

## Protokoll zur Sitzung vom 08.06.2015 – Computerlinguistisches Arbeiten

Im Laufe dieser Sitzung wurden drei Themen in Form eines längeren Vortrages inklusive einer kleinen Präsentation auf dem Beamer von den Studenten vorgestellt.

### 1. Vorstellung: [REDACTED] Entwicklung einer komputationalen Morphologie für die arabische Sprache (zweiter Teil)

Nicolas begann mit einer kleinen Zusammenfassung des ersten Teils seiner Vorstellung, die er bereits in der letzten Stunde begonnen hatte. Die stark flektierende arabische Grammatik und die Verbwurzeln wurden nochmals erwähnt. Der Hauptzweck der Stammbildung der 16 Stämme von Verben im Arabischen, wovon in der Sprache selbst eigentlich nur zehn Stämme verwendet werden, ist die Bedeutungsverschiebung. Es wird noch einmal genauer auf die Aufspaltung dieser Stämme eingegangen, beispielsweise ist der 7. Stamm das Passiv und der zehnte Stamm eine Bitte.

Die technische Umsetzung seines Themas besteht darin, ein Programm zu schreiben, das arabische Wortformen sowohl erkennen als auch generieren kann. Als Programmiersprache wird SFST benutzt, ausgeschrieben: Stuttgart Finite State Transducer. Diese wurde von dem Betreuer seiner Arbeit, Herrn Dr. Schmid mitentwickelt. Diese endlichen Transduktoren sind quasi endliche Automaten, aber, im Unterschied zu diesen, haben sie ein Ein- und Ausgangsalphabet. Es basiert auf regulären Ausdrücken, ähnlich wie bei der Programmiersprache Perl. Es gibt Zeichenketten, Operatoren, Variablen nur mit einem Wert und Agreement-Variablen, in diesen muss ein Wert jeweils in zwei oder mehr Variablen gleich sein. Die Struktur der Implementierung wird anhand eines Beispiels an der Tafel gezeigt. Es wird demonstriert, wie ein endlicher Transduktor arbeitet, indem er das doppelte -t des zweiten Stamms vom arabischen Wort „qatala“ (hier im zweiten Stamm: „qattala“) löscht und dafür nichts mehr ausgibt. Die Variable des Stamms wird somit einfach gelöscht, falls sie auftritt, und der pure Stamm wird ausgegeben. Ein großes Problem dieser Arbeit besteht auch darin, dass die arabische Schrift erst in das lateinische Alphabet transkribiert werden muss.

## 2. Vortrag: Reduktion von OCR-Fehlern durch Kombination der Ergebnisse zweier OCR Programme

In der heute vorgestellten Bachelorarbeit von Josef Pötzl mit Herrn Dr. Springmann als Themensteller, werden die Resultate der zwei OCR Maschinen "Ocropus" und „Tesseract“ verglichen, mit dem Ziel, die Fehler in den Ausgaben zu finden und zu korrigieren, bzw. dadurch zu reduzieren. Ein OCR-Programm digitalisiert Handgeschriebenes. Ähnlich wie beim von zwei unabhängigen Personen manuell gefertigtem „Double-Keying“ wird dieser Vergleich zweier unabhängig erstellter Transkriptionen der OCRs zur Reduzierung von Fehlern verwendet. Hier wird das teure, manuelle Annotieren der Daten von der regelbasierten Methode dieser zwei OCR Programme abgelöst. Aber es gibt hier noch viele Fehler zu beseitigen: An den Stellen, an denen Wörter unterschiedlich ausgegeben werden, steckt dann meistens ein solcher Fehler in der Ausgabe. Man muss die Ausgaben außerdem mit Zeilen versehen, und die Bilder herausfiltern, um ähnliche Zeilenangaben bei beiden Daten zu erhalten. Bei dieser Arbeit empfiehlt es sich, eine Kombination aus mehreren gängigen Verfahren, wie z.B. Klassifikatoren und unterschiedlichen Methoden zum Vergleich der beiden OCR-Resultate zu verwenden.

