Jakob Sharab
CIS, LMU München
Matr.Nr.: 11186993

Kolloquium zu Computerlinguistisches Arbeiten -

Protokoll zur Sitzung vom 25.05.2017

Präsentation 1: Optimierungen der linguistischen Suche beim XML-annotierten Nachlass von Ludwig Wittgenstein

Die erste Präsentation wurde von Faridis Alberteris Azar gehalten, deren Arbeit von Herrn Hadersbeck betreut wird. Zunächst erzählte sie uns von der Suchmaschine „WittFind“, die im Rahmen des Digital Humanities Projekts „Wittgenstein in Co-Text“ und in Zusammenarbeit mit der Universität Bergen in Norwegen entwickelt wurde. Diese Suchmaschine bezieht sich auf den digitalisierten Nachlass des Philosophen Wittgenstein. Faridis erklärte, dass es Ziel ihrer Arbeit sei, die linguistische Suche beim XML-annotierten Nachlass Wittgensteins zu optimieren.
Des Weiteren erklärte sie, dass es drei verschiedenen Arten von Wittgenstein Dokumenten gibt und zeigte uns dazu die jeweiligen Manuskripte. Die erste Art sind die originalen Dokumente, welche z.B. auch durchgestrichene Wörter enthalten, die von Wittgenstein verbessert wurden. Die zweite Art von Dokumenten sind die sog. „DIPLO-“ Dokumente, deren Inhalt so ist, wie Wittgenstein ihn schlussendlich veröffentlicht hat. Die dritte Art sind die normalisierten Dokumente, welche Faridis auch in ihrer Arbeit nutzt.
Um ihre Verbesserungen an der Suchmaschine durchzuführen, nutzt sie einmal einen probalistischen POS-Tagger und einen Treetagger, der Entscheidungsbäume für das Messen der Übergangswahrscheinlichkeiten erstellt. Faridis führte aus, dass ihr Fokus auf der Verbesserung der Eigennamen-Erkennung liegt, da dabei momentan noch fehlerhafte Ergebnisse ausgegeben werden und es zu *sparse-data* Problemen kommt, da es oft so ist, dass Eigennamen immer in einer etwas anderen Schreibweise auftreten. Der Ablauf beim Erkennen der Eigennamen soll so sein, dass die normalisierten Dateien getaggt werden und eine Datei generiert wird, die 2 Attribute enthält, nämlich einmal das Tag und das Lemma des jeweiligen Eigennamens.
In einem ersten Schritt will Faridis Eigennamen in den normalisierten XML Dateien lokalisieren. Dazu hat sie alle möglichen Fehler, die beim Tagging auftreten, mit Hilfe einer Schnittstelle zu Python gesammelt. Nachdem ihr dies gelungen war hat sie die gefundenen Eigennamen in einer Liste gespeichert, einmal mit dem Namen als Token und einmal als Lemma.
Im zweiten Schritt hat sie sich die Ergebnisse von „WittFinds“ Eigennamen-Erkennung angeschaut und festgestellt, dass es aktuell noch zu Fehlern kommt, bei denen beispielsweise das Wort „hellenistisch“ als adjektivische Verwendung des Namens „Hellena“ erkannt und so als Eigenname getaggt wird. Ihr Vorschlag war deshalb, um solchen Fehlern vorzubeugen, eine neue syntaktische Kategorie „persName“ zur Unterscheidung einzuführen, die sowohl in „WittFind“ als auch im CIS-Lexikon erzeugt werden soll. Das Resultat, dass dabei herauskam, war dass „WittFind“ in 13 Dateien, wo es vorher 186 Treffer ausgab, nun über 800 Eigennamen markierte. Die Erhöhung der Anzahl der gefundenen Eigennamen erstaunte vor Allem Herrn Schulz.
Zum Schluss gab Faridis einen Ausblick auf mögliche weitere Kapitel ihrer Arbeit. Darunter waren z.B. das Beheben der Transkriptionsfehler in den Dateien im Nachlass Wittgensteins und die Erweiterung

des Lexikons, wenn man die Frequenzliste der Eigennamen mit Hilfe des Treetaggers, anstatt mit regulären Ausdrücken, erzeugt.

Präsentation 2: Comparing representation learning over word-level, character-level and combination of both in NLP tasks

