Projektplan

Thema:

* Tippfehlerkorrektur/Autokorrektur: Tippfehler in freien Texten sollen erkannt und korrigiert werden, die korrekten Wörter müssen dazu im Vorhinein bereits vorliegen/ eingelesen werden damit im Nachhinein festgestellt werden kann, wie erfolgreich die Korrektur war. Es soll ein Vergleich zwischen verschiedenen Algorithmen sattfinden.

Benötigte Daten:

* Lexikon: Wird selbst geschrieben und ist vom Nutzer erweiterbar
* Korrekte Texte (zu Anfang sollen die Trainingstexte eher Kurz sein, also ca.100 – 200 Wörter lang)
  + Es muss ein korrekter Ausgangstext ohne Fehler vorliegen. Dieser wird benötigt um den Erfolg der Algorithmen zu berechnen
    - Wie viele Fehler wurden erkannt
    - Wie viele der fehlerhaften Wörter wurden richtig korrigiert
  + Hierbei kann auf die Tabellen (Gold Labels) von Naive Bayes zurückgegriffen werden
* Methode: Wortabgleich
* Textbeispiel: <http://www.writerswrite.com/books/excerpts/inkheart-excerpt-400014>
* Fehlerhafte Texte:
  + Der richtig vorliegende Text muss intellektuell so verändert werden, dass Fehler enthalten sind. Hierbei muss beachtet werden, dass es verschiedene Arten von Fehlern gibt (die auch besonders Häufig sind)
    - Buchstabe fehlt
    - Buchstabe zu viel
    - Buchstaben verdreht
    - Leerzeichen falsch (zu viel oder zu wenig)
    - Andere Fehler

Methoden (Diese können im Programm umgesetzt werden):

* N-Grams: Es wird kontrolliert, wie viele n-Gramme (z.B. Trigramme) von Wörtern miteinander übereinstimmen
* Acquaintance: Ähnlichkeit von Vektoren (Ein Wort wird in Form von einem Vektor repräsentiert)
* Damerau–Levenshtein distance: Misst die Anzahl der Veränderungen um einen String in einen anderen (den richtigen) zu verwandeln
  + Beispiel: <http://stackoverflow.com/questions/13928155/spell-checker-for-python>
  + <http://norvig.com/spell-correct.html>

Aufgaben:

* Textbearbetung:
  + Einlesen
  + Säubern
  + In Wörter zertrennen
  + In Teile Zertrennen, mit denen später gearbeitet werden kann (N-Grams, Buchstaben o.ä.)
  + Beispiel:
    - import string
    - clean = []
    - with open('book.txt') as f: // .txt einlesen
    - data = f.readlines()
    - for element in data:
    - s = element.translate(None, string.punctuation) // Punktation entfernen
    - clean.append(s)
* Linguistische Methoden anwenden: Die implementierten Methoden sollen auf einen fehlerhaften Text angewendet werden.
  + Fehlererkennung
  + Fehlerkorrektur
  + Darstellung der Ergebnisse (Gesamt oder einzelne Wörter mit ihrer Korrektur)
* Nutzen der verschiedenen Algorithmen
* Ausgabe erstellen:
  + Wie viele Worte sind Falsch
  + Wie sehen sie (wahrscheinlich) richtig aus
  + Dies soll in einer kleinen GUI dargestellt werden
* Darstellung jeweils gesplittet zwischen den einzelnen Methoden
  + Vergleich zwischen den Algorithmen:
    - Geschwindigkeit (dazu Graphen)
    - War das Wort wirklich richtig
    - Vergleich mit Mechanismen aus LibreOffice 🡪 Effizienz und Geschwindigkeit in Diagrammen (z.B. Balken)
    - Darstellung aller Wortvorschläge in GUI Ausgabe 🡪 Erfolgsmessung darstellen
* Eigene Worte dem Lexikon hinzufügen:
  + Button in der Haupt-GUI, die in zweites Fenster leitet 🡪 hier ist das Hinzufügen eigener Wörter möglich
    - Textfeld: Hier kann ein neuer Lexikoneintrag hineingeschrieben werden
    - Hinzufügen – Button: Hiermit wird das Wort im Textfeld in das Textdokument hinzugefügt, indem sich das bisherige Lexikon befindet
* Methodenvergleich und Analyse: Welche der Methoden ist wie erfolgreich, wie viele Wörter werden richtig korrigiert. Dies kann in Form von Graphen dargestellt werden
* Analyse: Werden die Methoden heute noch genutzt, ist es sinnvoll sie anzuwenden?
* Visualisierung: z.B. mit Matplotlib direkt mit ausgeben
  + als fasch erkannte Wörter
  + Wörter die richtig als falsch erkannt wurden 🡪 Gold Labels
  + Vergleich der Methoden z.B. als Balkendiagramme

Technische Details:

* Programmiersprache: Python
* Visualisierung: Matplotlib
* GUI: Tkinter

