft*, Göttingen.