Der zweite Vortrag wurde von Iuliia Khobotowa mit Wenpeng Yin, als ihr Betreuer, gehalten. Zuerst stellte sie die Ziele ihrer Arbeit vor, nämlich herauszufinden welchen Einfluss verschiedene Arten von Inputs die Accuracy von Convolutional Neural Networks (CNN) haben. Dazu gehören die Fragen, welches Set von Paramertern, das Beste ist, wie schnell die Accuracy berechnet wird und inwiefern sich die Accuracy verändert, wenn man beides, Word- und Characterembeddings, miteinander kombiniert. Dazu wolle sie sich deren Ergebnisse in verschieden Aufgaben im Bereich des NLP anschauen.
Als nächstes ging Iuliia kurz auf Convolutional- und auf Recurrent Neural Networks ein, welche am weitesten verbreitet sind, um NLP Aufgaben zu bewältigen und sie zeigte uns kurz zwei Schaubilder, die den Aufbau dieser Netze zeigte.
Anschließend erläuterte sie den Aufbau der Experimente, die sie durchführen will. Dabei will Iuliia die verschiedenen Parameter für das CNN verändern, nämlich die Größe der Embeddings, die Größe der Hidden Layer und die „batch" Größe. Das Ziel hierbei ist es, herauszufinden welche Parameter die höchste Accuracy liefern.
Die Daten die sie verwendet stammen aus der „Stanford Sentiment Treebank" und enthalten annotierte Daten von Filmkritiken mit ca. 215.000 einzigartigen Ausdrücken.
Ihre Evaluierung der Ergebnisse, welche sie graphisch in ihrer Arbeit aufarbeiten will, werden auf dem Vergleich der Accuracy basieren.

Präsentation 3: Comparison of Transfer Methods for low Ressource Morphology

Die dritte Präsentation wurde von Alexander Vordermaier gehalten, dessen Arbeit von Katharina Kann betreut wird. Alexander begann zunächst die Motivation hinter seiner Arbeit zu erklären, nämlich die Paradigmen Komplettierung von Sprachen, d.h. die Zuordnung eines Lemmas zu seinen flektierten Formen, was für Sprachen, bei denen nur wenige Daten zur Verfügung stehen, sehr schwierig ist. In seiner Arbeit möchte er die Frage erörtern, ob es möglich ist für eine Low Ressource Sprache, eine ähnliche High Ressource Sprache zu verwenden, um gute Ergebnisse zu erzielen. In seinem Fall entspricht Bulgarisch der Low Ressource Sprache und Mazedonisch der High Ressource Sprache.
Als nächstes stellte er seine Herangehensweise vor, bei der drei verschiedene Methoden der Paradigmen Komplettierung verwendet werden. Die erste Methode ist die Sprachübergreifende Paradigmen Komplettierung, bei der man die Daten der Low Ressource Sprache mit denen der ähnlichen High Ressource Sprache vermischt, mit der Hoffnung brauchbare Resultate zu erhalten. Dabei wird immer eine Kombination aus Wörtern der Sprache, für die es kaum Daten gibt, immer mit einer anderen Anzahl von Wörtern, für die es viele Daten gibt vermischt und ein Modell trainiert. Die zweite Methode, das Auto Encoding, erklärte Alexander, sei ein simples Verfahren, bei dem die Eingabe auch gleichzeitig die Ausgabe ist, wobei man hofft, dass möglichst viele Flektionen der Wörter gleich sind, was aber meist nicht der Fall ist. Bei der dritten Methode werden die ersten beiden Methoden miteinander kombiniert, d.h. dass annotierte Daten der Low Ressource Sprache mit immer einer anderen Anzahl einmal annotierter und einmal nicht annotierten Daten der High Ressource

Sprache kombiniert werden.
Anschließend erklärte Alexander, wie die Daten aussehen. Dabei gibt es für die Trainings,- Development- und Testdaten jeweils immer eine „source“ Datei, welche dem Modell übergeben werden, und eine „target“ Datei, in der die Lösung steht. Die „source“ Dateien enthalten Informationen über die Sprache des Wortes, der Art des Wortes, dem Tag und dem Lemma, das übergeben wird.
Danach stellte Alexander einen Teil seiner Ergebnisse, nämlich einmal mit einer Kombination von 50 Low Ressource Daten und dann mit 200, vor. Dabei konnte man im ersten Fall sehen, dass das Auto Encoding wie erwartet kein gutes Ergebnis mit einer Accuracy von 20% erzielte, die Paradigmen Komplettierung dafür aber eine Accuracy von 50% erreichte. Die Kombination aus beiden Methoden erzielte ein etwas besseres Ergebniss, was jedoch immer noch nicht perfekt ist. Mit der Kombination aus 200 Low Ressource Daten, waren die Ergebnisse insgesamt besser als davor, wobei das Auto Encoding wieder am schlechtesten und die Kombination aus beiden Methoden am besten abgeschnitten hatte.
Daraufhin gab Alexander eine kurze Fehleranalyse mit den möglichen Ursachen für Fehler die aufgetreten sind, wie zum Beispiel die häufige Verwendung der falschen Endung. Er merkte jedoch an, dass es für ihn nicht einfach sei alle Fehlerquellen zu finden, da er weder Bulgarisch noch Mazedonisch beherrsche.
Zum Schluss gab er noch einen Ausblick auf weitere Aufgaben, die er noch bewältigen muss, wie zum Beispiel weitere Fehlerquellen zu identifizieren und das Modell, welches in seiner Arbeit verwendet wird, noch genauer zu verstehen.