Aufgabenstellung für das Praktikum

„Extraktion von nichtdeterministischen Streaming-String-Transducern aus einem bilingualen Korpus“

Technische Universität Dresden
Fakultät Informatik

Student:	Alexander Caspar Jenke
Geburtsdatum:	14. Juli 1996
Matrikelnummer:	4503044
Studiengang:	Informatik (Master)
Immatrikulationsjahr:	2015
Modul:	INF-VERT2
Studienleistung:	150h (6LP)
Beginn am:	1. Dezember 2019
Einzureichen am:	31. Mai 2020
Verantw. Hochschullehrer:	Prof. Dr.-Ing. habil. Heiko Vogler
Betreuer:	Thomas Ruprecht, M. Sc.

Statistische Übersetzungssysteme. Beim *statistischen maschinellen Übersetzen* konstruiert und bewertet man auf *datengetriebene* Art Systeme zum automatischen Übersetzen von Texten, zum Beispiel vom Deutschen ins Englische. Datengetriebenes Konstruieren bedeutet hier, dass automatisch formale Übersetzungsregeln (innerhalb eines vorher festgelegten Rahmens) aus vorliegenden menschlichen Übersetzungen inferiert werden. Das Übersetzungssystem verwendet dann diese Regeln, um für einen beliebigen gegebenen Satz der Quellsprache eine potentiell unendliche, aber endlich repräsentierte Menge möglicher Übersetzungen (Kandidaten) zu generieren. Schließlich inferiert es aus dieser Menge eine finale Übersetzung für die Ausgabe.

Streaming-String-Transducer. *Streaming-String-Transducer* (*SST*s) [Alu10; AD11; Boj14] sind endliche Automaten, die neben den üblichen Zuständen, eine feste Anzahl von Registern als Speicher verwenden. In jeder Transition wird eine Manipulation der Register durchgeführt, die die ursprünglichen Registerinhalte nicht kopieren, wohl aber neue Symbole hinzufügen, Inhalte neu arrangieren oder löschen darf. Der Inhalt eines

festgelegten Registers nach den ausgeführten Manipulationen einer Reihe von Transitionen, die ein Eingabewort erkennen, wird als *Ausgabe* für das erkannte Wort definiert. Bei *nichtdeterministischen SSTs* (*NSST*s) ist der zugrundeliegende Automat nichtdeterministisch.

Aufgaben. Die Extraktion eines gewichteten NSSTs soll in den folgenden Schritten implementiert werden. Die einzelnen Schritte sollen ausreichend mit Tests verifiziert werden. Falls für die Ausführung eines Schrittes Parameter zu wählen sind, sollen diese ausreichend beschrieben werden.

1. Herr Jenke soll sich mit dem Europarl-Korpus[1] [Koe05] vertraut machen. Es soll ein Paar aus Quell- und eine Zielsprache gewählt werden (z.B. Englisch und Französisch), das im Korpus abgebildet wird. Ein (Teil-)Korpus, der parallele Sätze der gewählten Quell- und Zielsprache enthält, soll für die Weiterverarbeitung sinnvoll vorbereitet werden:
   - Artefakte, die keine natürliche Sprache darstellen, sollen entfernt werden.
   - Die Sätze sollen in Tokens (Wörter, Satzzeichen, …) aufgeteilt werden.
2. Herr Jenke soll einen endlichen Automaten aus den Sätzen der Quellsprache extrahieren. Dazu soll ein Hidden-Markov-Model auf den Sätzen der Quellsprache *unsupervised* trainiert werden [Bau+70; Rab89]. Für dieses Hidden-Markov-Model soll ein gewichteter endlicher Automat konstruiert werden, der die gleiche Sprache erkennt.
3. Herr Jenke soll Alignments zwischen den parallelen Sätzen der Quell- und Zielsprache erzeugen. Diese Alignments müssen für den folgenden Schritt total sein, d.h. jedes Wort des Satzes in der Zielsprache soll mit mindesten einem Wort der Quellsprache verbunden sein. Dazu soll eine geeignete Methode selbstständig gewählt werden, denkbare Möglichkeiten sind
   - GIZA++[2] [ON03] oder mGIZA[3],
   - fast_align[4] [DCA13], oder
   - efmaral[5] [ÖT16].
4. Mittels des aufbereiteten bilingualen Korpus, des endlichen Automaten und der Alignments soll nun ein nichtdeterministischer Streaming-String-Transducer extrahiert werden. Dafür sind geeignete Datenstrukturen zu wählen, die möglichst unabhängig von den gewählten Methoden sind. Der NSST soll in einem lesbaren Datenformat gespeichert werden können.

