Abschlussaufgabe „Wortvorhersage“

Praktikum Haskell für NLP – Sommersemester 2017

Tobias Denkinger

2017-05-23

Wortvorhersage

Das Ziel bei der *Wortvorhersage* ist es, ein gegebenes Präfix p eines natürlichsprachigen Satzes durch ein Wort w so zu erweitern, dass pw wieder ein Präfix eines natürlichsprachigen Satzes ist. Um w zu finden, wird ein Sprachmodell konsultiert, z.B. ein n-Gramm-Modell oder ein Hidden-Markov-Modell. Dabei wird w so gewählt, dass die Güte des Präfixes pw eines natürlichsprachigen Satzes unter dem Sprachmodell maximal ist. Die Güte wird üblicherweise als Wahrscheinlichkeit ausgedrückt. Für die praktische Anwendung (zum Beispiel während des Schreibens einer SMS-Nachricht) ist es manchmal sinnvoll, mehrere Vorschläge für das Wort w zu unterbreiten.

Aufgabe

Implementieren Sie ein Haskellprogramm zur Wortvorhersage in Texten. Das Programm soll eine natürliche Zahl k, ein n-Gramm-Modell, eine Textdatei und eine Position in der Textdatei entgegennehmen. Es soll dann eine Liste der besten k Worte ausgeben, die hinsichtlich des n-Gramm-Modells jeweils auf die angegebene Position im Text folgen können. Dafür soll nur das Präfix des Texts bis zur angegebenen Position in Betracht gezogen werden. Das n-Gramm-Modell [CG96] ist im *ARPA-Format*[1] gegeben (siehe Veranstaltungswebseite).

Das Programm soll folgende Kommandozeilensyntax haben:

```
NGramPredict <number> <model> <file> <line> <column>
```

Dabei ist `NGramPredict` der Name des Programms und die Parameter haben folgende Bedeutungen:

- `<number>` legt die Anzahl an Wortvorschläge fest, die das Programm unterbreitet.
- `<model>` legt den (möglicherweise relativen) Pfad zum n-Gramm-Modell fest.

[1] `http://www1.icsi.berkeley.edu/Speech/docs/HTKBook3.2/node213_mn.html`

- `<file>` legt den (möglicherweise relativen) Pfad zur Datei fest, die den zu ergänzenden Text enthält.
- `<line>` und `<column>` spezifizieren die Position im Text, wobei `<line>` die Zeilennummer und `<column>` die Spaltennummer ist; die Zählung der Zeilen und Spalten beginnt mit 1.

Anforderungen

An das Programm und seinen Quellcode werden folgende Anforderungen gestellt:

- Bei Ausführung mit fehlerhaften Kommandozeilenargumenten sollen sinnvolle Fehlermeldungen oder kurze allgemeine Benutzungshinweise ausgegeben werden.
- Der Quellcode soll in einem angemessenen Stil verfasst werden.
- Top-Level-Deklarationen sollen in Haddock-Syntax dokumentiert werden.
- Compilerwarnungen, die mit der Compileroption `-Wall`[2] ausgegeben werden, sollen behoben werden. Diese Option wird sowohl von `ghc` als auch von `ghci` unterstützt.
- Hinweise von `hlint`[3] sollen weitgehend umgesetzt werden.
- Das Projekt soll eine Paketbeschreibung für `cabal`[4] enthalten.
  - Mittels `cabal init` lässt sich leicht eine Paketbeschreibung generieren. Sollten bereits Quelldateien vorhanden sein, werden deren Abhängigkeiten beachtet.
  - Am Ende soll sich das Projekt mit folgenden Befehlen übersetzen und ausführen lassen:

    ```
    cabal sandbox init
    cabal install
    cabal exec <executable>
    ```

Literatur

[CG96] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, 13(4):359–393, 1996. DOI: `10.1006/csla.1999.0128`.

[2] `https://downloads.haskell.org/~ghc/latest/docs/html/users_guide/using-warnings.html#ghc-flag--Wall`
[3] `https://hackage.haskell.org/package/hlint`
[4] `https://www.haskell.org/cabal/`