Protokolle zur Sitzung vom 22.05.2017 - Computerlinguistisches Arbeiten

23.05.2017

Drei Studenten stellten in der Sitzung am 15.05.2017 mit Hilfe einer Präsentation mit dem Beamer ihre Bachelorarbeitsthemen vor.

1. Präsentation:

Referentin: Faridis Alberteris Azar

Titel: Optimierung der linguistischen Suche beim XML-annotierten Nachlass von Ludwig Wittgenstein

Betreuer: Dr. Maximilian Hadersbeck

Faridis erstellt ihre Bachelorarbeit im Rahmen des Digital-Humanities-Projekts "Wittgenstein in Co-Text" in Zusammenarbeit mit der Universität Bergen. Ziel der Arbeit ist die Optimierung der linguistischen Suche beim XML-annotierten Nachlass von Wittgenstein durch die optimale Ausnutzung der XML-Annotation und die Verbesserung der XML-Edition. Speziell kümmert sich Faridis um die Verbesserung der Namensfindung. Zum Nachlass Wittgensteins gehören unveröffentlichte Type-Skripte und Manuskripte, welche im Archiv in Bergen verwaltet werden.

Es gibt hierbei drei verschiedene Datentypen: "ORG", "NORM" und "DIPLO". Faridis zeigt in diesem Zusammenhang ein Beispiel ihres XML-Codes aus dem Nachlass. Hier hat Wittgenstein bspw. das Wort "Schmerzen" erwähnt und erst im Nachhinein das Wort "Zahn" hinzugefügt, jedoch ohne "Schmerzen" kleinzuschreiben. Der Unterschied der drei Datentypen besteht nun im Folgenden:

"ORG" enthält alle möglichen Kombinationen der Schreibweisen eines solchen Beispiels; "NORM" enthält die normalisierte Variante, also das kleingeschriebene "schmerzen"; "DIPLO" enthält das großgeschriebene Wort.

Für die technische Umsetzung der Optimierung der Namensfindung benutzt Faridis einen probabilistischen POS Tagger, bei welchem unter anderem Herr Schmid mitgearbeitet hat. Dieser basiert auf Markov Modellen. Hier zeigt sie ein Beispiel aus einer "NORM"-Datei, wo "Tolstois Erläuterung" nicht als Eigenname getaggt wird. Der Tagger taggt nun diese Dateien und generiert eine XML-Datei "norm_tagged.xml". Diese Datei enthält zwei Attribute: Tag und Lemma. Das Beispiel "Tolstois" wird aber immer noch falsch getaggt, Faridis vermutet hierbei die Wortendung. Auch mit den aktualisierten Daten die im März aus Bergen kamen, wurde dieses Beispiel immer noch falsch getaggt.

Faridis arbeitet in ihrer Bachelorarbeit an Vorschlägen und Möglichkeiten Eigennamen besser erkennen zu können. Um diese in der "norm.xml" Datei zu lokalisieren, sammelt sie alle möglichen Fehler des Taggings mit Hilfe einer Schnittstelle von Python: "The Element Tree". Ein weiterer Schritt besteht in der Verbesserung der semantischen Suche in WittFind, einem Suchprogramm für seinen digitalisierten Nachlass. Hier sucht Faridis mittels grammatikalischer Muster und erhält Vorschläge für Eigennamen, jedoch resultiert dies auch in einige Fehler. Ihr Vorschlag ist nun die Einführung einer neuen syntaktischen

Kategorie “persName”, welche zum Muster hinzugefügt wird, z.B. “Russellschen, Russel.EN+persName”. Abschließend stellt Faridis ihre Resultate dazu vor. Beim Testen von zwanzig Dateien findet “WittFind” aktuell 168 Treffer von Eigennamen. Das von Faridis empfohlene System findet 833 Eigennamen. Einige weitere Kapitel ihrer Arbeit sind das Vergleichen von Transkriptionsfehlern und Editionsproblemen bei XML-Dateien im Wittgenstein-Nachlass, die Verbesserung der Tokenisierung, des Taggings und die Erweiterung des Lexikons.

2. Präsentation:

Referentin: Iuliia Khobotova

Titel: Comparing representation learning over word-level, character-level and combination of both in NLP tasks

Betreuer: Wenpeng Yin

In der Bachelorarbeit von Iuliia geht es um den Vergleich von gewissen Inputstyles von neuronalen Netzwerken und wie diese die Accuracy beeinflussen, kurz: welcher Inputstyle erzielt die höchste Accuracy – Word-Embeddings oder Character Embeddings. Außerdem untersucht Iuliia welches das beste Parameterset für CNN ist, wie schnell die Accuracy berechnet wird und was die Kombination von Word- und Character-Embeddings verändert.

Für ihre Arbeit verwendet Iuliia zwei verschieden Arten von neuronalen Netzwerken: CNN (“Convolutional Neural Network”) und RNN (“Recurrent Neural Network”). Es handelt sich hierbei um weit verbreitete neuronale Netzwerke die große Verwendung in verschiedensten NLP Tasks finden. Der Unterschied besteht darin, dass bei CNN nur ein Output ausgegeben wird, bei RNN jedoch so viele Outputs wie Inputs vorhanden sind.

