CIS, LMU München

Studentin: Elena Atanasova

12.06.2017

Protokoll zur Sitzung vom 12.06.2017 – Computerlinguistisches Arbeiten

1. Vortrag: Tobias Ramoser, "Phonologically-Enhanced Character Embeddings"

BA-Betreuer: M.Sc. Martin Schmitt

Den ersten Vortrag des heutigen Tages hält Tobias Ramoser, die von Herr Schmitt betreut wird. Am Anfang der Präsentation stellte er seine Ideen für die Gliederung seiner Bachelorarbeit vor und danach präsentierte er die Motivation. Die Motivation ist, Anwendungen der maschinellen Sprachverarbeitung nicht mehr aus unserem Alltag wegzudenken. Bsp. Maschinelle Übersetzung, Spracherkennung , etc. Die meisten Ansätze beschäftigen sich mit Wörter.

Das Ziel seiner Arbeit ist Erstellung von verschiedenen Vektorrepresentationen von Buchstaben mit phonologischen Features und Vergleich mit Zufallsvektoren bei der Transkription in SAMPA.

Thematischer Hintergrund: Phonetik und Phonologie, Word2Vec, SAMPA-Alphabet.

Ramoser erklärte den Unterschied zwischen Artikulation und Stimmhaftigkeit. Wie ein Laut gebildet wird, wird Artikulationsart genannt (plosiv, frikativ, nasal, lateral, Vibranten, Aproximanten). Der Artikulationsort ist es, wo ein Laut gebildet wird (bilabial, labiodental, alveolar, dental, velar, glottal). Wenn der Kehlkopf bei Aussprache eines Lautes vibriert, laut der Stimmhaftigkeit ist dieser Laut stimmhaft, sonst ist es stimmlos.

Danach erklärte Ramoser kurz über Phonologie. Die Phonologie beschreibt die Systematik der Laute innerhalb einer Sprache, in seinem Fall Deutsch. Der vergleicht „rot" und „tot". Die Wörter unterscheiden sich in genau einem Laut, welcher wortunterscheidend wirkt (kontrastierende Eigenschaft). Man spricht nun von Phonemen: Schreibweise in [...]. Die Phoneme müssen sich in einer phonetischen Eigenschaft unterscheiden.

Word2Vec ist ein Programm zur automatischen Vektorerstellung von Wörtern. Der Input sind Trainingsdaten(Text mit vielen Wörtern) und der Output Wortvektoren und Distanz zu einem Wort. Die Verarbeitung wird durch neuronales Netz, die aus Architektur und Lernalgorithmus besteht und die ähnliche Wörter werden ähnlichen Vektor erhalten.

Ramoser benutzte zwei Architekturen und zwar Continuous Bag-of-Words Modell (sagt aus einem Kontext ein gewisses Wort voraus) und Skip-Gram Modell (sagt den Kontext aus gegebenen Wort voraus). Beide haben zweischichtiges neuronales Netz.

In seiner Bachelorarbeit benutzte Ramoser das SAMPA-Alphabet. Es handelt sich um ein auf ASCII-basiertes, maschinen-lesbares, phonetisches Alphabet, mit welchem die Aussprache der Laute dargestellt wird. Dieses Alphabet wurde zwischen 1987 und 1989 entwickelt, um phonemische Transkriptionen deroffiziellen Sprachen der damaligen Europäischen Gemeinschaft übermitteln und verarbeiten zu können.

Ramoser führt vier Experimente durch und zwar: 2 Implementierungen für Vektorenerstellung + Zufallsvektoren, Word2Vec Vektoren und Transkription in SAMPA.

Seine eigene Implementierung wird „Char-Vectors" genannt. Ramoser verwendet die bereits definierte phonologische Features und die Unterkategorien. Der Vektor wird mit 5 Features initialisiert: (0,0,0,0,0) und danach werden die Phonemvektoren berechnet. Immer wenn Eigenschaft

auf Phonem zutrifft erhält es den dementsprechenden Vektor. Die Buchstabenvektoren werden auch berechnet, indem den Durchschnitt aller entsprechenden Phonemvektoren bereechnet wird. Die zweite Implementierung heißt „One-hot Vektoren“. Hier werden die gleiche Features und Unterkategorien wie bei Char-Vectors verwendet. Allerdings handelt es sich hier um eine binäre Klassifizierung. Die Vektorgröße „wächst“ somit auf 22 Dimensionen. Die Implementierung mit Zufallsvektoren wird „Randomized Vectors“ genannt. Es werden 2 Arten von Buchstabenvektoren generiert: 15 Dimensionen und 100 Dimensionen. Dafür wird das „random“-numpy-Moduls verwendet.

Beim Word2Vec Vektoren bestehen die Trainingsdaten aus Buchstaben, anstatt aus Wörtern. Phonologisch ähnliche Buchstaben werden aufeinander trainiert.

Für Bewertung des Experimentes wird die Genauigkeit berücksichtigt. Das beste Ergebnis zeigt random Experimente mit der 100-Version. Um diese Ergebnise zu bestätigen, wird eine Fehlerquote mittels Levenshtein-Distanz berechnet. Es besagt, wie viele Korrekturendurchschnittlich gemacht werden mussten. Das bestätigt die bereits erhaltene Ergebnis derquantitativen Auswertung.