Er beginnt mit einer Geschichte der OCR-Programme, die in den 50er Jahren begann, und sich bis heute immer weiter entwickelt. Die Vorteile solcher Programme liegen auf der Hand: Es wird digital auf dem Computer bei weitem weniger Platz benötigt, als bei den mehr als 200 Millionen „analogen“ Büchern, die pro Jahr entstehen. Digitalisiert sind die Inhalte leichter verteilbar, leichter bearbeitbar und leichter zu durchsuchen, man muss anstatt lange herumzublättern nur noch den jeweiligen Suchbegriff eingeben, und hat im Idealfall das gesuchte Ergebnis. Das Problem ist: Computer können nur mit Daten arbeiten, d.h. die Bücher müssen erst richtig eingelesen und digitalisiert werden. In einigen Schritten wird aus einem Bild eine digitale Datei:

Am Anfang steht die Binarisierung, das Bild wird in schwarz-weiß eingescannt. Hierauf folgt die Segmentation, Text und Bilder werden zu einzelnen Zeichen isoliert. Das Problem hier sind Zeichen, die unvollständig sind, oder verschmutzte Bilder. Auch wenn der Text mit Grafik vermischt ist, gibt es Erkennungsprobleme. Das Preprocessing verringert diese Probleme, und mithilfe von Isolierung und Glättung wird bei den bereits isolierten Zeichen das Erkennen vereinfacht. Als vierter Schritt kommt die Feature Extraction, hierbei werden die Eigenschaften der Zeichen erfasst, z.B. ob es Punkte gibt, und andere verschiedene Kriterien. Als vorletzter Schritt ist die Klassifikation, also die Identifizierung der Buchstaben an der Reihe. Zum Schluss folgt die „Error

Detection & Correction“, damit beschäftigt sich Joseph hauptsächlich in seiner Bachelorarbeit. Es gibt zwar schon OCR-Programme mit 99%iger Genauigkeit, jedoch das Ziel ist es, mit einer automatischen Fehlerkorrektur noch mehr zu erreichen, da Menschen zum Korrigieren immer teurer sind als Maschinen. Das Ziel ist es jedoch, mit möglichst wenig Kosten eine möglichst hohe und korrekte Leistung für zukünftige Ergebnisse zu erzielen. Es gibt zwei Arten von Fehlern: Real-Word-Errors: Wörter, die nicht mit dem eingescannten Bild übereinstimmen, aber in einem Lexikon existieren, und Non-Word-Errors: Dies ist der häufigere Fehler, es sind Wörter, die weder auf dem Bild, noch in einem Lexikon existieren. Für seine Arbeit benutzt er 20 Seiten als Trainingsset und 20 Seiten als Testset aus „Der Grenzbote“, eine Zeitschrift, die 1841-1922 auf Deutsch herausgegeben wurde. Bei seiner Arbeit geht er folgendermaßen vor: Erst wird die Seite mit Scantailor, einem Bildbearbeitungsprogramm eingescannt und digital nachbearbeitet, dann werden mit Ocropus die Zeilen einzeln segmentiert. Die Dateien werden mit Ocropus und Tesseract Zeile für Zeile durchgegangen, und die beiden Ausgaben auf Übereinstimmungen getestet. An den Stellen, an denen sie sich unterscheiden, wird sein Logarithmus zur Korrektur angewendet. Er hat sich ein Java Tool implementiert, um die jeweils abweichenden Stellen mit einem „@“ zu markieren. Wenn beide ausgegebenen Wörter nicht im Lexikon stehen, wird nach dem Confidence-Wert der Worte das wahrscheinlichere ausgegeben. Joseph hat tatsächlich noch eine kleine Verbesserung gegenüber der Original OCR-Ausgabe erreicht. Da der Score aber bereits so hoch ist, kann man ihn nur noch mit Einzelfällen, in denen man sich 100% sicher ist, verbessern.