In Iuliias Experimenten verändert sie die Parameter des CNN. Dabei wird “embedding size”, “hidden size” und “batch size” verändert. Das Ziel besteht nun diese Parameter so zu verändern, sodass daraus die höchste Accuracy resultiert.

Die verwendeten Daten stammen aus der “Stanford Sentiment Treebank”, welche annotierte Daten von Filmrezensionen beinhaltet und aus ca. 215.000 verschiedene Phrasen besteht. Die Implementierung geschieht mittels “Theano” (Theano Development Team 2016), einem Python Framework. Dieses besteht aus Input Layer, Hidden Layer und Output Layer. Während Input- und Output Layer gleich bleiben, verändern sich die Hidden Layer im neuronalen Netzwerk. Abschließend erwähnt Iuliia, wie sie die Evaluation und die Ergebnisse durchführen wird. Die Evaluation geschieht mittels statistischem Vergleich der Accuracy der verschiedenen Inputstyles und die Ergebnisse wird Iuliia graphisch in ihrer Bachelorarbeit präsentieren.

3. Präsentation:

Referent: Alexander Vordermaier

Titel: Comparison of Transfer Methods for low Ressource Morphology

Betreuer: Katharina Kann

Alexander führt in das Thema seiner Bachelorarbeit mit der Motivation ein. Es geht im Grunde um die Zuordnung eines Lemmas zu seinen flektierten Formen (Paradigmen-Komplettierung). Für Sprachen, bei denen nur begrenzte Ressourcen vorhanden sind, ist dies oft schwierig, deshalb stellt sich die Frage: Kann man eine ähnlich Sprache verwenden um das Problem zu umgehen?
Bei der letzten Sitzung wurde dasselbe Thema bereits von Kristina Smirnov vorgestellt. In der Arbeit von Alexander werden jedoch nicht die Sprachen Russisch und Ukrainisch verwendet sondern die “high-ressource” Sprache Bulgarisch und die “low-ressource” Sprache Mazedonisch. Für die Paradigmen-Komplettierung verwendet Alexander drei Methoden: 1. Sprachübergreifende Paradigmen Komplettierung, 2. Auto Encoding und 3. die Kombination aus den beiden Methoden.
Wie funktionieren nun diese Methoden: Bei der sprachübergreifenden Paradigmen-Komplettierung sucht man sich eine “high-” und eine “low-ressource” Sprache. Anschließend holt man sich eine große Menge für “high-” und eine kleine Menge “low-ressource” Daten.
Auf die Frage von Herrn Schulz, wie ähnlich die Sprachen Bulgarisch und Mazedonisch sind, erklärt eine Kommilitonin, dass die Morphologie in den beiden Sprachen ähnlich sei, nur die Wörter sich teilweise sehr unterscheiden.
Auto Encoding ist ein simples Verfahren, bei dem die Eingabe gleichzeitig die Ausgabe ist. Man hofft also, dass viele Flektionen der Wörter gleich sind. Jedoch bestünde große Gefahr, dass dies nicht der Fall sei.
Bei der Kombination der beiden Methoden vermischt man annotierte Daten der “low-ressource” Sprache mit annotierten Daten der “high ressource” Sprache und mit nicht annotierten Daten der “low-ressource” Sprache. Hierbei gibt es verschiedene Datengrößen. Bei “low-ressource” Sprache gibt es 50 bzw. 100 annotierte und 50 – 12800 nicht annotierte Datensätze. Bei der “high-ressource” Sprache gibt es ebenfalls 50 – 12800 Datensätze. Diese Daten bestehen aus Test-, Development- und Trainingsdaten sowie aus Vokabulare zu “source” (Daten, die das Modell erhält) und “target” (Lösung).
Anhand eines Beispiels geht Alexander genauer auf den Aufbau eines solchen Datensatzes ein und erklärt die hier vorkommenden Tags. Zudem zeigt er auf wie der Output nach dem Auto Encoding aussieht. Die Ergebnisse seiner Arbeit werden in einem Diagramm vorgestellt, wobei die Accuracy der einzelnen Methoden, bei steigender Datenmenge aufgezeigt wird. Wie erwartet liefert das Auto Encoding die schlechtesten Ergebnisse. Bei der Auswertung von 50 Datensätzen erzielt die Paradigmen-Komplettierung die beste Accuracy mit ca. 50 %. Bei Verwendung von 200 Datensätzen wird eine Accuracy von 70 % bei der Paradigmen-Komplettierung und fast 80 % bei der Kombination der Methoden erreicht.
Als Abschluss präsentiert Alexander seine Fehleranalyse. Dabei stellt er fest, dass oft die falsche Endung

verwendet wird und beim Auto Encoding viele Fehler, auf Grund der Vorgehensweise, auftreten. Weitere Aufgaben seiner Bachelorarbeit bestehen in der Auffindung weiterer Fehlerquellen, sowie der Beleuchtung gängiger Verfahren. Außerdem möchte er sich das Modell anschauen und dies wenn möglich verbessern.