[1] `http://www.statmt.org/europarl/`
[2] `https://github.com/moses-smt/giza-pp`
[3] `https://github.com/moses-smt/mgiza`
[4] `https://github.com/clab/fast_align`
[5] `https://github.com/robertostling/efmaral`

Die Implementierung ist in einem Bericht, der den unter „Form“ aufgeführten Ansprüchen genügt, zu dokumentieren. Etwa in der Mitte der Bearbeitungszeit soll ein Statusvortrag, der über den aktuellen Stand des Praktikums berichtet, im Rahmen des Freitagsseminars gehalten werden. Das Praktikum wird mit der mündlichen Modulprüfung abgeschlossen.

Form. Die Arbeit muss den üblichen Standards wie folgt genügen. Die Arbeit muss in sich abgeschlossen sein und alle nötigen Definitionen und Referenzen enthalten. Die Urheberschaft von Inhalten – auch die eigene – muss klar erkennbar sein. Fremde Inhalte, z.,B. Algorithmen, Konstruktionen, Definitionen, Ideen, etc., müssen durch genaue Verweise auf die entsprechende Literatur kenntlich gemacht werden. Lange wörtliche Zitate sollen vermieden werden. Gegebenenfalls muss erläutert werden, inwieweit und zu welchem Zweck fremde Inhalte modifiziert wurden. Die Struktur der Arbeit muss klar erkenntlich sein, und der Leser soll gut durch die Arbeit geführt werden. Die Darstellung aller Begriffe und Verfahren soll mathematisch formal fundiert sein. Für jeden wichtigen Begriff sollen Erläuterungen und Beispiele angegeben werden, ebenso für die Abläufe der beschriebenen Verfahren sowie Konstruktionen. Wo es angemessen ist, sollen Illustrationen die Darstellung vervollständigen. Bei Diagrammen, die Phänomene von Experimenten beschreiben, muss deutlich erläutert werden, welche Werte auf den einzelnen Achsen aufgetragen sind, und beschrieben werden, welche Abhängigkeit unter den Werten der verschiedenen Achsen dargestellt ist.

Für die Implementierung soll eine ausführliche Dokumentation erfolgen, die sich angemessen auf den Quelltext und die schriftliche Ausarbeitung verteilt. Alle Abhängigkeiten und Installationsschritte sind lückenlos zu dokumentieren. Falls plattformunabhängig möglich soll die Installation in einem Installationsskript zu automatisiert sein. Andernfalls soll ein Container abgegeben werden, der das Programm in einer plattformunabhängig ausführbaren Form vorhält. Die Funktionsfähigkeit des Programms muss glaubhaft gemacht und durch geeignete Beispielläufe dokumentiert werden. Um die Reproduzierbarkeit aller dokumentierten Experimente zu gewährleisten, sind diese einschließlich aller Vorverarbeitungsschritte in einem Skript zu automatisieren. Falls die Ausführung aller Experimente sehr zeitaufwendig ist, soll das Skript so parametrisiert sein, dass einzelne Experimente isoliert gestartet werden können, mehrfach genutzte Zwischenergebnisse gespeichert werden und Experimente auf einem Teil der Datenmenge möglich sind. Die Parametrisierung des Skripts und Referenzergebnisse auf Teildaten sollen in diesen Fällen auch in der Arbeit dokumentiert werden. Einer späteren Veröffentlichung der Implementierung unter einer „Freie Software“-Lizenz stimmt Herr Jenke zu.

Dresden, 27. Mai 2020

Unterschrift von Heiko Vogler	Unterschrift von Alexander Caspar Jenke

Literatur

[AD11] Rajeev Alur und Jyotirmoy V Deshmukh. „Nondeterministic streaming string transducers“. In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2011, S. 1–20.

[Alu10] Rajeev Alur. „Expressiveness of streaming string transducers“. In: (2010).

[Bau+70] Leonard E Baum, Ted Petrie, George Soules und Norman Weiss. „A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains“. In: *The annals of mathematical statistics* 41.1 (1970), S. 164–171.

[Boj14] Mikołaj Bojańczyk. „Transducers with origin information“. In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2014, S. 26–37.

[DCA13] Chris Dyer, Victor Chahuneau und Noah A Smith. „A simple, fast, and effective reparameterization of ibm model 2“. In: (2013).

[Koe05] Philipp Koehn. „Europarl: A parallel corpus for statistical machine translation“. In: *MT summit*. Bd. 5. Citeseer. 2005, S. 79–86.

[ON03] Franz Josef Och und Hermann Ney. „A Systematic Comparison of Various Statistical Alignment Models“. In: *Computational Linguistics* 29.1 (2003), S. 19–51.

[ÖT16] Robert Östling und Jörg Tiedemann. „Efficient word alignment with Markov Chain Monte Carlo“. In: *Prague Bulletin of Mathematical Linguistics* 106 (Okt. 2016), S. 125–146. URL: `http://ufal.mff.cuni.cz/pbml/106/art-ostling-tiedemann.pdf`.

[Rab89] Lawrence R Rabiner. „A tutorial on hidden Markov models and selected applications in speech recognition“. In: *Proceedings of the IEEE* 77.2 (1989), S. 257–286.