### **3. Vortrag: [REDACTED], Ein elektronischer Apparat für Wittgensteins Nachlass**

Im letzten Vortrag des Tages von Pascal Zambito geht es um die Erstellung eines Apparates, um Wittgensteins Nachlass zu digitalisieren und den Menschen in digitalisierter Form zugänglich zu machen. Die ist keine Bachelorarbeit, sondern ein Projekt von ihm und Daniel Bruder, es wird von Dr. Maximilian Hadersbeck und Michael Nedo geleitet. In einem kurzen Lebenslauf von Wittgenstein wird sein Werdegang beschrieben. 1889 in Wien geboren, studierte er Maschinenbau und Philosophie, im 1. Weltkrieg hat er freiwillig für Österreich gedient. Es gibt verschiedene Phasen des Schreibens in seinem Leben, der frühe Wittgenstein ist eher philosophisch und abstrakt, der späte geht vom Stil eher in Richtung „pragmatisch und alltagsbezogen“. Die meisten seiner logisch-philosophischen Abhandlungen sind erst nach seinem Tod 1951 beim Suhrkamp Verlag erschienen. Er hat sehr viele Manuskripte, Typoskripte und Vorlesungsmitschriften verfasst, die alle in Beziehung zueinander gesetzt werden müssen. Nach seinem Leben und Werk folgt die Beschreibung des

Korpus für das Projekt selbst: Es besteht aus Wittgensteins Schriften „Philosophische Bemerkungen“ von 1964, vier Manuskripte sind der Grundstock, diese einzelnen Arbeitsstufen sind in dem gebundenen, herausgegebenen Buch jedoch nicht mehr sichtbar. Die Aufgabe ist es nun, die einzelnen Manuskripte, Streichungen, Bemerkungen und Einfügungen wieder hinzuzufügen, und deren Beziehungen untereinander erklärlich darzustellen. Das Konzept dieser sogenannten „Wiener Ausgabe“ besteht nun darin, eine philologisch korrekte Ausgabe zu liefern, die eine gute Lesbarkeit mit sich bringt, und alle Bezüge zu den Manuskripten nachvollziehbar erklärt. Es soll dazu eine lemmatisierte Suche inklusive aller Streichungen und Einfügungen entwickelt werden. Eine Verknüpfung der Daten wurde bereits Anfang der 90er Jahre in einer Excel Tabelle erfasst, daraus gilt es jetzt, eine HTML Datei zu erstellen, die durchsuchbar und editierbar sein soll. Sein Projektpartner Daniel Bruder hat zusätzlich dazu eine Datenbank erstellt. Man muss den Korpus zusammenfassen und als eine abstrakte Grammatik überführen. Dazu wird ein „abstract-syntax-tree“ erstellt. Das Identifizieren der einzelnen Objekte erfordert ein ideelles Denken, und die Dokumente als einzelne Klassen zu sehen. Als Klassen fungieren beispielsweise: Manuskript, Typoskript und Vorlesungsmitschrift. Jedes dieser Objekte hat einen spezifischen Namen und eine ID zum Identifizieren des jeweiligen Objekts. In dieser ID ist z.B. enthalten, wo es sich im Text befindet, oder welche Vorgänger/Vorstufen, Nachfolger es besitzt. Dies ist die Besonderheit bei Wittgenstein: Aufgrund der verzweigten Schriften kann man ihn auch „quer“ lesen, sie gehen auch in die Breite. Als Ausblick für weitere Arbeitsschritte werden das Auswerten der Excel Tabelle und das Erstellen einer Datei mit den Dokumenten und Bemerkungen als Objekten genannt. Zusätzlich soll die Webdarstellung überarbeitet werden. Das Projekt sucht außerdem immer neue Interessenten.

### **Formal:**

- \* übersichtliche Aufteilung**
- \* Name d. Referenten, Thema, BA- Betreuer**
- \* Rechtschreibung, Kommasetzung**

### **Inhaltlich:**

- \* ergebnisorientiert und nachvollziehbar**
- \* ausreichende Erläuterungen und Beispiele**